# TABLE OF CONTENTS

## Volume 1

1. Commands and Applications Programs

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intro(1)</td>
<td>Introduction to commands and application programs</td>
</tr>
<tr>
<td>300(1)</td>
<td>Handle special functions of DASI 300 and 300s terminals</td>
</tr>
<tr>
<td>4014(1)</td>
<td>Paginator for the Tektronix 4014 terminal</td>
</tr>
<tr>
<td>450(1)</td>
<td>Handle special functions of the DASI 450 terminal</td>
</tr>
<tr>
<td>acctcom(1)</td>
<td>Search and print process accounting file(s)</td>
</tr>
<tr>
<td>adb(1)</td>
<td>Absolute debugger</td>
</tr>
<tr>
<td>admin(1)</td>
<td>Create and administer SCCS files</td>
</tr>
<tr>
<td>ar(1)</td>
<td>Archive and library maintainer for portable archives</td>
</tr>
<tr>
<td>as(1)</td>
<td>Common assembler</td>
</tr>
<tr>
<td>ascvt(1)</td>
<td>Interpret ASA carriage control characters</td>
</tr>
<tr>
<td>at(1)</td>
<td>Execute commands at a later time</td>
</tr>
<tr>
<td>ati(1)</td>
<td>Read and write ANSI format tapes</td>
</tr>
<tr>
<td>audfmt(1)</td>
<td>Format audit trail log data</td>
</tr>
<tr>
<td>audit(1)</td>
<td>Modify/query audit trail security status</td>
</tr>
<tr>
<td>awk(1)</td>
<td>Pattern scanning and processing language</td>
</tr>
<tr>
<td>backup(1)</td>
<td>Backup, restore - backup or restore selected files</td>
</tr>
<tr>
<td>banner(1)</td>
<td>Make posters</td>
</tr>
<tr>
<td>basename(1)</td>
<td>Deliver portions of path names</td>
</tr>
<tr>
<td>bc(1)</td>
<td>Arbitrary-precision arithmetic language</td>
</tr>
<tr>
<td>bdiff(1)</td>
<td>Big diff</td>
</tr>
<tr>
<td>bfs(1)</td>
<td>Big file scanner</td>
</tr>
<tr>
<td>bs(1)</td>
<td>A compiler/interpreter for modest-sized programs</td>
</tr>
<tr>
<td>cal(1)</td>
<td>Print calendar</td>
</tr>
<tr>
<td>calendar(1)</td>
<td>Reminder service</td>
</tr>
<tr>
<td>cancel(1)</td>
<td>Cancel requests to an LP line printer</td>
</tr>
<tr>
<td>cat(1)</td>
<td>Concatenate and print files</td>
</tr>
<tr>
<td>cb(1)</td>
<td>C program beautifier</td>
</tr>
<tr>
<td>cc(1)</td>
<td>C compiler</td>
</tr>
<tr>
<td>cd(1)</td>
<td>Change working directory</td>
</tr>
<tr>
<td>cdc(1)</td>
<td>Change the delta commentary of an SCCS delta</td>
</tr>
<tr>
<td>cflow(1)</td>
<td>Generate C flow graph</td>
</tr>
<tr>
<td>chfn(1)</td>
<td>Change user login name</td>
</tr>
<tr>
<td>chmod(1)</td>
<td>Change mode</td>
</tr>
<tr>
<td>chown(1)</td>
<td>Change owner or group</td>
</tr>
<tr>
<td>chsh(1)</td>
<td>Change login shell</td>
</tr>
<tr>
<td>clear(1B)</td>
<td>Clear terminal screen</td>
</tr>
<tr>
<td>cmp(1)</td>
<td>Compare two files</td>
</tr>
<tr>
<td>col(1)</td>
<td>Filter reverse line-feeds</td>
</tr>
</tbody>
</table>
comb(1) ...................................................... combine SCCS deltas
comm(1) ..................................................... select or reject lines common to two sorted files
cp(1) .......................................................... copy, link or move files
cpio(1) .......................................................... copy file archives in and out
cpp(1) .......................................................... the C language preprocessor
crontab(1) .................................................... user crontab file
crypt(1) ....................................................... encode/decode
csh(1B) ....................................................... a shell (command interpreter) with C-like syntax
csplit(1) ...................................................... context split
ctags(1B) ..................................................... create a tags file
ctrace(1) ..................................................... ctrace, compile, and optionally run a C program
currence(1) .................................................... C program debugger
cu(1C) .......................................................... call another UNIX system
cut(1) .......................................................... cut out selected fields of each line of a file
cw(1) .......................................................... prepare constant-width text for troff
cxref(1) ...................................................... generate C program cross-reference
date(1) .......................................................... print and set the date
dc(1) .......................................................... desk calculator
dd(1) .......................................................... convert and copy a file
delta(1) .......................................................... make a delta (change) to an SCCS file
deroff(1) ..................................................... remove nroff/troff, tbl, and eqn constructs
diff(1) .......................................................... differential file comparator
diff3(1) .......................................................... 3-way differential file comparison
diffmk(1) ...................................................... mark differences between files
dircmp(1) ..................................................... directory comparison
dis(1) .......................................................... disassembler
du(1) .......................................................... summarize disk usage
dump(1) ....................................................... dump selected parts of an object file
echo(1) .......................................................... echo arguments
ed(1) .......................................................... text editor (variant of ex for casual users)
efl(1) .......................................................... Extended Fortran Language
enable(1) ..................................................... enable/disable LP printers
env(1) .......................................................... set environment for command execution
eqn(1) .......................................................... format mathematical text for nroff or troff
erase(1) .......................................................... erase a cartridge tape
eror(1B) ..................................................... analyze and disperse compiler error messages
rvt(1) .......................................................... screen text editor
ex(1) .......................................................... text editor
expr(1) .......................................................... evaluate arguments as an expression
f77(1) .......................................................... Fortran 77 compiler
factor(1) ...................................................... factor a number
file(1) .......................................................... determine file type
find(1) .......................................................... find files
finger(1) ...................................................... user information lookup program
fsplit(1) ...................................................... split f77 or ratfor files
gcore(1) ...................................................... get core images of running processes
gdev(1G) ..................................................... graphical device routines and filters
ged(1G) ..................................................... graphical editor
gence(1) ...................................................... create a front end to the cc command
Contents

get(1) .................................................. get a version of an SCCS file
getopt(1) ................................................ parse command options
glossary(1) ........................................... definitions of common UNIX system terms and symbols
gprof(1) ................................................ display call graph profile data
graph(1G) ................................................ draw a graph
graphics(1G) ........................................... access graphical and numerical commands
greek(1) ................................................ select terminal filter
grep(1) ................................................ search a file for a pattern
gsar(1) ................................................ graphical system activity reporter
gutil(1G) ............................................... graphical utilities
head(1B) .............................................. give first few lines
help(1) .................................................. UNIX system Help Facility
hp(1) .................................................. handle special functions of HP 2640 and 2621-series terminals
hpio(1) .................................................. HP 2645A terminal tape file archiver
hyphen(1) ................................................ find hyphenated words
id(1) .................................................. print user and group IDs and names
ifilter(1) ................................................ international line printer filter
lp(1) .................................................. send requests to line printer using int'l character set
iostat(1) ................................................ report I/O statistics
ipcrm(1) ................................................ remove message queue, semaphore set or shared memory id
ipcs(1) .................................................. report inter-process communication facilities status
isort(1) .................................................. sort and/or merge files
join(1) .................................................. relational database operator
kill(1) .................................................. terminate a process
last(1B) .................................................. indicate last logins of users and teletypes
ld(1) .................................................. link editor for common object files
level(1) .................................................. display system and product level information
lex(1) .................................................. generate programs for simple lexical tasks
line(1) .................................................. read one line
lint(1) .................................................. a C program checker
list(1) .................................................. produce C source listing from object-file
locate(1) ............................................. identify a UNIX system command using keywords
login(1) .................................................. sign on
logname(1) ............................................. get login name
look(1B) ................................................ find lines in a sorted list
lorder(1) ............................................. find ordering relation for an object library
lp(1) .................................................. send requests to an LP line printer
lpr(1) .................................................. line printer spooler
lps(1) .................................................. set parallel printer characteristics
lpstat(1) ............................................. print LP status information
ls(1) .................................................. list contents of directory
ls(1B) .................................................. list contents of directory (Berkeley)
[This page left blank.]
# TABLE OF CONTENTS

## Volume 2

1. Commands and Applications Programs (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m4(I)</td>
<td>macro processor</td>
</tr>
<tr>
<td>machid(I)</td>
<td>provide truth value about your processor type</td>
</tr>
<tr>
<td>mail(I)</td>
<td>send mail to users or read mail</td>
</tr>
<tr>
<td>mailx(I)</td>
<td>interactive message processing system</td>
</tr>
<tr>
<td>make(I)</td>
<td>maintain, update, and regenerate groups of programs</td>
</tr>
<tr>
<td>makekey(I)</td>
<td>generate encryption key</td>
</tr>
<tr>
<td>man(I)</td>
<td>print entries in this manual</td>
</tr>
<tr>
<td>mcs(I)</td>
<td>manipulate the object file comment section</td>
</tr>
<tr>
<td>msg(I)</td>
<td>permit or deny messages</td>
</tr>
<tr>
<td>mkdir(I)</td>
<td>make a directory</td>
</tr>
<tr>
<td>mklost+found(I)</td>
<td>create an error message file by massaging C source</td>
</tr>
<tr>
<td>mkstr(I)</td>
<td>print/check documents formatted with the MM macros</td>
</tr>
<tr>
<td>mmt(I)</td>
<td>typeset documents, viewgraphs, and slides</td>
</tr>
<tr>
<td>more(I)</td>
<td>file perusal filter for crt viewing</td>
</tr>
<tr>
<td>newform(I)</td>
<td>change the format of a text file</td>
</tr>
<tr>
<td>newgrp(I)</td>
<td>log in to a new group</td>
</tr>
<tr>
<td>news(I)</td>
<td>print news items</td>
</tr>
<tr>
<td>nice(I)</td>
<td>run a command at low priority</td>
</tr>
<tr>
<td>nl(I)</td>
<td>line numbering filter</td>
</tr>
<tr>
<td>nm(I)</td>
<td>print name list of common object file</td>
</tr>
<tr>
<td>nohup(I)</td>
<td>run a command immune to hangups and quits</td>
</tr>
<tr>
<td>nroff(I)</td>
<td>format or typeset text</td>
</tr>
<tr>
<td>od(I)</td>
<td>octal dump</td>
</tr>
<tr>
<td>pack(I)</td>
<td>compress and expand files</td>
</tr>
<tr>
<td>packsf(I)</td>
<td>compress and uncompress sparse file</td>
</tr>
<tr>
<td>passwd(I)</td>
<td>change login password</td>
</tr>
<tr>
<td>paste(I)</td>
<td>merge same lines of several files or subsequent lines of a file</td>
</tr>
<tr>
<td>pcdsk(I)</td>
<td>PC-DOS to UNIX file transfer</td>
</tr>
<tr>
<td>pg(I)</td>
<td>file perusal filter for screen terminals</td>
</tr>
<tr>
<td>pr(I)</td>
<td>print files</td>
</tr>
<tr>
<td>print(I)</td>
<td>line printer spooler</td>
</tr>
<tr>
<td>prof(I)</td>
<td>display profile data</td>
</tr>
<tr>
<td>prs(I)</td>
<td>print an SCCS file</td>
</tr>
<tr>
<td>ps(I)</td>
<td>report process status</td>
</tr>
<tr>
<td>ptx(I)</td>
<td>permuted index</td>
</tr>
<tr>
<td>pwd(I)</td>
<td>working directory name</td>
</tr>
<tr>
<td>ratfor(I)</td>
<td>rational Fortran dialect</td>
</tr>
<tr>
<td>regcmp(I)</td>
<td>regular expression compile</td>
</tr>
<tr>
<td>rev(I)</td>
<td>reverse lines of a file</td>
</tr>
</tbody>
</table>

UP-11760 R2B, V2
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rm()</td>
<td>remove files or directories</td>
</tr>
<tr>
<td>rmdel()</td>
<td>remove a delta from an SCCS file</td>
</tr>
<tr>
<td>rz()</td>
<td>XMODEM, YMODEM, ZMODEM (batch) file receive</td>
</tr>
<tr>
<td>sact()</td>
<td>print current SCCS file editing activity</td>
</tr>
<tr>
<td>sag()</td>
<td>system activity graph</td>
</tr>
<tr>
<td>sar()</td>
<td>system activity reporter</td>
</tr>
<tr>
<td>sccsdiff()</td>
<td>compare two versions of an SCCS file</td>
</tr>
<tr>
<td>script()</td>
<td>make typescript of terminal session</td>
</tr>
<tr>
<td>sdb()</td>
<td>symbolic debugger</td>
</tr>
<tr>
<td>sdiff()</td>
<td>side-by-side difference program</td>
</tr>
<tr>
<td>sed()</td>
<td>stream editor</td>
</tr>
<tr>
<td>setalign()</td>
<td>set/unset alignment emulation</td>
</tr>
<tr>
<td>setlp()</td>
<td>set parameters for a line printer type device (parallel)</td>
</tr>
<tr>
<td>set_tape()</td>
<td>change the logical tape size for tape device</td>
</tr>
<tr>
<td>sh()</td>
<td>shell, the standard/restricted command programming language</td>
</tr>
<tr>
<td>shl()</td>
<td>shell layer manager</td>
</tr>
<tr>
<td>show()</td>
<td>display current hardware configuration</td>
</tr>
<tr>
<td>size()</td>
<td>print section sizes of common object files</td>
</tr>
<tr>
<td>sleep()</td>
<td>suspend execution for an interval</td>
</tr>
<tr>
<td>sln()</td>
<td>link files symbolically</td>
</tr>
<tr>
<td>sno()</td>
<td>SNOBOL interpreter</td>
</tr>
<tr>
<td>sort()</td>
<td>sort and/or merge files</td>
</tr>
<tr>
<td>spell()</td>
<td>find spelling errors</td>
</tr>
<tr>
<td>spline()</td>
<td>interpolate smooth curve</td>
</tr>
<tr>
<td>split()</td>
<td>split a file into pieces</td>
</tr>
<tr>
<td>spool()</td>
<td>spool queue manager</td>
</tr>
<tr>
<td>ssp()</td>
<td>make output single spaced</td>
</tr>
<tr>
<td>starter()</td>
<td>information about the UNIX system for beginning users</td>
</tr>
<tr>
<td>stat()</td>
<td>statistical network useful with graphical commands</td>
</tr>
<tr>
<td>strings()</td>
<td>find the printable strings in an object, or other binary, file</td>
</tr>
<tr>
<td>strip()</td>
<td>strip symbol and line number information from object file</td>
</tr>
<tr>
<td>stty()</td>
<td>set the options for a terminal</td>
</tr>
<tr>
<td>su()</td>
<td>become superuser or another user</td>
</tr>
<tr>
<td>sum()</td>
<td>print checksum and block count of a file</td>
</tr>
<tr>
<td>sync()</td>
<td>update the super block</td>
</tr>
<tr>
<td>sz()</td>
<td>XMODEM, YMODEM, ZMODEM batch file send</td>
</tr>
<tr>
<td>tabs()</td>
<td>set tabs on a terminal</td>
</tr>
<tr>
<td>tail()</td>
<td>deliver the last part of a file</td>
</tr>
<tr>
<td>tape_size()</td>
<td>print the logical tape size to standard out</td>
</tr>
<tr>
<td>tar()</td>
<td>tape file archiver</td>
</tr>
<tr>
<td>tbl()</td>
<td>format tables for nroff or troff</td>
</tr>
<tr>
<td>tc()</td>
<td>phototypesetter simulator</td>
</tr>
<tr>
<td>tee()</td>
<td>copy input to standard output and to files</td>
</tr>
<tr>
<td>tension()</td>
<td>tension a cartridge tape</td>
</tr>
<tr>
<td>test()</td>
<td>condition evaluation command</td>
</tr>
<tr>
<td>time()</td>
<td>time a command</td>
</tr>
<tr>
<td>timex()</td>
<td>time a command; report process data and system activity</td>
</tr>
<tr>
<td>toc()</td>
<td>graphical table of contents routines</td>
</tr>
<tr>
<td>touch()</td>
<td>update access and modification times of a file</td>
</tr>
<tr>
<td>tpcvt()</td>
<td>filter for old streaming tape format</td>
</tr>
</tbody>
</table>
tpplot(1G) ........................................ graphics filters
tps(1) ........................................ show processes use of GPTF
tput(1) ........................................ query terminfo database
tr(1) ........................................ translate characters
true(1) ........................................ provide truth values
tsort(1) ........................................ topological sort
tty(1) ........................................ get the name of the terminal
ul(1B) .......................................... underline output for a terminal
ulim(1) ........................................ increase maximum file size limit
umask(1) ...................................... set file-creation mode mask
uname(1) ...................................... print name of current UNIX system
unget(1) ...................................... undo a previous get of an SCCS file
uniq(1) ...................................... report repeated lines in a file
units(1) ...................................... interactive conversion program
uptime(1) ..................................... show how long system has been up
usage(1) ...................................... retrieve a command description and usage examples
uucp(1C) ...................................... UNIX system to UNIX system copy
uustat(1C) ..................................... uucp status inquiry and job control
uuto(1C) ...................................... public UNIX-to-UNIX system file copy
uux(1C) ....................................... UNIX-to-UNIX system command execution
val(1) .......................................... validate SCCS file
vc(1) .......................................... version control
version(1) ..................................... display release identifications of installed software
vi(1) .......................................... screen-oriented (visual) display editor based on ex
vmstat(1) ..................................... report virtual memory statistics
vpr(1) .......................................... Versatec printer spooler
vs(1) .......................................... report statistics of major subsystems
w(1) .......................................... who is on and what they are doing
wait(1) ........................................ await completion of process
wc(1) .......................................... word count
what(1) ........................................ identify SCCS files
whereis(1) .................................... locate source, binary, and or manual for program
who(1) .......................................... who is on the system
whoami(1) .................................... print effective current user id
write(1) ........................................ write to another user
xargs(1) ...................................... construct argument list(s) and execute command
xstr(1B) ........................................ extract strings from C prog. to implement shared strings
yacc(1) ........................................ yet another compiler-compiler

6. Games

intro(6) ........................................ introduction to games
arithmetic(6) ................................... provide drill in arithmetic problems
back(6) .......................................... the game of backgammon
bj(6) .......................................... the game of black jack
chess(6) ........................................ the game of chess
craps(6) ........................................ the game of craps
hangman(6) ..................................... guess the word
jotto(6) ........................................ secret word game
maze(6) ........................................ generate a maze
moo(6) .......................................... guessing game
quiz(6) ........................................ test your knowledge
Contents

reversi(6) ................................... a game of dramatic reversals
rogue(6) ................................... Exploring The Dungeons of Doom
sky(6) ........................................... obtain ephemerides
ttt(6) ........................................... tic-tac-toe
wump(6) ........................................... the game of hunt-the-wumpus
The names, places, and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living, or otherwise. Any similarity or likeness of the names, places and/or events with the names of any individual living or otherwise, or that of any group or association is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

DEC and PDP are trademarks of Digital Equipment Corporation. Heathkit is a registered trademark of Heath Company. HP is a registered trademark of Hewlett-Packard, Inc. IBM is a registered trademark of International Business Machines Corporation. PS-DOS is an additional trademark of International Business Machines Corporation. LEAR-SIEGLER, INC. is a registered trademark of LEAR-SIEGLER, INC. Microterm is a registered trademark of Micro-Term Corporation. Motorola is a registered trademark of Motorola, Inc. 68000, 68010, and 68020 are additional trademarks of Motorola Inc. MS-DOS is a registered trademark of Microsoft Corporation. Othello is a registered trademark of Gabriel Industries Inc. Q-Chart is a trademark of Quadratron Systems. SPERRY is a registered trademark of Unisys Corporation. Tektronix is a registered trademark of Tektronix, Inc. Teletype is a trademark of AT&T Teletype Corporation. Versatec is a registered trademark of Versatec Corporation. WANG is a registered trademark of Wang Laboratories Inc. XEROX is a registered trademark of Xerox Corporation. Diablo is an additional trademark of Xerox Corporation.

Copyright © 1985 by NCR Corporation
Copyright © 1985 by Computer Consoles, Inc.

Portions of this material are copyrighted © by AT&T Technologies and are reprinted with their permission.

This documentation is based in part on the fourth Berkeley Software Distribution, under license from the Regents of the University of California. We acknowledge the following individuals and institutions for their role in its development:

Computer Science Division
Department of Electrical Engineering and Computer Science
University of California
Berkeley, California 94720
convert between 3-byte integers and long integers \ldots 13tol
68881 control register access \ldots. access881
generate an abnormal process termination \ldots abort
Fortran absolute value \ldots \ldots abs
return integer absolute value \ldots \ldots abs
floor, ceiling, remainder, absolute value functions \ldots\ldots floor
accept a connection on a socket \ldots accept
68881 control register access register\ldots \ldots access881
set file access and modification times \ldots utime
GPTF access control file \ldots gptfctrl
floating point processor access long numeric data in a machine \ldots splutl
independent fashion access routines \ldots ldfcn
common object file access utmp file entry \ldots getut
accessibility of a file \ldots access
determine enable or disable process \ldots acct
per-process accounting \ldots acct
return Fortran time accounting \ldots acct
change or add value to environment \ldots putenv
set the process alarm clock for a process \ldots alarm
the calling process allocate a GPTF timer structure for \ldots tptmalloc
change data segment space allocation \ldots brk
fast main memory allocator \ldots malloc
main memory allocator \ldots malloc
set system power up time and/or immediately power down system \ldots syspwr
format of cpio archive \ldots cpio
common archive file format \ldots ar
the archive header of a member of an archive file read \ldots ldahread
Fortran imaginary part of complex argument \ldots almag
return Fortran command-line argument \ldots getarg
handle variable argument \ldots varargs
Permuted Index

print formatted output of a varargs argument list \ldots \ldots vprintf
get option letter from argument vector \ldots \ldots getopt
number of command line arguments \ldots \ldots argc
map of ASCII character set \ldots \ldots asciil
between long integer and base-64 ASCII string convert \ldots \ldots a64l
common assembler and link editor output \ldots \ldots a.out
verify program assertion \ldots \ldots assert
assign buffering to a stream \ldots \ldots setbuf
functions sinh881, cosh881, tanh881 \ldots \ldots trigh881
enable/disable audit trail facility \ldots \ldots audon
modify/query audit trail security status \ldots \ldots audit
query audit trail security status \ldots \ldots ckaudit
push character back into input stream \ldots \ldots ungetc
terminal capability data base \ldots \ldots btermcap
terminal capability data base \ldots \ldots termcap
terminal capability data base \ldots \ldots terminfo
convert between long integer and base-64 ASCII string \ldots \ldots a64l
Bessel functions \ldots \ldots bessel
binary input/output \ldots \ldots fread
binary search \ldots \ldots bsearch
manage binary search trees \ldots \ldots tsearch
bind a name to a socket \ldots \ldots bind
Fortran bitwise boolean functions \ldots \ldots bool
block operations \ldots \ldots block
Fortran bitwise boolean functions \ldots \ldots bool
standard buffered input/output package \ldots \ldots stdio
assign buffering to a stream \ldots \ldots setbuf
swap bytes \ldots \ldots swap
data returned by stat system \ldots \ldots stat
a GPTF timer structure for the calling process allocate \ldots \ldots tptmalloc
introduction to system \ldots \ldots intro
special system \ldots \ldots symsm68k
terminal capability data base \ldots \ldots btermcap
terminal capability data base \ldots \ldots termcap
terminal capability data base \ldots \ldots terminfo
functions floor, ceiling, remainder, absolute value \ldots floor
create an interprocess channel \ldots \ldots pipe
push character back into input stream \ldots \ldots ungetc
neqn special character definitions for eqn and eqnchar
neqn special character definitions for eqn and eqnchar
get character login name of the user \ldots \ldots cuserid
get character or word from stream \ldots \ldots getc
put character or word on a stream \ldots \ldots putc
map of ASCII character set \ldots \ldots asciil
translate characters \ldots \ldots conv
classify characters \ldots \ldots ctype
get process and child process times \ldots \ldots times
wait for child process to stop or terminate \ldots \ldots wait
classify characters \ldots \ldots ctype
<table>
<thead>
<tr>
<th>Function Call</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>set the process alarm</td>
<td>clock for a process</td>
</tr>
<tr>
<td>issue a shell command from Fortran</td>
<td>system</td>
</tr>
<tr>
<td>return Fortran command-line argument</td>
<td>getarg</td>
</tr>
<tr>
<td>output common assembler and link editor</td>
<td>a.out</td>
</tr>
<tr>
<td>open a common object file for reading</td>
<td>ldfcn</td>
</tr>
<tr>
<td>manipulate line number entries of a common object file function</td>
<td>ldread</td>
</tr>
<tr>
<td>read the file header of a section header for a common object file file header</td>
<td>ldfhread</td>
</tr>
<tr>
<td>seek to the optional file header of a common object file seek to</td>
<td>ldseek</td>
</tr>
<tr>
<td>relocation entries of a section of a common object file seek to</td>
<td>ldseek</td>
</tr>
<tr>
<td>an indexed/named section header of a common object file seek to</td>
<td>ldseek</td>
</tr>
<tr>
<td>an indexed symbol table entry of a common object file seek to</td>
<td>ldsseek</td>
</tr>
<tr>
<td>line number entries in a relocation information for a section header for a common object file</td>
<td>linenum</td>
</tr>
<tr>
<td>retrieve symbol name for a common object file symbol tbl entry</td>
<td>ldgetname</td>
</tr>
<tr>
<td>format file header for a common object file</td>
<td>filehdr</td>
</tr>
<tr>
<td>standard interprocess communication package</td>
<td>stdipc</td>
</tr>
<tr>
<td>create an endpoint for communication</td>
<td>socket</td>
</tr>
<tr>
<td>string comparison intrinsic functions</td>
<td>strcmp</td>
</tr>
<tr>
<td>expression compile and execute regular</td>
<td>regcmp</td>
</tr>
<tr>
<td>regular expression compile and match routines</td>
<td>regexp</td>
</tr>
<tr>
<td>format of compiled term file</td>
<td>term</td>
</tr>
<tr>
<td>error function and complementary error function</td>
<td>erf</td>
</tr>
<tr>
<td>Fortran imaginary part of complex argument</td>
<td>aimag</td>
</tr>
<tr>
<td>Fortran complex conjugate intrinsic function</td>
<td>conjg</td>
</tr>
<tr>
<td>of object file compute index of symbol table entry</td>
<td>ldtbindex</td>
</tr>
<tr>
<td>MPCC configuration file</td>
<td>mpccctab</td>
</tr>
<tr>
<td>Fortran complex conjugate intrinsic function</td>
<td>conjg</td>
</tr>
<tr>
<td>get name of connected peer</td>
<td>getpeername</td>
</tr>
<tr>
<td>create a pair of connected sockets</td>
<td>socketpair</td>
</tr>
<tr>
<td>establish an out-going terminal line connection</td>
<td>dial</td>
</tr>
<tr>
<td>accept a connection on a socket</td>
<td>accept</td>
</tr>
<tr>
<td>initiate a connection on a socket</td>
<td>connect</td>
</tr>
<tr>
<td>shut down part of a full-duplex connection</td>
<td>shutdown</td>
</tr>
<tr>
<td>listen for connections on a socket</td>
<td>listen</td>
</tr>
<tr>
<td>math functions and constants</td>
<td>math</td>
</tr>
<tr>
<td>unistd file header for symbolic constants</td>
<td>unistd</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
</tr>
<tr>
<td>loctl</td>
<td>fcntl</td>
</tr>
<tr>
<td>virtually “hangup” the current control terminal</td>
<td>conventionally for terminals term</td>
</tr>
<tr>
<td>sinh881, cosh881, tanh881, atanh881 - M8881, sinh881, cosh881, tanh881, atanh881</td>
<td>cpio archive</td>
</tr>
<tr>
<td>format of core image file</td>
<td>CPIO</td>
</tr>
<tr>
<td>_hyperbolic functions sinh881, cosh881, tanh881, atanh881</td>
<td>Cpio</td>
</tr>
<tr>
<td>terminal capability</td>
<td>btermcap</td>
</tr>
<tr>
<td>get/set unique identifier of</td>
<td>get/set name of</td>
</tr>
<tr>
<td>get/put name of</td>
<td>get/put name of</td>
</tr>
<tr>
<td>current UNIX system</td>
<td>current host</td>
</tr>
<tr>
<td>get name of</td>
<td>get name of</td>
</tr>
<tr>
<td>find the slot in the utmp file of the</td>
<td>get pathname of</td>
</tr>
<tr>
<td>get</td>
<td>get</td>
</tr>
<tr>
<td>screen functions with optimal cursor motion</td>
<td>data base</td>
</tr>
<tr>
<td>data returned by stat system call</td>
<td>data in memory</td>
</tr>
<tr>
<td>change data segment space allocation</td>
<td>brk</td>
</tr>
</tbody>
</table>

Index-4
<table>
<thead>
<tr>
<th>Function/Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operatin system</td>
<td>date and release number of the system</td>
</tr>
<tr>
<td>get date and time</td>
<td>gettimeofday</td>
</tr>
<tr>
<td>convert date and time to string</td>
<td>ctime</td>
</tr>
<tr>
<td>structure deallocates a specified GPTF timer</td>
<td>tptimdealloc</td>
</tr>
<tr>
<td>introduction to system calls</td>
<td>definitions, and error numbers</td>
</tr>
<tr>
<td>special character definitions for eqn and neqn</td>
<td>eqnchar</td>
</tr>
<tr>
<td>special character definitions for eqn and neqn</td>
<td>eqnchar</td>
</tr>
<tr>
<td>generate DES encryption</td>
<td>crypt</td>
</tr>
<tr>
<td>get disk description by its name</td>
<td>getdisk</td>
</tr>
<tr>
<td>disk description file</td>
<td>disktab</td>
</tr>
<tr>
<td>description of output language</td>
<td>troff</td>
</tr>
<tr>
<td>close a file</td>
<td>close</td>
</tr>
<tr>
<td>duplicate an open file</td>
<td>dup</td>
</tr>
<tr>
<td>duplicate a descriptor</td>
<td>dup2</td>
</tr>
<tr>
<td>get file system descriptor file</td>
<td>getfsent</td>
</tr>
<tr>
<td>determine accessibility of a file</td>
<td>access</td>
</tr>
<tr>
<td>paging/swapping add a swap</td>
<td>swapon</td>
</tr>
<tr>
<td>master device information table</td>
<td>master</td>
</tr>
<tr>
<td>control device</td>
<td>loctl</td>
</tr>
<tr>
<td>error records for devices exceeding threshold values</td>
<td>alert</td>
</tr>
<tr>
<td>dialup line file</td>
<td>dialups</td>
</tr>
<tr>
<td>dialup password file</td>
<td>d_passwd</td>
</tr>
<tr>
<td>positive difference intrinsic functions</td>
<td>dim</td>
</tr>
<tr>
<td>mount and unmount directories across file systems</td>
<td>rmnt</td>
</tr>
<tr>
<td>format of directories</td>
<td>dir</td>
</tr>
<tr>
<td>dirent-format of directories</td>
<td>dirent</td>
</tr>
<tr>
<td>change working directory</td>
<td>chdir</td>
</tr>
<tr>
<td>change root directory</td>
<td>chroot</td>
</tr>
<tr>
<td>remove directory entry</td>
<td>unlink</td>
</tr>
<tr>
<td>make a directory file</td>
<td>mkdir</td>
</tr>
<tr>
<td>remove a directory file</td>
<td>rmdir</td>
</tr>
<tr>
<td>get pathname of current working directory</td>
<td>getcwd</td>
</tr>
<tr>
<td>file make a directory, or a special or ordinary mknod</td>
<td>mknod</td>
</tr>
<tr>
<td>file make a directory, or a special or ordinary mknod</td>
<td>mknod</td>
</tr>
<tr>
<td>get current working directory</td>
<td>getwd</td>
</tr>
<tr>
<td>scan a directory</td>
<td>scandir</td>
</tr>
<tr>
<td>mounted directory table</td>
<td>rmnttab</td>
</tr>
<tr>
<td>dirent-format of directories</td>
<td>dirent</td>
</tr>
<tr>
<td>enter GPTF timer disable mode</td>
<td>tptimedisable</td>
</tr>
<tr>
<td>enable or disable process accounting</td>
<td>acct</td>
</tr>
<tr>
<td>get disk description by its name</td>
<td>getdisk</td>
</tr>
<tr>
<td>disk description file</td>
<td>disktab</td>
</tr>
<tr>
<td>Euclidean distance function</td>
<td>hypot</td>
</tr>
<tr>
<td>generate uniformly distributed pseudo-random numbers</td>
<td>drand48</td>
</tr>
<tr>
<td>the MM macro package for formatting adapter macro</td>
<td>mm</td>
</tr>
<tr>
<td>generate uniformly distributed pseudo-random numbers</td>
<td>documents</td>
</tr>
<tr>
<td>the OSDD</td>
<td>mosd</td>
</tr>
<tr>
<td>function</td>
<td>documents the OSDD</td>
</tr>
<tr>
<td>convert string to double-precision number</td>
<td>strtod</td>
</tr>
</tbody>
</table>
Permuted Index

Incremental dump format .................. fdump
duplicate a descriptor ................. dup2
duplicate an open file descriptor .. dup
editor output .......................... a.out
effective group IDs get real ........ getuid
effective user, real group, and .... getuid
effective group IDs get real user .. getrealuid
enter GPTF timer ....................... tptimenable
effective or disable process accounting acct
eable/disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon
generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon

generate DES encryption .................. crypt
crypt end point for communication ........ socket
duplicate a descriptor .................. dup
enable or disable process accounting acct
disable audit trail facility .... audon
duplicate an open file descriptor .. dup
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer ....................... tptimenable
duplicate file descriptor .............. dup2
duplicate file descriptor .............. dup
enter GPTF timer disable mode .... tptimdisable
disable audit trail facility .... audon

Index-8

UP-11762 R2A, V1
suspend execution for interval .......... sleep
prepare execution profile ............... monitor
evaluation time profile ................. profile
create a new file or rewrite an existing one .......... create
explicit Fortran type conversion .. ftype
root functions exponential, logarithm, pwr, sqr .. exp
intrinsic functions Fortran exponential, logarithm, square root exp
regular expr. compile and match routines ... regexp
compilation and execute regular expression ........... regcmp
graphics for the extended TTY-37 type-box .......... greek
numeric data in a machine independent fashion. access long .......... sputil
fast main memory allocator .......... malloc
set file access and modification times .. utime
determine accessibility of a file .............. access
common object file access routines .......... ldfcn
logalert summary message file ............... alertmsg
change mode of a file .................. chmod
change owner and group of a file .......... chown
file control options .................. fctl
file control .................. fctl
format of core image file ............... core
set and get file creation mask .......... umask
close a file descriptor ................ close
duplicate an open file descriptor .......... dup
dialup line file .................. dialups
disk description file ................. disktab
dialup password file .......... d_passwd
get file system descriptor file entry .......... getfsent
access utmp file entry .......... getut
write password file entry .......... putpwent
execute a file .................. exec
open a common object file for reading .......... ldopen
per-process accounting file format .......... acct
common archive file format .......... ar
error-log file format .......... errfile
introduction to file formats .............. intro
number entries of a common object file function manipulate line .......... ldiread
get group file ................. getgrent
GPTF access control file .......... gptfctrl
group file .................. group
unistd file header for common object files .. filehdr
read the file header of a common object file .. ldhdr
seek to the optional file header of a common object file .......... ldohseek
USER-DEFD, issue issue identification file .................. isort
header of a member of an archive file read the archive .......... ldahread
close a common object file .......... ldclose
the file header of a common object file read .......... ldhdr
of a section of a common object file seek to line number entries .......... ldlsseek
file header of a common object  file seek to the optional ......... ldohseek
of a section of a common object file seek to relocation entries ....... ldrseek
section header of a common object file read an indexed/named ......... ldshread
section of a common object file seek to an indexed/named ............ ldsseek
index of symbol table entry of object file compute ..................... ldttbindex
symbol table entry of a common object file read an indexed ............ ldttbseek
the symbol table of a common object file seek to ........................ ldtbseek
number entries in a common object file line ............................. linenum
link to a file file make .................................... mknod
a directory, or a special or ordinary file make ....................... mkdir
make a directory file make .................................... mknod
a directory, or a special or ordinary file make ....................... mknod
a directory, or a special or ordinary file make ....................... mknod
MPCC configuration file name for terminal ............................... ctermid
generate make a unique ..................................... mktemp
find the slot in the utmp file of the current user ..................... ttyslot
create a new file or rewrite an existing one ............................ creat
password file pointer in a stream .................................. fseek
reposition a file pointer ........................................... lseek
information for a common object file queues for relocation ......... queueinfo
read from file regions for reading and writing lockf
provide exclusive file rewrite an existing one........................ creat
change the name of a password file .................................. passwd
remove a directory file pointer in a stream ............................ fseek
move read/write file pointer ........................................ lseek
format of SCFS file name ............................................ scfs
section header for a common object file queues for relocation ......... queueinfo
get file status ....................................................... stat
synchronously write on file .......................................... swrite
symbol name for common object file symbol table entry retrieve .... ldgetname
common object file symbol table format ................................ sym
get file system descriptor file entry .................................. fsent
mount a file system ................................................. mount
get file system statistics ............................................. ustat
format of compiled term file system table .............................. mnttab
threshold - logalert file system statistics ............................. mnttab
create temporary file system .......................................... mnttab
create a temporary file system ........................................ mnttab
unmount a file system ................................................ remnt
list of file systems processed by fsck ................................ remnt
mount and unmount directories across file systems .................... slink
link files across file systems ....................................... slink
format of compiled term file .......................................... term
threshold - logalert file ............................................. threshold
create a temporary file ............................................. tmpfile
create a name for a temporary file to a specified length ............. tmpnam
truncate a file tree ................................................. ftw
write on a file ...................................................... write
link files across file systems ....................................... slink

Index-8
file header for common object files................................................. filehdr
format specification in text files.................................................. fspec
primitive string, format of graphical files..................................... gps
record locking on files..................................................................... lockf
record locking on files..................................................................... lockf
static information about the filesystems.......................................... fstab
find name of a terminal...................................................................... ttyname
current user find the slot in the utmp file of the .. ttyslot
set a process priority to a fixed value............................................. tpfix
convert floating-point number to string............................................ ecvt
manipulate parts of floating-point numbers..................................... frexp
value functions floor, ceiling, remainder, absolute........................... floor
close or flush a stream...................................................................... fclose
per-process accounting file format.................................................... acct
common archive file format............................................................. ar
float format conversions.................................................................... flcvt
error-log file format......................................................................... errfile
incremental dump format.................................................................... fdump
format of an inode............................................................................. inodc
format of compiled term file............................................................ term
format of core image file................................................................... core
format of cpio archive....................................................................... cpio
format of directories......................................................................... dir

graphical primitive string,

common object file symbol table format.............................................. syms
introduction to file formats............................................................... intro
utmp and wtmp entry formats............................................................ utmp
convert formatted input...................................................................... scanf
argument list print formatted output of a varargs.............................. vprintf
print formatted output........................................................................ printf
the macro package for formatting a permuted index............................ mptx
the MM macro package for formatting documents............................... mm
the OSDD adapter macro package for formatting entries in this manual.. man
text formatting macros...................................................................... ms
macros for formatting papers............................................................ me
Fortran absolute value....................................................................... abs
signal specify Fortran action on receipt of a system signal.................. signal
Fortran bitwise boolean functions..................................................... bool
return Fortran command-line argument.............................................. getarg
function Fortran complex conjugate intrinsic.................................... conjg
return Fortran environment variable................................................... getenv
square root intrinsic functions Fortran exponential, logarithm, .......... exp
functions Fortran hyperbolic intrinsic.............................................. trigh
Permuted Index

argument Fortran imaginary part of complex .. aimag
function Fortran integer part intrinsic .... aint
Fortran maximum-value functions .... max
Fortran minimum-value functions .... min
Fortran nearest integer functions .... round
terminate Fortran program .................. abort
functions Fortran remaining intrinsic .... mod
return length of Fortran string ........ len
return location of Fortran substring ... index
issue a shell command from Fortran ........ system
return Fortran time accounting .......... mclock
function Fortran transfer-of-sign intrinsic .. sign
functions Fortran trigonometric intrinsic .... trig
explicit Fortran type conversion ........ ftype
generator Fortran uniform random-number .... rand
get a message from a shared-memory queue ...... smq_get
receive a message from a socket ........ recv
send a message from a socket ........ send
get a string from a stream ........ gets
get option letter from argument vector .... getopt
read from file .......................... read
issue a shell command from Fortran ........ system
generate from name list .................. nlist
get character or word from stream ...... getc
get name from UID ......................... getpw
list of file systems processed by fsck .................. checklist
shut down part of a full-duplex connection .... shutdown
Fortran integer part intrinsic function .................. aint
function error function and complementary error .. erf
Fortran complex conjugate intrinsic function .................. conjg
double precision product intrinsic function .................. dprod
function and complementary error function error .................. erf
log gamma function ........................ gamma
Euclidean distance function .................. hypot
entries of a common object file
error-handling function .................. mather
nan881 M68881 test for Not-A-Number function .................. nan881
profile within a function ................ prof
Fortran transfer-of-sign intrinsic function .................. sign
sqrt881 - M68881 square root function .................. sqrt881
math functions and constants ........ math
Bessel functions .......................... bessel
Fortran bitwise boolean functions .................. bool
positive difference intrinsic functions .................. dim
logarithm, power, square root functions exponential, .... exp
logarithm, square root intrinsic functions Fortran exponential, .... exp
ceiling, remainder, absolute value functions floor, .................. floor
M68881 log functions .................. log881
Fortran maximum-value functions .................. max
Permuted Index

Fortran minimum-value functions .................................. min
Fortran remaindering intrinsic functions ....................... mod
Fortran nearest integer functions ................................ round
String comparison intrinsic functions ........................... strcmp
Fortran trigonometric intrinsic functions ........................ trig
M68881 trigonometric functions ................................... trig881
Fortran hyperbolic intrinsic functions .......................... trigh

Hyperbolic functions ............................................... sinh, cosh, tanh, atanh

Screen functions with optimal cursor motion ........................ curses

Log gamma function .................................................. gamma

Log termination generate an abnormal process .................... abort

Generate DES encryption ............................................. crypt

Generate file name for terminal ................................... ctermid

Pseudo-random numbers get a uniformly distributed ............ drand48

Fortran uniform random-number generator ....................... rand

Simple random-number generator .................................. rand

Queue get a message from a shared-memory ...................... smq_get

Get a set of semaphores .......................................... semget

Get a string from a stream ....................................... gets

Get and set options on sockets ................................... getsockopt

Get and set user limits ............................................ ulimit

Get character login name of the user ............................. cuserid

Get character or word from stream ................................ getc

Pathname get current working directory ............................ getwd

Get date and time ................................................... gettimeofday

Get disk description by its name ................................ getdisk

Get entries from name list ........................................ nlist

Set and get file creation mask ................................... umask

Get file status ...................................................... stat

Get file system descriptor file entry ............................ getfsent

Get file system statistics ......................................... ustat

Get group file ...................................................... getgrent

Get login name ..................................................... getlogin

Get message queue .................................................. msgget

Get name from UID .................................................. getpwuid

Get name of connected peer ....................................... getpeername

Get name of current UNIX system ................................... uname

Get option letter from argument .................................. getopt

Get password ......................................................... getpwent

Directory get pathname of current working ....................... getcwd

Get process and child process times ............................ times

Parent process IDs get process, process group, and ............ getcwd

Get real user, effective user, real user ........................ getuid

Get serial number of current system ............................. serno

Get shared memory segment ....................................... shmget

Get socket name .................................................... getsockname

Get time ............................................................ gettime

UP-11762 R2A, V1

Index-11
Permuted Index

- get/set name of current host .......... gethostname
- get/set unique identifier of current host .......... gethostid
- speed and terminal settings used by non-local
- lock resp. unlock a initialization a enter
- process allocate a deallocate a specified reset the specified set a specified
- graphical primitive string, format of graphical files
- type-box
- troff macro package to typeset view graphs and slides
- get real user, effective user, real get process, process get
- set process user, real group, and effective group IDs
- set user and send signal to a process change owner and send a signal to a process or a
- CRT screen handling and optimization package curses
- terminal virtually
- manage hash search tables
- section file
- unistd file
- read the file
- seek to the optional file
- read an indexed/named section
- read the archive
- get/set unique identifier of current host
- host
- cosh88l, tanh88l atanh88l - M6888I

- Fortran
- set process group
- USER-DEFD, issue issue
- get/set unique process group, and parent process

Index-12

UP-11762 R2A, V1
Permuted Index

user, real group, and effective group IDs get real user, effective setuid
set user and group IDs setuid
format of core image file core
Fortran imaginary part of complex argument almag
set system power up time and/or immediately power down system syspwr
special system calls for Motorola sysm68k
incremental dump format fdump
access long numeric data in a machine independent fashion sputl
terminal independent operation routines termcap
terminal independent operation routines termcap
package for formatting a permuted index the macro mptx
file compute format of core image file file
common object file read an indexed symbol table entry of object ldtbindex
common object file read an indexed symbol table entry of a ldshread
object file seek to an indexed/named section of a common ldseek
script for the init process inittab
semaphore initialize a shared-memory queue smq_init
format of an semaphore initializes a GPTF shared-memory smainl
format of an semaphore initiate a connection on a socket connect
format of an semaphore initiate pipe to/from a process popen
convert character back into input stream scanf
push character back into input stream ungetc
binary input/output package fread
standard buffered input/output package stdio
stream status inquiries ferror
return integer absolute value abs
convert between long integer and base-64 ASCII string a64l
Fortran nearest integer functions round
Fortran integer part intrinsic function ant
convert string to integer strtol
convert between 3-byte integers and long integers ltol
between 3-byte integers and long integers ltol
graphics interface plot
graphics interface subroutines plot
add a swap device for interleaved paging/swapping swapon
create an interprocess channel pipe
standard interprocess communication package stdipc
suspend execution for interval sleep
power recovery interval to single-user state pwrtime
Fortran integer part intrinsic function ant
Fortran complex conjugate intrinsic function conjg
double precision product intrinsic function dprod
Fortran transfer-of-sign intrinsic function sign
positive difference intrinsic functions dim
exponential, logarithm, square root intrinsic functions exp
Fortran remaining intrinsic functions Fortran
string comparision intrinsic functions streqcmp
Fortran trigonometric intrinsic functions trig
Permuted Index

Fortran hyperbolic intrinsic functions ............. trigh
introduction to file formats ............... intro
introduction to miscellany ............... intro
libraries introduction to subroutines and .... intro
definitions, and error numbers
synchronous i/o multiplexing .................. select
issue a shell command from Fortran system
issue a shell command .............. system
USER-DEFD, issue
USER-DEFD, description of output
issue identification file .............. isort
issue issue identification file ........ isort
return language ............................. troff
length of Fortran string .................... len
truncate a file to a specified
get option letter from argument vector .... getopt
number of command line arguments ........... large
establish an out-going terminal
get and set user line arguments ............. large
line connection ...................... dial
dialup line file ........................... dialups
object file line number entries in a common
object file function manipulate ........................
common object file seek to
common assembler and
object file line number entries of a common
common object file seek to
link editor output ........................... a.out
link files across file systems .......... slink
link to a file ................................ link
get entries from name list ................ nlist
fsck list of file systems processed by .... checklist
output of a varargs argument
return location of Fortran substring ........ Index
listen for connections on a socket ... listen
last locations in program ............ end
lock process, text, or data in mem .......... plock
shared-memory semaphore
shared-memory semaphore
record locking on files .................... lockf
record locking on files .................... lockf
M68881 log functions .................... log881
log gamma function ........................ gamma
logalert summary message file .......... alertmesg
threshold logalert threshold file .......... threshold
functions exponential, log function
functions Fortran exponential, log function
get getlogin
get get character
get get character
return return
setting up an environment at
convert between convert between 3-byte integers and
long int and base-64 ASCII string ................ a64l
long integers ................................ 13tol
Permuted Index

independent fashion. access long numeric data in a machine ..... sputl

M68881 hyperbolic functions ..... trigh881
M68881 log functions ..... log881
M68881 pow881

M68881 square root function ..... sqrt881
M68881 test for Not-A-Number fnctn . nan881
M68881 trigonometric functions ..... trig881

access long numeric data in a machine independent fashion. . sputl

machine-dependent values values

permuted index the macro package for formatting a .... mpty

documents the MM macro package for formatting .... mm
documents the OSDD adapter macro package for formatting .... mosd
and slides troff macro pkg to typeset view graphs . mv
manual macros for formatting entries in this man

macros for formatting papers me

text formatting macros ..... ms
main memory allocator malloc
fast main memory allocator malloc
make a directory file mkdir
ordinary file make a directory, or a special or ... mknod
ordinary file make a directory, or a special or ... mknod
ordinary file make a directory, or a special or ... mknod
make a unique file name mktmp
manage binary search trees tsearch
manage hash search tables hsearch

common object file function manipulate line number entries of a ldldread
numbers manipulate parts of floating-point frexp
macros for formatting entries in this manual man
map of ASCII character set ascii
set and get file creation mask umask
master device information table master
regular expression compile and match routines regexp
math functions and constants math

Fortran maximum-value functions max
fast main memory allocator malloc
main memory allocator malloc
shared memory control operations shmat
memory operations memory
shared memory operations shmop
lock process, text, or data in memory plock
get shared memory segment shmget
message control operations msgctl
logalert summary message file alertmsg
get a msg from a shared-memory queue . smq_get
receive a message from a socket ..... recv
send a message from a socket send
put a message into a shared-memory queue smq_put
get message queue msgget
<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>system error</td>
<td>message send and receive operations msgop</td>
</tr>
<tr>
<td>Fortran</td>
<td>messages .............................................</td>
</tr>
<tr>
<td>documents</td>
<td>minimum-value functions ............................</td>
</tr>
<tr>
<td>change</td>
<td>MM macro package for formatting ...................</td>
</tr>
<tr>
<td>enter GPTF timer disable</td>
<td>mode of file ........................................</td>
</tr>
<tr>
<td>enter GPTF timer enable</td>
<td>mode ..................................................</td>
</tr>
<tr>
<td>set file access and status</td>
<td>modification times ...................................</td>
</tr>
<tr>
<td>screen functions with optimal</td>
<td>motion ................................................</td>
</tr>
<tr>
<td>cursor</td>
<td>Motorola implementation .............................</td>
</tr>
<tr>
<td>file systems</td>
<td>mount a file system ..................................</td>
</tr>
<tr>
<td>synchronous I/O</td>
<td>multiplexing .........................................</td>
</tr>
<tr>
<td>function</td>
<td>nan881 M68881 test for Not-A-Number nan881</td>
</tr>
<tr>
<td>Fortran</td>
<td>nearest integer functions ..........................</td>
</tr>
<tr>
<td>character definitions for eqn</td>
<td>local-to-goto .......................................</td>
</tr>
<tr>
<td>and eqn</td>
<td>non-local goto ......................................</td>
</tr>
<tr>
<td>character definitions for</td>
<td>select</td>
</tr>
<tr>
<td>eqn</td>
<td>character definitions for eqn and eqn</td>
</tr>
<tr>
<td>line number entries of a</td>
<td>numeric data in a machine-independent sputl</td>
</tr>
<tr>
<td>common</td>
<td>object file access routines ........................</td>
</tr>
<tr>
<td>open a common</td>
<td>object file for reading ............................</td>
</tr>
<tr>
<td>close a common</td>
<td>object file function manipulate ...................</td>
</tr>
<tr>
<td>read the file header of a</td>
<td>object file ..........................................</td>
</tr>
<tr>
<td>common</td>
<td>object file seek to line number ...................</td>
</tr>
<tr>
<td>entries of a section of a</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>relocation information for a common ................</td>
</tr>
<tr>
<td>an indexed/named section of a</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>relocation information for a common ................</td>
</tr>
<tr>
<td>index of symbol table entry</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>symbol table entry of</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>seek to the symbol table of a</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>line number entries in a</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>relocation information for a</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>retrieve symbol name for</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>common</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>file header for common</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>duplicate an</td>
<td>object file seek to relocation .....................</td>
</tr>
<tr>
<td>Index-16</td>
<td>UP-11762 R2A, V1</td>
</tr>
</tbody>
</table>
Permuted Index

- open for reading or writing ......... open
- opendir, readdir, telldir, seekdir, .directory
- operation system ..................... sysident
- operation routines ................... termcap
- operation routines ................... termcap
- block operations ...................... block
- memory operations ..................... memory
- message control operations .......... msgctl
- message send and receive operations msgop
- semaphore control operations ........ semctl
- semaphore operations ................. semop
- shared memory control operations ... shmctl
- shared memory operations ............. shmp
- screen functions with optimal cursor motion .......... cursesr2
- CRT screen handling and optimization package .......... curses
- object file seek to the optional file header of a common.Idiosseek
- get options on sockets ................ getsockopt
- make a directory, or a special or ordinary file mjknod
- make a directory, or a special or ordinary file mknod
- make a directory, or a special or ordinary file mknod
- formatting documents the OSDD adapter macro package for ... mptx
- establish an out-going terminal line connection .. dial
- common assembler and link editor output a.out
- description of output language troff
- print formatted output of a varargs argument list vprintf
- print formatted output printf
- change owner and group of a file chown
- CRT screen handling and optimization package .......... curses
- index the macro package for formatting a permuted ... mptx
- the MM macro package for formatting documents ... mm
- the OSDD adapter macro package for formatting documents .. mptx
- standard buffered input/output package .......... stdio
- standard interprocess communication package .... stdpipc
- slides troff macro package to typeset view graphs and mv
- add a swap device for interleaved paging/swapping swapon
- create a pair of connected sockets socketpair
- macros for formatting parent process IDs getpid
- get process, process group, and get password file d_passwd
- dialup password file entry putpwent
- write password file password
- read a password getpass
- get password getpwent
- get current working directory pathname getwd
- get pathname of current working dir .. getcwd
- get name of connected peer getpeername
- the macro package for formatting a permuted index mptx

UP-11762 R2A, V1
Permutated Index

initiate
reposition a file
move read/write file functions
exponential, logarithm, double
files graphical
argument list
change
set a process
enable or disable
set the
set the process alarm clock for a
get
terminate
create a new
get process,
set
send signal to a
process, process group, and parent
script for the init
change priority of a
send a signal to a
initiate pipe to/from a
set a
process IDs get
generate an abnormal
lock
get process and child
wait for child
wait for
GPTF timer structure for the calling
suspend
list of file systems
a signal to a process or a group of
floating point
double precision
prepare execution
execution time
reading and writing
generate uniformly distributed

per-process accounting file format .. acct
pipe to/from a process .......... popen
pointer in a stream ................ fseeks
pointer ............................ lseek
positive difference intrinsic ...... dim
power, square root functions ...... exp
precision product intrinsic function dprod
prepare execution profile .......... monitor
primitive string, format of graphical gps
primitive system data types ......... types
print formatted output of a varargs . vprintf
print formatted output .......... printf
priority of a process .............. nice
priority to a fixed value .......... tpfix
process accounting ............... acct
process alarm clock for a process .. alarm
process ......................... alarm
process and child process times .... times
process ......................... exit
process ......................... fork
process group, and parent proc IDs getpid
process group ID ............... setpgp
process group .................... killpg
process IDs get .................. getpid
process ......................... initab
process ......................... nice
process or a group of processes ... kill
process ......................... popen
process priority to a fixed value .... tpfix
process, process group, and parent getpids
process termination .......... abort
process, text, or data in memory .. plock
process times .......... times
process to stop or terminate .. wait
process to terminate .......... wait3
process allocate a .......... tptmalloc
process trace ................. ptrace
process until signal .......... pause
processed by fsck ............ checklist
processes send ................ kill
processor access .......... ffp
product intrinsic function ...... dprod
profile ......................... monitor
profile ....................... profil
profile within a function ......... prof
provide exclusive file regions for ... lockf
test pseudo-random numbers .... drand48
push char back into input stream .. ungetc
query audit trail security status ... ckaudit
get message from a shared-memory queue ................................. msgget
initialize a shared-memory queue ........................................ smq_init
put a message into a common object file ............................... smq_put

Fortran uniform random-number generator ............................... rand
read a password .................................................................... getpass
read an indexed symbol tbl entry of ldttbread ............................ getpass
read an indexed/named section hdr ldshread .............................. smq_get
read from file ...................................................................... read
read the archive header of a member ldahread .............................. read
deadir, telldir, seekdir, directory ......................................... opendir, readdir, telldir, seekdir
provide exclusive file regions for a common object file ............. lockf
open a common object file for reading and writing ..................... lseek
open for reading or writing .................................................. open
open a file .......................................................................... fopen

specify what to do upon receipt of a message from a socket .......... recv
specify Fortran action on receipt of a system signal .................... signal
receive a message from a socket ............................................. recv
receive operations .................................................................. msgop
record locking on files ......................................................... lockf
record locking on files ......................................................... lockf
read operations ..................................................................... lseek
read an indexed symbol tbl entry of ldttbread ............................ qsort
read an indexed/named section hdr ldshread .............................. qsort
read from file ...................................................................... read
read the archive header of a member ldahread .............................. read
deadir, telldir, seekdir, directory ......................................... opendir, readdir, telldir, seekdir
provide exclusive file regions for a common object file ............. lockf
open a common object file for reading and writing ..................... lseek
open for reading or writing .................................................. open
open a file .......................................................................... fopen

threshold values error records for devices exceeding ................ alert
state power recovery interval to single-user .............................. pwrtime
power recovery notification ................................................... pwrnote
provide exclusive file regions for reading and writing ................ lockf
6881 control register access ................................................ access881
routines regular expr compile and match ................................. regexp
compile and execute regular expression .................................... regcmp
system date and release number of the operating system ......... sysident
common object file relocation information for a common object file ldshread
common object file seek to relocation entries of a section of a ldhrread
floor, ceiling, Fortran remainder, absolute value functions ........ floor
remaining intrinsic functions ................................................... mod
remove a directory file ......................................................... rmdir
remove directory entry ......................................................... unlink
report CPU time used ........................................................... clock
reposition a file pointer in a stream ......................................... fseek
reset the specified GPTF timer ............................................... tpttimreset
resp. unlock a GPTF shared-memory Individually reset ............... lockf
return Fortran command-line arg .......................................... getarg
return Fortran environment variable ....................................... getenv
return Fortran time accounting ............................................ mcllck
return integer absolute value ................................................ abs
Permuted Index

return length of Fortran string  ..... len
return location of Fortran substring  index
return login name of user  ........... logname
return value for environment name  getenv

data
create a new file or
change

sqrt881 - M68881 square

exponential, logarithm, power, square

common object file access

regular expression compile and match

terminal independent operation

terminal independent operation

format of
motion
package CRT

linear

binary

manage hash

manage binary

file

read an indexed/named

seek to line number entries of a

seek to relocation entries of a

modify/query audit trail

query audit trail

a common object file

section of a common object file

section of a common object file

common object file

object file

opendir, readdir, telldir, get shared memory

change data

resp. unlock a GPTF shared-memory

initializes a GPTF shared-memory

get a set of

of processes

message

send signal to a process or a group

send signal to a process group

sernum - get

system

get serial number of current system  sernum

sernum - get serial num of current  sernum


Index-20  UP-11762 R2A, V1
a system signal
signal
getty
sqrt881 - M68881
exponential, logarithm, power,
Fortran exponential, logarithm,
package
package
data returned by
filesystems
get file system
modify/query audit trail security
query audit trail security
stream
get file
wait for child process to
varchar, strncat, strup,
strdup, strcat, strncat,
strcat, strncat, strup,
strdup, strncat, strncat,
strdup, strncat, strncat,
close or flush a
stream
one
reposition a file pointer in a
get character or word from
get a string from a
put character or word on a
put a string on a
assign buffering to a
stream
open a
push character back into input
long integer and base-64 ASCII
functions
convert date and time to
convert floating-point number to
graphical primitive
get a
return length of Fortran
put a
convert
convert
strncat, strdmp, strcat,
strcat, strncat, strmp,
strncat, strmp, strncat,
strmp, strncat, strmp,
strncat, strncat, strmp,
strncat, strmp, strncat,
structure for the calling process
deallocates a specified GPTF timer
introduction to
graphics interface
return location of Fortran

specify Fortran action on receipt of . signal
specify what to do upon receipt of a signal
speed and terminal settings used by gettydefs
sqrt881 - M68881 square root fnctn .. sqrt881
square root functions ............. exp
square root functions ............. exp
standard buffered input/output ...... stdio
std interprocess communication ...... stdipc
stat system call ............... stat
static information about the ....... fsstab
statistics .................. ustat
status .......................... audit
c not .......................... okaudit
status inquiries .......................... ferror
status .......................... stat
stop or terminate .......................... wait
strcat, strncat, strmp, strncat,
strmp, strncat, strmp, strncat,
strmp, strncat, strmp, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
strdup, strcat, strncat,
logalert summary message file .......... alertmesg
update super-block ........................... sync
suspend execution for interval .......... sleep
synchronize process until signal ........ pause
swap bytes ............................... swap
swap device for interleaved .............. swapon
symbol name for common object file . ldgetname
symbol table entry retrieve ............ ldgetname
symbol table entry of a common obj . ldbitbread
symbol table entry of object file ........ ldbitindex
symbol table format ........................... syms
symbol table of a common object file . ldbsseek
symbolic constants ......................... unistd
synchronous i/o multiplexing ............ select
synchronously write on a file .......... swrite
name for common object file symbol
read an indexed symbol
compute index of symbol
common object file symbol
master device information
mounted file system
mounted file system
seek to the symbol
mounted directory
manage hash search
tables on a terminal ..................... tabs
functions sinh881, cosh881, opendir, readdir,
create a name for a
format of compiled
term file ........................... term
generate file name for
routines
terminal capability data base ............ btermcap
terminal capability data base ............ termcap
terminal capability data base ............ terminfo
terminal ........................... ctermid
terminal independent operation .......... termcap
terminal independent operation .......... termcap
terminal line connection .................. dial
terminal settings used by getty .......... gettydefs
terminal ........................... term
find name of a
set tabs on a
"hangup" the current control
terminfo
set tabs on a
"hangup" the current control
terminfo
wait for child process to stop or
wait for process to
nan881 M68881
test for Not-A-Number function ........ nan881
UP-11762 R2A, V1
Index-23
<table>
<thead>
<tr>
<th>Format specification in text files</th>
<th>fspec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lock process, text, or data in memory</td>
<td>plock</td>
</tr>
<tr>
<td>Threshold - logical error records for devices exceeding</td>
<td>threshold</td>
</tr>
<tr>
<td>Return Fortran</td>
<td>mclock</td>
</tr>
<tr>
<td>System set system power up</td>
<td>gettimeofday</td>
</tr>
<tr>
<td>Set date and time</td>
<td>profil</td>
</tr>
<tr>
<td>Execute</td>
<td>time</td>
</tr>
<tr>
<td>Setting up an environment at login</td>
<td>profile</td>
</tr>
<tr>
<td>Set/get</td>
<td>tslice</td>
</tr>
<tr>
<td>Get</td>
<td>stime</td>
</tr>
<tr>
<td>Convert date and time</td>
<td>ctime</td>
</tr>
<tr>
<td>Report CPU</td>
<td>clock</td>
</tr>
<tr>
<td>Enter GPTF timer disable mode</td>
<td>tptimendisable</td>
</tr>
<tr>
<td>Enter GPTF timer enable mode</td>
<td>tptimenable</td>
</tr>
<tr>
<td>Process allocate a GPTF timer structure for the calling process</td>
<td>tptimalloc</td>
</tr>
<tr>
<td>Deallocates a specified GPTF timer structure</td>
<td>tptimdealloc</td>
</tr>
<tr>
<td>Reset the specified GPTF timer</td>
<td>tptimreset</td>
</tr>
<tr>
<td>Set a specified GPTF timer</td>
<td>tptimset</td>
</tr>
<tr>
<td>Get process and child process</td>
<td>times</td>
</tr>
<tr>
<td>Set file access and modification</td>
<td>utime</td>
</tr>
<tr>
<td>Initiate pipe to/from a process</td>
<td>popen</td>
</tr>
<tr>
<td>Process</td>
<td>ptrace</td>
</tr>
<tr>
<td>Enable/disable audit</td>
<td>audon</td>
</tr>
<tr>
<td>Modify/query audit</td>
<td>audit</td>
</tr>
<tr>
<td>Query audit</td>
<td>ckaudit</td>
</tr>
<tr>
<td>Fortran transfer-of-sign intrinsic function</td>
<td>sign</td>
</tr>
<tr>
<td>Translate characters</td>
<td>conv</td>
</tr>
<tr>
<td>Walk a file</td>
<td>ftw</td>
</tr>
<tr>
<td>Manage binary search</td>
<td>tsearch</td>
</tr>
<tr>
<td>Manage GPTF trigonometric functions</td>
<td>trig</td>
</tr>
<tr>
<td>Manage GPTF trigonometric functions</td>
<td>trig881</td>
</tr>
<tr>
<td>Manage Fortran trigonometric intrinsic functions</td>
<td>trig</td>
</tr>
<tr>
<td>Manage graphs and slides</td>
<td>mv</td>
</tr>
<tr>
<td>Manage for the extended graphics</td>
<td>greek</td>
</tr>
<tr>
<td>Explicit Fortran</td>
<td>type conversion</td>
</tr>
<tr>
<td>Manage for the extended TTY-37 graphics</td>
<td>ftype</td>
</tr>
<tr>
<td>Type-box</td>
<td>greek</td>
</tr>
<tr>
<td>Primitive system data types</td>
<td>types</td>
</tr>
<tr>
<td>Truncate macro package to typeset view graphs and slides</td>
<td>mv</td>
</tr>
<tr>
<td>Etc/TIMEZONE set the env variable</td>
<td>timezone</td>
</tr>
<tr>
<td>Get name from UID</td>
<td>getpw</td>
</tr>
<tr>
<td>Fortran uniform random-number generator</td>
<td>rand48</td>
</tr>
<tr>
<td>Make a unique file name</td>
<td>mktemp</td>
</tr>
<tr>
<td>Get/set unique identifier of current host</td>
<td>gethostid</td>
</tr>
</tbody>
</table>
NAME
   m4 - macro processor

SYNOPSIS
   m4 [ options ] [ files ]

DESCRIPTION
   M4 is a macro processor intended to be a pre-processor for Ratfor, C, and other languages. Each of the argument files is processed in order; if there are no files, or if a file name is -, the standard input is read. The processed text is written on the standard output.

OPTIONS
   The options and their effects are as follows:
   -Dname [=val]      Defines name to val or to null if val is not specified.
   -Uname             undefines name.

   The following options must appear before the file names and before any -D or -U options.
   -e                 Operate interactively. Interrupts are ignored and the output is unbuffered.
   -s                 Enable line sync output for the C preprocessor (#line . . . )
   -Bint              Change the size of the push-back and argument collection buffers from the default of 4,096.
   -Hint              Change the size of the symbol table hash array from the default of 199. The size should be prime.
   -Sint              Change the size of the call stack from the default of 100 slots. Macros take three slots, and non-macro arguments take one.
   -Tint              Change the size of the token buffer from the default of 512 bytes.

SYNTAX
   Macro calls have the form:
   name(arg1,arg2, . . . , argn)

   The ( must immediately follow the name of the macro. If the name of a defined macro is not followed by a (, it is assumed to be a call of that macro with no arguments. Potential macro names consist of alphabetic letters, digits, and underscore; the first character may not be a digit.

   Leading unquoted blanks, tabs, and new-lines are ignored while collecting arguments. Left and right single quotes are used to quote strings. The value of a quoted string is the string stripped of the quotes.
When a macro name is recognized, its arguments are collected by searching for a matching right parenthesis. If fewer arguments are supplied than are in the macro definition, the trailing arguments are assumed to be null. Macro evaluation proceeds normally during the collection of the arguments, and any commas or right parentheses which happen to turn up within the value of a nested call function as if the expanded macro had been placed into the text in the first place. After argument collection, the value of the macro is pushed back onto the input stream and rescanned.

*M4* makes available the following built-in macros. They may be redefined, but once this is done the original meaning is lost. Their values are null unless otherwise stated.

define (mname, arg1, arg2, ...)  
defines arg1 as the value of the macro mname. When mname is subsequently used, *M4* replaces each occurrence of $n$ in the replacement text (where $n$ is a digit) with the $n$-th argument. $0$ is the name of the macro; missing arguments are replaced by the null string; $# is the number of arguments; $* is a list of all the arguments separated by commas; $@ is like $*, but each argument is quoted with the current quotes (see changequote).

undefine (mname)  
removes the definition of the macro named in its argument.

defn (m1, m2, m3, ...)  
returns the quoted definition of its argument(s). Defn is useful for renaming macros, especially built-ins.

pushdef (mname, arg1, arg2, ...)  
like define, but saves any previous definition.

popdef (m1, m2, ...)  
removes current definition of its argument(s), exposing the previous one, if any.

ifdef (mname, val1, val2)  
if mname is defined, returns val1; otherwise returns val2. If val2 is not specified, ifdef returns null. The word unix is predefined on the UNIX system versions of *M4*.

shift (arg1, arg2, ...)  
returns all but arg1. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that is subsequently performed.

changequote (lq, rq)  
changes the quote symbols to lq and rq. The symbols may be up to five characters long. Changequote without arguments restores the original values (left and right single quotes).
changecom \((lm, rm)\)
change left and right comment markers from the default 
# and new-line. If \(lm\) and \(rm\) are not specified, the com-
ment mechanism is effectively disabled. If only \(lm\) is 
specified, the left marker becomes \(lm\) and the right 
marker becomes new-line. If both \(lm\) and \(rm\) are speci-
fied, both markers are changed. Comment markers may 
be up to five characters long.

divert \((\text{digit-string})\)
\(m4\) maintains 10 output streams, numbered 0-9. The 
final output is the concatenation of the streams in 
numerical order; initially stream 0 is the current stream. 
The divert macro changes the current output stream to 
digit-string argument. Output diverted to a stream 
other than 0 through 9 is discarded.

undivert \((\text{dstring1, dstring2, ...})\)
causes immediate output of text from the output streams 
named as arguments, or from all output streams if no 
arguments are specified. Text may be undiverted into 
another output stream. Undiverting discards the text in 
the output stream(s) specified by the arguments.

divnum
returns the value of the current output stream.

dnl
reads and discards characters up to and including the 
next new-line.

ifelse \((\text{arg1, arg2, arg3 [, arg4 ... ]})\)
returns \(\text{arg3}\) if \(\text{arg1}\) is the same string as \(\text{arg2}\). \(\text{arg1}\), 
\(\text{arg2}\), \(\text{arg3}\) must be given. If \(\text{arg1}\) is not equal to \(\text{arg2}\), 
and if five or more \(\text{args}\) are specified, ifelse repeats, 
using \(\text{args}\) 4, 5, 6, and 7; otherwise, ifelse returns 
\(\text{arg4}\), or, if \(\text{arg4}\) is not present, null.

incr \((\text{arg})\)
returns the value of \(\text{arg}\) incremented by 1. The value of 
\(\text{arg}\) is calculated by interpreting an initial digit-string 
as a decimal number.

decr \((\text{arg})\)
returns the value of \(\text{arg}\) decremented by 1.

eval \((\text{expr, radix, digits})\)
evaluates \(\text{expr}\) as an arithmetic expression, using 32-bit 
arithmetic. Operators include \(+, -, *, /, ^\) (exponenti-
tation), bitwise \&, |, ^, \(\oplus\), and \(\ominus\); relationals;
parentheses. Octal and hex numbers may be specified 
as in C. Radix specifies the radix for the result; the 
default is 10. Digits may be used to specify the minimum 
number of digits in the result.

len \((\text{string})\)
returns the number of characters in \(\text{string}\).
index (string, pattern)
returns the position in string where pattern begins (zero origin), or -1 if pattern does not occur in string.

substr (string, start, length)
returns a substring of string. Start is a zero origin number selecting the first character; length indicates the length of the substring. If length is not specified, length is assumed to be large enough to extend to the end of the first string.

translit (chars, set1, set2)
transliterates the characters in chars from the set set1 to the set set2. No abbreviations are permitted.

include (fname)
returns the contents of the file named fname.

sinclude (fname)
returns the contents of the file named fname, but does not print a message if the file is inaccessible.

syscmd (unixcmd)
executes the UNIX system command unixcmd. No value is returned.

sysval
is the return code from the last call to syscmd.

maketemp (string)
fills in a string of XXXXX in string with the current process ID.

m4exit (xcode)
exits immediately from m4. xcode, if given, is the exit code; if not given, xcode is assumed to be 0.

m4wrap (arg)
arg will be pushed back at final EOF; example:
m4wrap("cleanup ()")

errprint (arg)
prints arg on the diagnostic output file.

dumpdef (item1, item2, ...)
prints current names and definitions, for the named items, or for all items if no arguments are given.

traceon (m1, m2, ...)
with no arguments, turns on tracing for the all macros (including built-ins). Otherwise, turns on tracing for named macros, m1, m2, ..., etc.

traceoff (m1, m2, ...)
turns off trace globally and for any macros specified. Macros specifically traced by traceon can be untraced only by specific calls to traceoff.

SEE ALSO
cc(1), cpp(1).
The M4 Macro Processor in the Support Tools Guide.
NAME
pdp11, s5k20, s5k30, s5k40, s5k50, s5k60, s5k80, s5k90, s7k30, s7k40, tahoe, u3b, u3b5, vax - provide truth value about your processor type

SYNOPSIS
pdp11
s5k20
s5k30
s5k40
s5k50
s5k60
s5k80
s5k90
s7k30
s7k40
tahoe (7000 Series Systems only)
u3b
u3b5
vax

DESCRIPTION
The following commands return a true value (exit code of 0) if you are on a processor that the command name indicates.

pdp11 True if you are on a PDP-11/45 or PDP-11/70.
s5k20 True if you are on a Sperry 5000/20.
s5k30 True if you are on a Sperry 5000/30.
s5k40 True if you are on a Sperry 5000/40.
s5k50 True if you are on a Sperry 5000/50.
s5k60 True if you are on a Sperry 5000/60.
s5k80 True if you are on a Sperry 5000/80.
s5k90 True if you are on a Sperry 5000/90.
s7k30 True if you are on a Sperry 7000/30.
s7k40 True if you are on a Sperry 7000/40.
tahoe True if you are on a Sperry 7000/40 (tahoe is found only on the 7000/30 and 7000/40 and will be removed in a later release).
u3b True if you are on a 3B 20 computer.
u3b5 True if you are on a 3B 5 computer.
vax True if you are on a VAX-11/750 or VAX-11/780.
The commands that do not apply return a false (non-zero) value. These commands are often used within `make(1)` makefiles and shell procedures to increase portability.

**SEE ALSO**

`make(1), sh(1), test(1), true(1).`
NAME
mail, rmail, smail - send mail to users or read mail

SYNOPSIS
mail [ -epqr ] [ -f file ]
smail [ -epqr ] [ -f file ] (5000/20/30/40/50 only)
mail [ -t ] persons
smail [ -t ] persons (5000/20/30/40/50 only)
rmail [ -t ] persons

DESCRIPTION
Mail, without arguments, prints mail for a user, message-by-message, in last-in, first-out order. For each message, the user is prompted with a ?, and a line is read from the standard input to determine the disposition of the message:

<newline> Go on to next message.
+ Same as <newline>.
d Delete message and go on to next message.
p Print message again.
- Go back to previous message.
s [ files ] Save message in the named files (mbox is default).
w [ files ] Save message, without its header, in the named files (mbox is default).
m [ persons ] Mail the message to the named persons (yourself is default).
q Put undeleted mail back in the mailfile and stop.
EOT (control-d) Same as q.
x Put all mail back in the mailfile unchanged and stop.
!command Escape to the shell to do command. (Not valid for smail)
* Print a command summary.

Smail is linked to mail and works like mail except that it does not allow the !command. Smail is primarily used as a security feature to prevent an unauthorized user access to UNIX utilities, but allows them to read their mail. Smail is available only on the 5000/20/30/40/50.

Rmail permits only the sending of mail; uucp(1C) uses rmail as a security precaution.

OPTIONS
The options alter the printing of the mail:
-e Do not print mail. An exit value of 0 is returned if the user has mail; otherwise, an exit value of 1 is returned.
-p Print all mail without prompting for disposition.
-q Terminate mail after interrupts. Normally an interrupt only causes the termination of the message being printed.
-r Print messages in first-in, first-out order.
-f file Use file (e.g., mbox) instead of the default mailfile.
-t Place the names of all persons to whom the mail was sent on the postmark of the mail for each person. This allows all who receive mail to know who else received that letter.

Addressing Mail
When persons are named, mail takes the standard input up to an end-of-file (typically control-d) or up to a line consisting of just a period and adds it to the mailfile for each person. The message is preceded by the name of the sender and a postmark. Lines that look like postmarks in the message, (i.e., From ...) are preceded with a >. A person is usually a user name recognized by login(1). If a person being sent mail is not recognized, or if mail is interrupted during input, the file dead.letter is saved to allow editing and resending. Note that this is regarded as a temporary file in that it is recreated each time it is needed erasing the previous contents of dead.letter.

Remote Systems
To denote a recipient on a remote system, prefix person by the system name and exclamation mark (see uucp(1C)). Everything after the first exclamation mark in persons is interpreted by the remote system. In particular, if persons contains additional exclamation marks, it can denote a sequence of machines through which the message is to be sent on the way to its ultimate destination. For example, specifying a!b!cde as the name of the recipient causes the message to be sent to user b!cde on system a. System a interprets that destination as a request to send the message to user cde on system b. This might be useful, for instance, if the sending system can access system a but not system b, and system a has access to system b. Mail does not use uucp if the remote system is the local system name (i.e., localsystem!user).

Privacy
The mailfile may be manipulated in two ways to alter the function of mail. The other permissions of the file may be read-write, read-only, or neither read nor write to allow different levels of privacy. If changed to other than the default, the file is preserved even when empty to perpetuate the desired permissions.

Forwarding Mail
The file may also contain the first line:

    Forward to person

which causes all mail sent to the owner of the mailfile to be forwarded to person. This is especially useful to forward all of the mail for one person to one machine in a multiple machine environment. In order for forwarding to work properly, the mailfile should have "mail" as group ID, and the group permission should be read-write.

Rmail permits only the sending of mail; uucp(1C) uses rmail as a security precaution.

When a user logs in, the presence of mail, if any, is indicated. Also, notification is made if new mail arrives while using mail.
FILES
/etc/passwd to identify sender and locate persons
/usr/mail/user incoming mail for user; i.e., the mailfile
$HOME/mbox saved mail
$MAIL variable containing path name of mailfile
/tmp/mail temporary file
/usr/mail/*.<lock lock for mail directory
dead.letter unmailable text

SEE ALSO
login(1), mailx(1), uucp(1C), write(1).

RESTRICTIONS
Race conditions sometimes result in a failure to remove a lock file.
After an interrupt, the next message may not be printed; printing
may be forced by typing a p.
NAME
mailx - interactive message processing system

SYNOPSIS
mailx [options] [name...]

DESCRIPTION
The command mailx provides a comfortable, flexible environment for sending and receiving messages electronically. When reading mail, mailx provides commands to facilitate saving, deleting, and responding to messages. When sending mail, mailx allows editing, reviewing and other modification of the message as it is entered.

Incoming mail is stored in a standard file for each user, called the system mailbox for that user. When mailx is called to read messages, the mailbox is the default place to find them. As messages are read, they are marked to be moved to a secondary file for storage, unless specific action is taken, so that the messages need not be seen again. This secondary file is called the mbox and is normally located in the user's HOME directory; see "MBOX" (ENVIRONMENT VARIABLES) for a description of this file. Messages remain in this file until forcibly removed.

OPTIONS
On the command line, options start with a dash (-) and any other arguments are taken to be destinations (recipients). If no recipients are specified, mailx attempts to read messages from the mailbox. Command line options are:

- e Test for presence of mail. Mailx prints nothing and exits with a successful return code if there is mail to read.
- f [filename] Read messages from filename instead of mailbox. If no filename is specified, the mbox is used.
- F Record the message in a file named after the first recipient. Overrides the "record" variable, if set (see ENVIRONMENT VARIABLES).
- h number The number of network "hops" made so far. This is provided for network software to avoid infinite delivery loops.
- H Print header summary only.
- i Ignore interrupts. See also "ignore" (ENVIRONMENT VARIABLES).
- n Do not initialize from the system default Mailx.rc file.
- N Do not print initial header summary.
- r address Pass address to network delivery software. All tilde commands are disabled.
- s subject Set the Subject header field to subject.
- u user Read user's mailbox. This is only effective if user's mailbox is not read protected.
MAILX(1)

-U Convert uucp(1) style addresses to internet standards. Overrides the "conv" environment variable.

SENDING AND RECEIVING
When reading mail, mailx is in command mode. A header summary of the first several messages is displayed, followed by a prompt indicating mailx can accept regular commands (see COMMANDS below). When sending mail, mailx is in input mode. If no subject is specified on the command line, a prompt for the subject is printed. As the message is typed, mailx reads the message and stores it in a temporary file. Commands may be entered by beginning a line with the tilde (\~) escape character followed by a single command letter and optional arguments. See TILDE ESCAPES for a summary of these commands.

The user can access a secondary file by using the -f option of the mailx command. Messages in the secondary file can then be read or otherwise processed using the same COMMANDS as in the primary mailbox. This gives rise within these pages to the notion of a current mailbox.

At any time, the behavior of mailx is governed by a set of environment variables. These are flags and valued parameters which are set and cleared via the set and unset commands. See ENVIRONMENT VARIABLES below for a summary of these parameters.

ADDRESSING MAIL
Recipients listed on the command line may be of three types: login names, shell commands, or alias groups. Login names may be any network address, including mixed network addressing. If the recipient name begins with a pipe symbol (\%), the rest of the name is taken to be a shell command to pipe the message through. This provides an automatic interface with any program that reads the standard input, such as lp(1) for recording outgoing mail on paper. Alias groups are set by the alias command (see COMMANDS below) and are lists of recipients of any type.

COMMAND SYNTAX
Regular commands are of the form

[ command ] [ msglist ] [ arguments ]

If no command is specified in command mode, print is assumed. In input mode, commands are recognized by the escape character, and lines not treated as commands are taken as input for the message.

Each message is assigned a sequential number, and there is at any time the notion of a 'current' message, marked by a '>' in the header summary. Many commands take an optional list of messages (msglist) to operate on, which defaults to the current message. A msglist is a list of message specifications separated by spaces,
which may include:

- **n** Message number n.
- **:** The current message.
- **$** The last message.
- *** All messages.
- **n-m** An inclusive range of message numbers.
- **user** All messages from user.
- **/string** All messages with string in the subject line (case ignored).
- **:c** All messages of type c, where c is one of:
  - d deleted messages
  - n new messages
  - o old messages
  - r read messages
  - u unread messages

Note that the context of the command determines whether this type of message specification makes sense.

Other arguments are usually arbitrary strings whose usage depends on the command involved. File names, where expected, are expanded via the normal shell conventions (see *sh*(1)). Special characters are recognized by certain commands and are documented with the commands below.

### STARTUP COMMANDS

At start-up time, *mailx* reads commands from a system-wide file (`/usr/lib/mailx/mailx.rc`) to initialize certain parameters, then from a private start-up file (`$HOME/.mailrc`) for personalized variables. Most regular commands are valid inside start-up files, the most common use being to set up initial display options and alias lists. The following commands are not valid in the start-up file: !, Copy, edit, followup, Followup, hold, mail, preserve, reply, Reply, shell, and visual. Any errors in the start-up file cause the remaining lines in the file to be ignored.

### COMMANDS

The following is a complete list of *mailx* commands:

- **!shell-command**
  Escape to the shell. See "SHELL" (ENVIRONMENT VARIABLES).

- **# comment**
  Null command (comment). This may be useful in *.mailrc* files.

- **=**
  Print the current message number.
MAILX(1)

?  Print a summary of commands.

alias alias name ...
group alias name ...  
Declare an alias for the given names. The names are substituted when alias is used as a recipient. Useful in the .mailrc file.

alternates name ...  
Declare a list of alternate names for your login. When responding to a message, these names are removed from the list of recipients for the response. With no arguments, alternates prints the current list of alternate names. See also "allnet" (ENVIRONMENT VARIABLES).

cd [directory]
chdir [directory]  
Change directory. If directory is not specified, $HOME is used.

copy [filename]
copy [msglist] filename  
Copy messages to the file without marking the messages as saved. Otherwise equivalent to the save command.

Copy [msglist]  
Save the specified messages in a file whose name is derived from the author of the message to be saved, without marking the messages as saved. Otherwise equivalent to the Save command.

delete [msglist]  
Delete messages from the mailbox. If "autoprint" is set, the next message after the last one deleted is printed (see ENVIRONMENT VARIABLES).

discard [header-field ...]
ignore [header-field ...]  
Suppress printing of the specified header fields when displaying messages on the screen. Examples of header fields to ignore are "status" and "cc". The fields are included when the message is saved. The Print and Type commands override this command.

dp [msglist]
dt [msglist]  
Delete the specified messages from the mailbox and print the next message after the last one deleted. Roughly equivalent to a delete command followed by a print command.
echo string ...
   Echo the given strings (like echo(1)).

edit [msglist]
   Edit the given messages. The messages are placed in a temporary file and the "EDITOR" variable is used to get the name of the editor (see ENVIRONMENT VARIABLES). Default editor is ed(1).

exit
exit
   Exit from mailx, without changing the mailbox. No messages are saved in the mbox (see also quit).

file [filename]
folder [filename]
   Quit from the current file of messages and read in the specified file. Several special characters are recognized when used as file names, with the following substitutions:
   %  the current mailbox.
   $user  the mailbox for user.
   #  the previous file.
   &  the current mbox.
   Default file is the current mailbox.

folders
   Print the names of the files in the directory set by the "folder" variable (see ENVIRONMENT VARIABLES).

followup [message]
   Respond to a message, recording the response in a file whose name is derived from the author of the message. Overrides the "record" variable, if set. See also the Followup, Save, and Copy commands and "outfolder" (ENVIRONMENT VARIABLES).

Followup [msglist]
   Respond to the first message in the msglist, sending the message to the author of each message in the msglist. The subject line is taken from the first message and the response is recorded in a file whose name is derived from the author of the first message. See also the followup, Save, and Copy commands and "outfolder" (ENVIRONMENT VARIABLES).

from [msglist]
   Print the header summary for the specified messages.

group alias name ...
alias alias name ...
   Declare an alias for the given names. The names are substituted when alias is used as a recipient. Useful in the .mailrc
headers [message]
   Print the page of headers which includes the message specified.
   The "screen" variable sets the number of headers per page (see
   ENVIRONMENT VARIABLES). See also the z command.

help
   Print a summary of commands.

hold [msglist]
preserve [msglist]
   Hold the specified messages in the mailbox.

if s | r
mail-commands
else
mail-commands
endif
   Conditional execution, where s executes following mail-
   commands, up to an else or endif, if the program is in send
   mode, and r causes the mail-commands to be executed only in
   receive mode. Useful in the .mailrc file.

ignore header-field ...
discard header-field ...
   Suppress printing of the specified header fields when display-
   ing messages on the screen. Examples of header fields to
   ignore are "status" and "cc". All fields are included when the
   message is saved. The Print and Type commands override this
   command.

list
   Print all commands available. No explanation is given.

mail name ...
   Mail a message to the specified users.

mbox [msglist]
   Arrange for the given messages to end up in the standard mbox
   save file when mailx terminates normally. See "MBOX"
   (ENVIRONMENT VARIABLES) for a description of this file.
   See also the exit and quit commands.

next [message]
   Go to next message matching message. A msglist may be speci-
   fied, but in this case the first valid message in the list is the
   only one used. This is useful for jumping to the next message
   from a specific user, because the name would be taken as a com-
   mand in the absence of a real command. See the discussion of
   msglists above for a description of possible message
Pipe the message through the given shell-command. The message is treated as if it were read. If no arguments are given, the current message is piped through the command specified by the value of the "cmd" variable. If the "page" variable is set, a form feed character is inserted after each message (see ENVIRONMENT VARIABLES).

Preserve the specified messages in the mailbox.

Print the specified messages on the screen including all header fields. Overrides suppression of fields by the ignore command.

Print the specified messages. If "crt" is set, the messages longer than the number of lines specified by the "crt" variable are paged through the command specified by the "PAGER" variable. The default command is pg(1) (see ENVIRONMENT VARIABLES).

Exit from mailx, storing messages that were read in mbox and unread messages in the mailbox. Messages that have been explicitly saved in a file are deleted.

Send a response to the author of each message in the msglist. The subject line is taken from the first message. If "record" is set to a filename, the response is saved at the end of that file (see ENVIRONMENT VARIABLES).

Reply to the specified message including all other recipients of the message. If "record" is set to a filename, the response is saved at the end of that file (see ENVIRONMENT VARIABLES).

Save the specified messages in a file whose name is derived from the author of the first message. The name of the file is taken to be the author's name with all network addressing stripped off. See also the Copy, followup, and Followup commands and
"outfolder" (ENVIRONMENT VARIABLES).

save [filename]
save [msglist] filename
Save the specified messages in the given file. The file is created if it does not exist. The message is deleted from the mailbox when mailx terminates unless "keepsave" is set (see also ENVIRONMENT VARIABLES and the exit and quit commands).

set
set name
set name=string
set name=number
Define a variable called name. The variable may be given a null, string, or numeric value. Set by itself prints all defined variables and their values. See ENVIRONMENT VARIABLES for detailed descriptions of the mailx variables.

shell
Invoke an interactive shell (see also "SHELL" (ENVIRONMENT VARIABLES)).

size [msglist]
Print the size in characters of the specified messages.

source filename
Read commands from the given file and return to command mode.

top [msglist]
Print the top few lines of the specified messages. If the "toplines" variable is set, it is taken as the number of lines to print (see ENVIRONMENT VARIABLES). The default is 5.

touch [msglist]
Touch the specified messages. If any message in msglist is not specifically saved in a file, it is placed in the mbox upon normal termination. See exit and quit.

Type [msglist]
Print [msglist]
Print the specified messages on the screen including all header fields. Overrides suppression of fields by the ignore command.

type [msglist]
print [msglist]
Print the specified messages. If "crt" is set, the messages longer than the number of lines specified by the "crt" variable are paged through the command specified by the "PAGER" variable. The default command is pg(1) (see ENVIRONMENT VARIABLES).
MAILX(1)

VARIABLES).

undele [msglist]
  Restore the specified deleted messages. This only restores
  messages deleted in the current mail session. If "autoprint" is
  set, the last message of those restored is printed (see
  ENVIRONMENT VARIABLES).

unset name ...
  Erase the specified variables. If the variable was imported
  from the execution environment (i.e., a shell variable) then it
  cannot be erased.

version
  Print the current version and release date.

visual [msglist]
  Edit the given messages with a screen editor. The messages
  are placed in a temporary file and the "VISUAL" variable is
  used to get the name of the editor (see ENVIRONMENT VARI-
  ABLES).

write [msglist] filename
  Write the given messages on the specified file minus the header
  and trailing blank line. Otherwise equivalent to the save com-
  mand.

xit
exit
  Exit from mailx, without changing the mailbox. No messages
  are saved in the mbox (see also quit).

Z[+/-]
  Scroll the header display forward or backward one screen-full.
  The number of headers displayed is set by the "screen" vari-
  able (see ENVIRONMENT VARIABLES).

TILDE ESCAPES
  The following commands may be entered only from input mode, by
  beginning a line with the tilde escape character (~). See "escape"
  (ENVIRONMENT VARIABLES) for changing this special character.

~! shell-command
  Escape to the shell.

~
  Simulate end of file (terminate message input).

~: mail-command
  ~ mail-command
  Perform the command-level request. Valid only when sending a
  message while reading mail.
-?  Print a summary of tilde escapes.

-\ A  Insert the autograph string "Sign" into the message (see ENVIRONMENT VARIABLES).

-\ a  Insert the autograph string "sign" into the message (see ENVIRONMENT VARIABLES).

-\ b \ name ...  Add the names to the blind carbon copy (Bcc) list.

-\ c \ name ...  Add the names to the carbon copy (Cc) list.

-\ d  Read in the dead.letter file. See "DEAD" (ENVIRONMENT VARIABLES) for a description of this file.

-\ e  Invoke the editor on the partial message. See also "EDITOR" (ENVIRONMENT VARIABLES).

-\ f \ [msglist]  Forward the specified messages. The messages are inserted into the message, without alteration.

-\ h  Prompt for Subject line and To, Cc, and Bcc lists. If the field is displayed with an initial value, it may be edited as if you had just typed it.

-\ i \ string  Insert the value of the named variable into the text of the message. For example, -\ A is equivalent to "i Sign."

-\ m \ [msglist]  Insert the specified messages into the letter, shifting the new text to the right one tab stop. Valid only when sending a message while reading mail.

-\ p  Print the message being entered.

-\ q  Quit from input mode by simulating an interrupt. If the body of the message is not null, the partial message is saved in dead.letter. See "DEAD" (ENVIRONMENT VARIABLES) for a description of this file.
MAILX(1)

`-r filename`
`-< filename`
`-< !shell-command`
Read in the specified file. If the argument begins with an exclamation point (!), the rest of the string is taken as an arbitrary shell command and is executed, with the standard output inserted into the message.

`-s string ...`
Set the subject line to \texttt{string}.

`-t name ...`
Add the given \texttt{names} to the To list.

`-v`
Invoke a preferred screen editor on the partial message. See also "VISUAL" (ENVIRONMENT VARIABLES).

`-w filename`
Write the partial message onto the given file without the header.

`-x`
Exit as with `-q` except the message is not saved in \texttt{dead.letter}.

`-! shell-command`
Pipe the body of the message through the given \texttt{shell-command}. If the \texttt{shell-command} returns a successful exit status, the output of the command replaces the message.

ENVIRONMENT VARIABLES
The following are environment variables taken from the execution environment and are not alterable within \texttt{mailx}.

\texttt{HOME=directory}
The user's home directory during execution of \texttt{mailx}.

\texttt{MAILRC=filename}
The name of the start-up file. Default is \texttt{$HOME/.mailrc}.

The following variables are internal \texttt{mailx} variables. They may be imported from the execution environment or set via the \texttt{set} command at any time. The \texttt{unset} command may be used to erase variables.

\texttt{allnet}
All network names whose last component (login name) match are treated as identical. This causes the msglist message specifications to behave similarly. Default is noallnet. See also the alternates command and the meteo variable.
append
    Upon termination, append messages to the end of the *mbox* file
    instead of prepending them. Default is noappend.

askcc
    Prompt for the Cc list after message is entered. Default is
    noaskcc.

asksub
    Prompt for subject if it is not specified on the command line with
    the -s option. Enabled by default.

autoprint
    Enable automatic printing of messages after delete and undelete
    commands. Default is noautoprint.

bang
    Enable the special-casing of exclamation points (!) in shell
    escape command lines as in *vi*(1). Default is nobang.

cmd=shell·command
    Set the default command for the pipe command. No default
    value.

conv=conversion
    Convert *uucp*(1) addresses to the specified address style. The
    only valid conversion is *internet*, which requires a mail
    delivery program conforming to the RFC822 standard for elec-
    tronic mail addressing. Conversion is disabled by default. See
    also "sendmail" and the -U command line option.

crt=number
    Pipe messages having more than number lines through the com-
    mand specified by the value of the "PAGER" variable (*pg*(1) by
    default). Disabled by default.

DEAD=filename
    The name of the file in which to save partial letters in case of
    untimely interrupt or delivery errors. Default is
    $HOME/dead.letter.

debug
    Enable verbose diagnostics for debugging. Messages are not
    delivered. Default is nodebug.

dot
    Take a period on a line by itself during input from a terminal as
    end-of-file. Default is nodot.

EDITOR=shell·command
    The command to run when the edit or "e command is used.
Default is `ed(1)`.

`escape=c`
Substitute c for the `escape character`.

`folder=directory`
The directory for saving standard mail files. User specified file names beginning with a plus (+) are expanded by preceding the filename with this directory name to obtain the real filename. If `directory` does not start with a slash (/), `$HOME` is prepended to it. In order to use the plus (+) construct on a `mailx` command line, `folder` must be an exported `sh` environment variable. There is no default for the `folder` variable. See also `outfolder` below.

`header`
Enable printing of the header summary when entering `mailx`. Enabled by default.

`hold`
Preserve all messages that are read in the `mailbox` instead of putting them in the standard `mbox` save file. Default is `nohold`.

`ignore`
Ignore interrupts while entering messages. Handy for noisy dial-up lines. Default is `noignore`.

`ignoreeof`
Ignore end-of-file during message input. Input must be terminated by a period (.) on a line by itself or by the `.` command. Default is `noignoreeof`. See also "dot" above.

`keep`
When the `mailbox` is empty, truncate it to zero length instead of removing it. Disabled by default.

`keepsave`
Keep messages that have been saved in other files in the `mailbox` instead of deleting them. Default is `nokeepsave`.

`MBOX=filename`
The name of the file to save messages which have been read. The `xit` command overrides this function, as does saving the message explicitly in another file. Default is `$HOME/mbox`.

`metoo`
If your login appears as a recipient, do not delete it from the list. Default is `nometoo`.

`LISTER=shell-command`
The command (and options) to use when listing the contents of
the folder directory. The default is \texttt{ls(1)}.

\textbf{onehop}

When responding to a message that was originally sent to several recipients, the other recipient addresses are normally forced to be relative to the originating author's machine for the response. This flag disables alteration of the recipients' addresses, improving efficiency in a network where all machines can send directly to all other machines (i.e., one hop away).

\textbf{outfolder}

Put the files used to record outgoing messages in the directory specified by the folder variable unless the pathname is absolute. Default is \texttt{noutfolder}. See folder above and the Save, Copy, followup, and Followup commands.

\textbf{page}

Used with the pipe command to insert a form feed after each message sent through the pipe. Default is \texttt{nopage}.

\textbf{PAGER=shell-command}

Use the command as a filter for paginating output. This can also be used to specify the options to be used. Default is \texttt{pg(1)}.

\textbf{prompt=string}

Set the command mode prompt to \texttt{string}. Default is "? ".

\textbf{quiet}

Do not print the opening message and version when entering \texttt{mailx}. Default is \texttt{noquiet}.

\textbf{record=filename}

Record all outgoing mail in \texttt{filename}. Disabled by default. See also \texttt{outfolder} above.

\textbf{save}

Enable saving of messages in \texttt{dead.letter} on interrupt or delivery error. See "DEAD" for a description of this file. Enabled by default.

\textbf{screen=number}

Set the number of lines in a screen-full of headers for the headers command.

\textbf{sendmail=shell-command}

Alternate command for delivering messages. Default is \texttt{mail(1)}.

\textbf{sendwait}

Wait for background mailer to finish before returning. Default
is nosendwait.

SHELL=shell-command
The name of a preferred command interpreter. Default is sh(1).

showto
When displaying the header summary and the message is from
you, print the recipient's name instead of the author's name.

sign=string
The variable inserted into the text of a message when the "a
(autograph) command is given. No default. See also "I (TILDE
ESCAPES).

Sign=string
The variable inserted into the text of a message when the "A
command is given. No default. See also "I (TILDE ESCAPES).

toplines=number
The number of lines of header to print with the top command.
Default is 5.

VISUAL=shell-command
The name of a preferred screen editor. Default is vi(1).

FILES
$HOME/.mailrc personal start-up file
$HOME/mbox secondary storage file
/usr/mail/* post office directory
/usr/lib/mailx/mailx.help* help message files
/usr/lib/mailx/mailx.rc global start-up file
/tmp/R[emqtsx]* temporary files

SEE ALSO
mail(1), pg(1), ls(1), uucp(1).

RESTRICTIONS
Where shell-command is shown as valid, arguments are not always
allowed. Experimentation is recommended.

Internal variables imported from the execution environment cannot
be unset.

The full internet addressing is not fully supported by mailx. The
new standards are still evolving.

Attempts to send a message having a line consisting only of a "."
are treated as the end of the message by mail(1), the standard mail
delivery program.
[This page left blank.]
NAME
make - maintain, update, and regenerate groups of programs

SYNOPSIS
[-t] [-d] [-q] [names]

DESCRIPTION
Make executes commands in makefile to update one or more target files designated by names. Name is typically a program. If no -f option is present, makefile, Makefile, s.makefile, and s.Makefile are tried in order. If makefile is -, the standard input is taken. More than one - makefile option pair may appear.

Make updates a target only if that file depends on other newer files (the prerequisites for the target). Make recursively adds all prerequisite files of a target to the list of targets. Make assumes that missing files are out-of-date.

Makefile contains a sequence of entries that specify dependencies (which files depend on other files). The first line of an entry is a blank-separated, non-null list of targets, then a :, then a (possibly null) list of prerequisite files or dependencies. Text following a ; and all following lines that begin with a tab are shell commands to be executed to update the target. Shell commands may be continued across lines with the \<backslash><new-line> sequence. Everything printed by make (except the initial tab) is passed directly to the shell as is. Thus,

```
    echo a\n    b
```

produces

```
    ab
```

exactly the same as the shell would.

Sharp (#) and new-line surround comments.

The first line that does not begin with a tab or # begins a new dependency or macro definition.

The following makefile says that pgm depends on two files a.o and b.o, and that they in turn depend on their corresponding source files (a.c and b.c) and a common file incl.h:

```
    pgm: a.o b.o
        cc a.o b.o -o pgm
    a.o: incl.h a.c
        cc -c a.c
    b.o: incl.h b.c
        cc -c b.c
```

Make executes command lines one at a time, each by its own shell.
The first one or two characters in a command can be the following: -, @, -@, or @-. If @ is present, make does not print the command. If - is present, make ignores an error.
Make prints each command line when the command is executed unless the -s option is present, or the entry .SILENT: is in makefile, or the initial character sequence contains a @.

The -n option specifies printing without execution; however, if the command line has the string $(MAKE) in it, the line is always executed (see discussion of the MAKEFLAGS macro under Environment).

The -t (touch) option updates the modified date of a file without executing any commands.

Commands returning non-zero status normally terminate make. Make ignores errors if the -i option is present, or the entry .IGNORE: appears in makefile, or the initial character sequence of the command contains - .. If the -k option is present, make abandons work on the current entry, but continues on other branches that do not depend on that entry.

The -b option allows old makefiles (those written for the old version of make) to run without errors. The difference between the old version of make and this version is that this version requires all dependency lines to have a (possibly null or implicit) command associated with them. The previous version of make assumed if no command was specified explicitly that the command was null.

Pressing interrupt or quit deletes the target unless the target is a dependency of the special name .PRECIOUS. (See SPECIAL NAMES below)

OPTIONS
-f makefile Assumes makefile is the name of a description file. A file name of - denotes the standard input. The contents of makefile override the built-in rules if they are present.
-p Print out the complete set of macro definitions and target descriptions.
-i Ignore error codes returned by invoked commands. Make also enters this mode if the fake target name .IGNORE appears in the description file.
-k Abandon work on the current entry, but continue on other branches that do not depend on that entry.
-s Enter silent mode, do not print command lines before executing. Make also enters this mode if the fake file name .SILENT appears in the description file.
-r Do not use the built-in rules.
-n No execute mode. Print commands, but do not execute them. Note that make prints lines beginning with an @ when this option is used.
-b Enter compatibility mode for old makefiles.
-e Override assignments in makefiles with the environment variables.
-m  Print a memory map showing text, data, and stack. This option is a no-operation on systems without the getu system call.

-t  Change the dates on the target files so that they are up-to-date, rather than issuing the usual commands (see touch(1)).

-d  Enter debug mode. Print out detailed information on files and the times when they were examined.

-q  Question. The make command returns a zero status code if the file is up-to-date; returns a non-zero status code otherwise.

SPECIAL NAMES

.DEFAULT
Use the commands associated with the name .DEFAULT, if it exists, when a file must be made but there are no explicit commands or relevant built-in rules.

.PRECIOUS
Do not remove dependents of this file when quit or interrupt is pressed. Quit and interrupt are signals that are usually generated by the rubout and break keys.

.SILENT
Same effect as the -s option.

.IGNORE
Same effect as the -i option.

Environment
Make reads the environment; it assumes all variables to be macro definitions and processes them as such. Make processes environment variables before any makefile and after the internal rules; thus, macro assignments in a makefile override environment variables. The -e option overrides the macro assignments in a makefile with the environment.

The MAKEFLAGS environment variable may contain any legal input option (except -f, -p, and -d) defined for the command line. Further, upon invocation, make creates MAKEFLAGS if it is not in the environment, puts the current options into it, and passes it on to invocations of commands. Thus, MAKEFLAGS always contains the current input options. This proves very useful for super-makes. In fact, as noted above, when the -n option is used, the command $(MAKE) is executed anyway. Therefore, one can perform a make -n recursively on a whole software system to see what would have been executed, because the -n is put in MAKEFLAGS and passed to further invocations of $(MAKE). This is one way of debugging all of the makefiles for a software project without actually making changes to any files.

Macros
Entries of the form string1 = string2 are macro definitions. String2 is defined as all characters up to a comment character or an
unescaped newline. Subsequent appearances of 
\$(string1 [: subst1 = subst2]) are replaced by string2. The 
parentheses are optional if a single character macro name is used 
and there is no substitute sequence. The optional : subst1 = subst2 
is a substitute sequence. If it is specified, all non-overlapping 
ocurrences of subst1 in the named macro are replaced by subst2. 
Strings (for the purposes of this type of substitution) are delimit­
ed by blanks, tabs, new-line characters, and beginnings of 
lines. An example of the use of the substitute sequence is shown 
under Libraries.

Internal Macros

The five internally maintained macros are useful for writing rules 
for building targets.

$* The macro $* stands for the file name part of the current 
dependent with the suffix deleted. Make evaluates $* only for 
inference rules.

$@ The $@ macro stands for the full target name of the current 
target. Make evaluates $@ only for explicitly named dependen-
cies.

$< The $< macro (only evaluated for inference rules or the 
.DEFAULT rule) is the module which is out-of-date with 
respect to the target (i.e., the generated dependent file 
name). Thus, in the .c.o rule, the $< macro would evaluate to 
the .c file. An example for making optimized .o files from .c 
files is:

.c.o:
 cc -c -O $*.c

or:

.c.o:
 cc -c -O $<

$? The $? macro (evaluated when explicit rules from the makefile 
are evaluated) is the list of prerequisites that are out-of-date 
with respect to the target; essentially, those modules which 
must be rebuilt.

$% The $% macro is only evaluated when the target is an archive 
library member of the form lib(file.o). In this case, $@ evalu-
ates to lib and $% evaluates to the library member, file.o.

Each macro, except $?, has an alternative form. An upper case D 
or F appended to any of the four macros changes the meaning to 
directory part for D and file part for F. Thus, $(@D) refers to the 
directory part of the string $@. If there is no directory part, ./ is 
generated.

Suffixes

Certain names (for instance, those ending with .o) have inferable 
prerequisites such as .c, .s, etc. If no update commands for such 
a file appear in makefile, and if an inferable prerequisite exists, 
make compiles that prerequisite to create the target. In this case,
make has inference rules which allow building files from other files by examining the suffixes and determining an appropriate inference rule to use. The default inference rules are:

```
.c .c" .sh .sh" .c.o .c".c .s.o .s".o .y.o .y".o
.l.o .l".o .y.c .y".c .l.c .c.a .c".a .s".a .h".h
```

The source file rules.c contains the internal rules for make. These rules can be locally modified. To print out the rules compiled into the make in a form suitable for recompilation, the following command is used:

```
make -fp - 2>/dev/null </dev/null
```

The only peculiarity in this output is the (null) string which printf(3S) prints when handed a null string.

A tilde in the above rules refers to an SCCS file (see sccsfile(4)). Thus, the rule .c".o transforms an SCCS C source file into an object file (.o). Because the s. of the SCCS files is a prefix, it is incompatible with the suffix point-of-view taken by make. The tilde is a way of changing any file reference into an SCCS file reference.

A rule with only one suffix (i.e. .c:) defines how to build x from x.c. In effect, the other suffix is null. This is useful for building targets from only one source file (e.g., shell procedures, simple C programs).

Additional suffixes are given as the dependency list for .SUFFIXES. Order is significant; the first possible name for which both a file and a rule exist is inferred as a prerequisite. The default list is:

```
.SUFFIXES: .o .c .y .l .s
```

Here again, the above command for printing the internal rules displays the list of suffixes implemented. Multiple suffix lists accumulate; .SUFFIXES: with no dependencies clears the list of suffixes.

**Inference Rules**

The first example can be specified more briefly:

```
pgm: a.o b.o
c c a.o b.o -o pgm
a.o b.o: incl.h
```

This abbreviation can be made because make has a set of internal rules for building files. The user may add rules to this list by simply putting them in the makefile.

The default inference rules use certain macros to permit the inclusion of optional matter in any resulting commands. For example, CFLAGS, LFLAGS, and YFLAGS are used for compiler options to cc(1), lex(1), and yacc(1) respectively. Again, the command suggested for examining the current rules (see suffixes) is recommended.
The inference of prerequisites can be controlled. The rule to create a file with suffix `.o` from a file with suffix `.c` is specified as an entry with `.c.o:` as the target and no dependents. Shell commands associated with the target define the rule for making a `.o` file from a `.c` file. Any target that has no slashes in it and starts with a dot is identified as a rule and not a true target.

**Libraries**

If a target or dependency name contains parentheses, `make` assumes that the file is an archive library, the string within parentheses referring to a member within the library. For example, `lib(file.o)` and `$(LIB)(file.o)` both refer to an archive library which contains `file.o`. (This assumes the `LIB` macro has been previously defined.) The expression

$$ (\text{LIB})(\text{file1.o file2.o}) $$

is not legal. Rules pertaining to archive libraries have the form `.XX.a` where the `XX` is the suffix from which the archive member is to be made. An unfortunate byproduct of the current implementation requires the `XX` to be different from the suffix of the archive member. Thus, one cannot have `lib(file.o)` depend upon `file.o` explicitly. The most common use of the archive interface follows. Here, we assume the source files are all C type source:

```makefile
lib:
    lib(file1.o) lib(file2.o) lib(file3.o)
    @echo lib is now up to date
    .c.a:
    $(CC) -c $(CFLAGS) $<
    ar rv $@ $*.o
    rm -f $*.o
```

In fact, the `.c.a` rule listed above is built into `make` and is unnecessary in this example. A more interesting, but more limited example of an archive library maintenance construction follows:

```makefile
lib:
    lib(file1.o) lib(file2.o) lib(file3.o)
    $(CC) -c $(CFLAGS) $(?:.o=.c)
    ar rv lib $?
    rm $? @echo lib is now up to date
    .c.a:;
```

Here the substitution mode of the macro expansions is used. The `$?` list is the set of object file names (inside `lib`) whose C source files are out-of-date. The substitution mode translates the `.o` to `.c`. (One cannot transform to `.c`.) Note also, the disabling of the `.c.a:` rule, which would have created each object file, one by one. This particular construct speeds up archive library maintenance considerably. This type of construct becomes very cumbersome if the archive library contains a mix of assembly programs and C programs.

**FILES**

[Mm]akefile and s. [Mm]akefile
SEE ALSO
cc(1), cd(1), lex(1), sh(1), yacc(1).
A Program for Maintaining Computer Programs (make) and Augmented Version of make in the Support Tools Guide.

RESTRICTIONS
Some commands return non-zero status inappropriately; use -i to overcome the difficulty.
Filenames with the characters =, :, or @ do not work.
Commands that are directly executed by the shell, notably cd(1), are ineffectual across new-lines in make.
The syntax (lib(file1.o file2.o file3.o) is not valid. You cannot build lib(file.o) from file.o.
The macro $(a:.o=.c-) does not work.
[This page left blank.]
NAME
makekey - generate encryption key

SYNOPSIS
/usr/lib/makekey

DESCRIPTION
Makekey improves the usefulness of encryption schemes depending on a key by increasing the amount of time required to search the key space. It reads 10 bytes from its standard input, and writes 13 bytes on its standard output. The output depends on the input in a way intended to be difficult to compute (i.e., to require a substantial fraction of a second).

The first eight input bytes (the input key) can be arbitrary ASCII characters. The last two (the salt) are best chosen from the set of digits, ., /, and upper- and lower-case letters. The salt characters are repeated as the first two characters of the output. The remaining 11 output characters are chosen from the same set as the salt and constitute the output key.

The transformation performed is essentially the following: the salt is used to select one of 4,096 cryptographic machines all based on the National Bureau of Standards DES algorithm, but broken in 4,096 different ways. Using the input key as key, a constant string is fed into the machine and recirculated a number of times. The 64 bits that come out are distributed into the 66 output key bits in the result.

Makekey is intended for programs that perform encryption (e.g., ed(1) and crypt(1)). Usually, its input and output are pipes.

SEE ALSO
crypt(1), ed(1), passwd(4).
NAME
man - print entries in this manual

SYNOPSIS
man [ options ] [ section ] titles

DESCRIPTION
Man is an online help system and is available if your system administrator installed the manual files on your system. Man locates and prints an entry of this manual, the Programmer Reference Manual or the Superuser Reference Manual named title in the specified section. The title is entered in lower case. The section number may not have a letter suffix. If no section is specified, all manuals are searched for title and all occurrences of it are printed. Section may be changed before each title.

OPTIONS
Man examines the environment variable $TERM (see environ(5)) and attempts to select options that adapt the output to the terminal being used. The -Tterm option overrides the value of $TERM; in particular, one should use -Tlpr when sending the output of man to a line printer.

-Tterm
Print the entry as appropriate for terminal type term. For a list of the recognized values of term, type help term2. The default value of term is 450.

-w Print on the standard output only the path names of the entries, relative to /usr/catman, or to the current directory for -d option.

-d Search the current directory rather than /usr/catman; requires the full file name (e.g., cu.1c, rather than just cu).

-c Invoke col(1); note that col(1) is invoked automatically by man unless term is one of 300, 300s, 450, 37, 4000a, 382, 4014, tek, 1620, or X.

EXAMPLE
To display the entry for the man command, enter
man 1 man

FILES
/usr/catman/?_man/man[1-8]/* preformatted manual entries

WARNING
Man prints manual entries that were formatted by nroff(1).

Entries are originally formatted with terminal type 32, and are printed using the correct terminal filters as derived from the -Tterm and $TERM settings. Typesetting or other non-standard printing of manual entries may require installation of Documenter's Workbench.
[This page left blank.]
NAME
mcs - manipulate the object file comment section

SYNOPSIS
mcs
   [ options ] object-files

DESCRIPTION
This command is available on the 5000/30/35/50/55 Release 2.00 only. The mcs command manipulates the comment section, named ".comment", in an object file. It is used to add to, delete, print, and compress the contents of the comment section. Mcs must be given one or more of the options specified below. It takes each of the options given and applies them in order to the object-files file list.

If an object file is an archive, the file is treated as a set of individual object files. For example, if the -a option is specified, the string is appended to the comment section of each archive element.

OPTIONS
Options may be used in any order, in any combination, and may appear anywhere in the command line.

-a "string"
The -a option appends string to the comment section of the object-files.

-c The -c option compresses the contents of the comment section. All duplicate entries are removed. The ordering of the remaining entries is not disturbed.

-d The -d option deletes the contents of the comment section from the object file. The object file comment section header is also removed.

-n "name"
The -n option specifies the name of the section to access. By default, mcs deals with the section named ".comment". This option can be used to specify another section.

-p The -p option prints the contents of the comment section on the standard output. If more than one object file name is specified, each entry printed is tagged by the name of the file from which it was extracted, using the format filename:string.

EXAMPLES
mcs -p file           # Print file's comment section.
mcs -a string file    # Append string to file's comment section

FILES
/usr/tmp/mcs*         temporary files
/usr/tmp/*            temporary files
/usr/tmp              the usual temporary file directory, but can be redefined by setting the environment variable TMPDIR (see tempnam () in tmpnam (3S)).

SEE ALSO
cpp(1), a.out(4).
WARNINGS
Mcs can not add new sections or delete existing sections to executable objects configured for paging (see a.out (4)).
NAME
    mesg - permit or deny messages

SYNOPSIS
    mesg [ n ] [ y ]

DESCRIPTION
    Mesg permits or denies messages to be received by your terminal from another user via write(1).

    Mesg with argument n forbids messages via write(1) by revoking non-user write permission on the terminal of the user. Mesg with argument y reinstates permission. Mesg with no argument reports the current state without changing it.

FILES
    /dev/tty*

SEE ALSO
    write(1).

DIAGNOSTICS
    The exit status is 0 if messages are permitted, 1 if they are denied, or 2 if an error occurred.
[This page left blank.]
NAME
mkdir - make a directory

SYNOPSIS
mkdir dir ... 

DESCRIPTION
Mkdir creates specified directories. Standard entries . (dot) for the directory itself and .. (dotdot) for its parent are made automatically.

Mkdir typically creates directories in mode 777 which are readable, writable, and searchable by the owner, group, and everyone.
Mkdir requires write permission in the parent directory.

EXAMPLE
To make may, june, and july directories in the current directory enter
mkdir may june july

SEE ALSO
sh(1), rm(1), umask(1).

DIAGNOSTICS
Exit code is 0 if all directories are successfully made or non-zero if and error occurred.
MKDIR(1)

[This page left blank.]
NAME
mklost+found - make a lost+found directory for fsck

SYNOPSIS
/etc/mklost+found

DESCRIPTION
Not on 7000/40.
A directory lost+found is created in the current directory and a
color of empty files are created therein and then removed so that
there are empty slots for fsck(1).
This command should be run immediately after first mounting and
changing directory to a newly created file system.
For small file systems, it is sufficient (and much faster) to simply
make a lost+found directory. Up to 30 files can be recovered in it.

SEE ALSO
fsck(1)
[This page left blank.]
NAME
mkstr - create an error message file by massaging C source

SYNOPSIS
mkstr [- ] messagefile prefix file ...

DESCRIPTION
5000 Series Systems only.

Mkstr creates files of error messages. Its use can make programs
with large numbers of error diagnostics much smaller, and reduce
system overhead in running the program because the error mes-
sages do not have to be constantly swapped in and out.

Mkstr processes each of the specified files, placing a massaged
version of the input file in a file whose name consists of the speci-
ified prefix and the original name. A typical usage of mkstr would be

    mkstr pistrings xx * . c

This command would place all the error messages from the C source
files in the current directory into the file pistrings and place pro-
cessed copies of the source for these files into files whose names
are prefixed with xx.

To process the error messages in the source to the message file
mkstr keys on the string

    error(" in the input stream. Each time it occurs, the C string starting at
the double quote is placed in the message file followed by a null
character and a new-line character; the null character terminates
the message so it can be easily used when retrieved, the new-line
character makes it possible to sensibly cat the error message file to
see its contents. The altered copy of the input file then contains a
lseek pointer into the file which can be used to retrieve the mes-
sage, i.e.:

    char efilname[] = "/usr/lib/pi_strings";
    int efil = -1;

    error(a1, a2, a3, a4)
    {
        char buf[256];

        if (efil < 0) {
            efil = open(efilname, 0);
            if (efil < 0) {
                oops:
                    perror(efilname);
                    exit(1);
            }
    }

    if (lseek(efil, (long) a1, 0) | | read(efil, buf, 256) <= 0)
MKSTR(1B)

... goto oops;
   printf(buf, a2, a3, a4);
}

OPTIONS
   The optional - places the error messages at the end of the specified message file for recompiling part of a large mkstr ed program.

SEE ALSO
   lseek(2), xstr(1)
NAME

mm, osdd, checkmm - print/check documents formatted with the MM macros

SYNOPSIS

mm [ options ] [ files ]
osdd [ options ] [ files ]
checkmm [ files ]

DESCRIPTION

Not on 7000/40.

Mm can be used to type out documents using nroff and the MM text-formatting macro package. It has options to specify prepro­
cessing by tbl(1) and/or neqn (see eqn(1)) and postprocessing by
diff erent terminal-oriented output filters. The proper pipelines and
and the required arguments and flags for nroff and MM are generated,
depending on the options selected.

Mm reads the standard input when - is specified instead of any file
names. (Mentioning other files together with - leads to disaster.)
This option allows mm to be used as a filter, e.g.:

    cat report | mm -

Osdd is equivalent to the command mm -mosd. For more information
about the OSDD adapter macro package, see mosd(5).

Checkmm is a program for checking the contents of the named files
for errors in the use of the Memorandum Macros, missing or unbal­
anced neqn delimiters, and .EQ/.EN pairs. Note: The user need
not use the checkeq program (see eqn(1)). Appropriate messages
are produced. The program skips all directories, and if no file
name is given, standard input is read.

OPTIONS

Any other arguments or options (e.g., -rC3) other than those
below are passed to nroff or to MM, as appropriate. Such options
can occur in any order, but they must appear before the files
arguments. If no arguments are given, mm prints a list of its
options.

-Tterm Specifies the type of output terminal; for a list of recog­
nized values for term, type help term2. If this option is
not used, mm uses the value of the shell variable $TERM
from the environment (see profile(4) and environ(5)) as
the value of term, if $TERM is set; otherwise, mm uses 450
as the value of term. If several terminal types are speci­
fied, the last one takes precedence.

-12 Produces the document in 12-pitch. May be used when
$TERM is set to one of 300, 300s, 450, and 1620. (The
pitch switch on the DASI 300 and 300s terminals must be
manually set to 12 if this option is used.)

-c Invokes col(1); note that col(1) is invoked automatically by
mm unless term is one of 300, 300s, 450, 37, 4000a, 382,
4014, tek, 1620, and X.

-e Invokes neqn; also causes neqn to read the /usr/pub/eqnchar file (see eqnchar(5)).
-t Invokes tbl(1).
-E Invokes the -e option of nroff.
-y Uses the non-compacted version of the macros (see mm(5)).

EXAMPLE
Assuming that the shell variable $TERM is set in the environment to 450, the two command lines below are equivalent:

```
mm -t -rC3 -12 chapter*
```

```
tbl chapter* | nroff -cm -T450-12 -h -rC3
```

HINTS
1. Mm invokes nroff with the -h option. With this option, nroff assumes that the terminal has tabs set every 8 character positions.
2. Use the -olist option of nroff to specify ranges of pages to be output. Note, however, that mm, if invoked with one or more of the -e, -t, and - options, together with the -olist option of nroff may cause a harmless broken pipe diagnostic if the last page of the document is not specified in list.
3. If you use the -s option of nroff (to stop between pages of output), use line-feed (rather than return or new-line) to restart the output. The -s option of nroff does not work with the -c option of mm, or if mm automatically invokes col(1) (see -c option above).
4. If you mistake the kind of terminal the output from mm will be printed on, you get (often subtle) garbage; however, if you are redirecting output into a file, use the -T37 option, and then use the appropriate terminal filter when you actually print that file.

SEE ALSO
col(1), env(1), eqn(1), greek(1), mmt(1), nroff(1), tbl(1), profile(4), mm(5), mosd(5), term(5).

DIAGNOSTICS
```
mm mm: no input file
```
No arguments are readable files and mm is not used as a filter.
```
checkmm Cannot open filename
```
Unreadable file(s). The remaining output of the program is diagnostic of the source file.
NAME
mmt, mvt - typeset documents, viewgraphs, and slides

SYNOPSIS
mmt [ options ] [ files ]
mvt [ options ] [ files ]

DESCRIPTION
Not on 7000/40.

Mmt and mvt commands are very similar to mm(1), except that they both typeset their input via troff(1), as opposed to formatting it via nroff(1). Mmt uses the MM macro package, while mvt uses the Macro Package for View Graphs and Slides. These two commands have options to specify preprocessing by tbl(1) and/or pic(1) and/or eqn(1). The proper pipelines and the required arguments and options for troff(1) and for the macro packages are generated, depending on the options selected.

These commands read the standard input when - is specified instead of any file names.

Mvt is just a link to mmt.

OPTIONS
Options are given below. Any other arguments or options (e.g., -rC3) are passed to troff(1) or to the macro package, as appropriate. Such options can occur in any order, but they must appear before the files arguments. If no arguments are given, these commands print a list of their options.

-e Invoke eqn(1) and cause eqn to read the /usr/pub/eqnchar file (see eqnchar(5)).
-t Invoke tbl(1).
-p Invoke pic(1).
-Taps
Create output for an Autologic APS-5 phototypesetter and send it to the default destination at this installation.
-Tdest
Create output for troff device dest (see troff(1)). The output is sent through the appropriate postprocessor (see daps(1)).
-Tcat
Use otroff(1) to generate output for an on-line Wang CAT phototypesetter.
-D4014
Direct the output to a TEKTRONIX 4014 terminal via the tc(1) filter.
-Dtek
Same as -D4014.
-Di10
Direct the output to the local Imagen Imprint-10 laser printer.
-a Invoke the -a option of troff(1).

-y Cause mmt to use the non-compacted version of the macros. This is the default except when using -Tcat.

-z Invoke no output filter to process or redirect the output of troff(1).

HINT
Use the -olist option of troff(1) to specify ranges of pages to be output. Note, however, that these commands, if invoked with one or more of the -e, -t, and - options, together with the -olist option of troff(1) may cause a harmless broken pipe diagnostic if the last page of the document is not specified in list.

SEE ALSO
daps(1), env(1), eqn(1), mm(1), nroff(1), pic(1), tbl(1), tc(1), profile(4), environ(5), mm(5), mv(5).

DIAGNOSTICS
"m[mv]t: no input file" if none of the arguments is a readable file and the command is not used as a filter.
NAME
more, page - file perusal filter for crt viewing

SYNOPSIS
more [ -cdflsu ] [ -n ] [ +linenumber ] [ +/pattern ] [ name ... ]
page more options

DESCRIPTION
More is a filter which allows examination of a continuous text one
screenful at a time on a soft-copy terminal.

More normally pauses after each screenful, printing --More-- at the
bottom of the screen. If the user then presses a carriage return,
more displays one more line. If the user presses the space bar,
more displays another screenful.

Other possible responses are enumerated in COMMAND CHARAC-
TERS.

If more is reading from a file, rather than a pipe, then a percen-
tage is displayed along with the --More-- prompt. This gives the
fraction of the file (in characters, not lines) that has been read so
far.

If the standard output is not a teletype, then more acts just like
cat, except that a header is printed before each file (if there is
more than one).

Page.
If the program is invoked as page, then the screen is cleared
before each screenful is printed (but only if a full screenful is
being printed), and k - 1 rather than k - 2 lines are printed in each
screenful, where k is the number of lines the terminal can display.

Window size.
More looks in the file /etc/termcap to determine terminal charac-
teristics, and to determine the default window size. On a terminal
capable of displaying 24 lines, the default window size is 22 lines.

Environment.
More looks in the environment variable MORE to pre-set any flags
desired. For example, if you prefer to view files using the -c mode
of operation, or the sh command sequence MORE="c" ; export
MORE would cause all invocations of more, including invocations by
programs such as man and msgs, to use this mode. Normally, the
user places the command sequence which sets up the MORE
environment variable in the .profile file.

COMMAND CHARACTERS
Other sequences which may be typed when more pauses, and their
effects, are described below (i is an optional integer argument,
defaulting to 1).

The commands take effect immediately; that is, it is not necessary
to press a carriage return. Up to the time when the command char-
acter itself is given, the user may press the line kill character to
cancel the numerical argument being formed. In addition, the user
may press the erase character to redisplay the --More--(xx%) mes-
sage.

\texttt{i <space>}

Display \textit{i} more lines, (or another screenful if no argument
is given).

\texttt{\textasciitilde D}

Display 11 more lines (a \textit{scroll}). If \textit{i} is given, then the
scroll size is set to \textit{i}.

\texttt{d}

Same as \texttt{\textasciitilde D} (control-D).

\texttt{iz}

Same as typing a space except that \textit{i}, if present, becomes
the new window size.

\texttt{is}

Skip \textit{i} lines and print a screenful of lines.

\texttt{sf}

Skip \textit{i} screenfuls and print a screenful of lines.

\texttt{q}

Exit from \textit{more}.

\texttt{Q}

Exit from \textit{more}.

\texttt{=}

Display the current line number.

\texttt{v}

Help command; give a description of all the \textit{more}
commands.

\texttt{i/exp}\texttt{r}

Search for the \textit{i}-th occurrence of the regular expression
\textit{expr}.

If there are less than \textit{i} occurrences of \textit{expr}, and the input
is a file (rather than a pipe), then the position in the file
remains unchanged. Otherwise, a screenful is displayed,
starting two lines before the place where the expression
was found.

The user may use erase and kill characters to edit the regu-
lar expression. Erasing back past the first column cancels
the search command.

\texttt{in}

Search for the \textit{i}-th occurrence of the last regular expres-
sion entered.

\texttt{'}

(single quote) Go to the point from which the last search
started. If no search has been performed in the current
file, this command goes back to the beginning of the file.

\texttt{!cmd}

Invoke a shell with the command \textit{cmd}. The characters \texttt{%} and
\texttt{!} in \textit{cmd} are replaced with the current file name and the
previous shell command respectively. If there is no
current file name, \texttt{%} is not expanded. The sequences \texttt{%} and
\texttt{!} are replaced by \texttt{%} and \texttt{!} respectively.

\texttt{i:n}

Skip to the \textit{i}-th next file given in the command line (skips
to last file if \textit{n} is not sensible).

\texttt{i:p}

Skip to the \textit{i}-th previous file given in the command line. If
this command is given in the middle of printing out a file,
then more goes back to the beginning of the file. If \textit{i}
doesn't make sense, more skips back to the first file. If
more is not reading from a file, the bell is rung and nothing else happens.

: f Display the current file name and line number.
: q Same as q.
: Q Same as Q.
. (dot) Repeat the previous command.

At any time when output is being sent to the terminal, the user can press the quit key (normally control-\). More stops sending output, and displays the usual --More-- prompt. The user may then enter one of the above commands in the normal manner. Unfortunately, some output is lost when this is done, because any characters waiting in the output queue of the terminal are flushed when the quit signal occurs.

OPTIONS
The command line options are:

- n Use a window n lines long (where n is an integer) instead of the default window size.
- c Print each page by beginning at the top of the screen and erasing each line just before printing over it. This avoids scrolling the screen, making it easier to read while more is writing.

This option will be ignored if the terminal does not have the ability to clear to the end of a line.
- d Prompt the user with the message

    Hit space to continue, rubout to abort

at the end of each screenful.

- f Count logical, rather than screen lines. That is, do not fold long lines.

This option is recommended if nroff output is being piped through ul, since the latter may generate escape sequences. These escape sequences contain characters which would ordinarily occupy screen positions, but which do not print when they are sent to the terminal as part of an escape sequence. Thus more may think that lines are longer than they actually are, and fold lines erroneously.

- l Do not treat `L (form feed) specially.

If this option is not given, more pauses after any line that contains a `L, as if the end of a screenful had been reached. Also, if a file begins with a form feed, the screen will be cleared before the file is printed.

- s Squeeze multiple blank lines from the output, producing only one blank line.

Especially helpful when viewing nroff output, this option maximizes the useful information present on the screen.
-u Do not attempt to underline on the terminal.

Normally, more handles underlining such as produced by nroff in a manner appropriate to the particular terminal: if the terminal can perform underlining or has a stand-out mode, more generates appropriate escape sequences to enable underlining or stand-out mode for underlined information in the source file.

+l/linenumber
Start at linenumber.

+l/pattern
Start two lines before the line containing the regular expression pattern.

EXAMPLE
A sample usage of more in previewing nroff output would be
nroff -ms +2 doc.n | more -s

FILES
/etc/termcap Terminal data base
/usr/lib/more.help Help file

SEE ALSO
man(1), sh(1), environ(5)

RESTRICTIONS
When performing more, the user may not redirect stderr to any terminal (/dev/tty). To do so causes more to abort after displaying the first screen.
NAME
newform - change the format of a text file

SYNOPSIS

DESCRIPTION
Newform reads lines from the named files, or the standard input if no input file is named, and reproduces the lines on the standard output. Lines are reformatted in accordance with command line options in effect.

OPTIONS
Except for -s, command line options may appear in any order, may be repeated, and may be intermingled with the optional files. Command line options are processed in the order specified. For example, -e15 -l60 yields results different from -l60 -e15. Options are applied to all files on the command line.

-itabspec
(Input tab specification) Expands tabs to spaces, according to the tab specifications given. Tabspec recognizes all tab specification forms described in tabs(1). In addition, tabspec may be --, in which newform assumes that the tab specification is to be found in the first line read from the standard input (see fspec(4)). If no tabspec is given, tabspec defaults to -8. A tabspec of -0 expects no tabs; if any are found, they are treated as -1. The value for tabspec can not be greater than 46.

Newform does not prompt the user if a tabspec is to be read from the standard input (by use of -i-- or -o--).

-otabspec
(Output tab specification) Replaces spaces by tabs, according to the tab specifications given. The tab specifications are the same as for -itabspec. If no tabspec is given, tabspec defaults to -8. A tabspec of -0 means that no spaces are converted to tabs on output.

-ln
Sets the effective line length to n characters. If n is not entered, -l defaults to 72. If -l is not specified, the line length is assumed to be 80 characters. Tabs and backspaces are considered to be one character (use -i to expand tabs to spaces).

-ln must be used in conjunction with and precede one of the following options:

-bn or -en if the effective line length is less than the existing line length.

-pn or -an if the effective line length is greater than the existing line length.
NEWFORM(1)

-\texttt{bn}  Truncates \( n \) characters from the beginning of the line when the line length is greater than the effective line length (see -ln). If -b is not specified, or if \( n \) is omitted, newform truncates the number of characters necessary to obtain the effective line length.

-\texttt{en}  Same as -bn except that characters are truncated from the end of the line.

-\texttt{ck}  Changes the prefix/append character to \( k \). Default character for \( k \) is a space. (See -pn.)

-\texttt{pn}  Prefixes \( n \) characters (see -ck) to the beginning of a line when the line length is less than the effective line length. If -p is not specified, newform prefixes the number of characters necessary to obtain the effective line length.

-\texttt{an}  Same as -pn except characters are appended to the end of a line. (See also -ck.)

-\texttt{f}  Writes the tab specification format line on the standard output before any other lines are output. The tab specification format line corresponds to the format specified in the last -o option. If no -o option has been specified, the tab specification format line contains the default specification of -8.

-\texttt{s}  Removes leading characters on each line up to the first tab and places up to 8 of the removed characters at the end of the line. If more than 8 characters (not counting the first tab) are removed, the eighth character is replaced by a * and any characters to the right of it are discarded. The first tab is always discarded.

The characters removed are saved internally until all other options specified are applied to that line. The characters are then added at the end of the processed line. An error message and program exit occurs if this option is used on a file without a tab on each line.

**EXAMPLES**

To convert a file with leading digits, one or more tabs, and text on each line, to a file beginning with the text, all tabs after the first expanded to spaces, padded with spaces out to column 72 (or truncated to column 72), and the leading digits placed starting at column 73, the command would be:

\begin{verbatim}
newform -s -i -l -a -e file-name
\end{verbatim}

The -b option can be used to delete the sequence numbers from a COBOL program as follows:

\begin{verbatim}
newform -l1 -b7 file-name
\end{verbatim}

The -l1 must be used to set the effective line length shorter than any existing line in the file so that the -b option is activated.
DIAGNOSTICS
All diagnostics are fatal.

usage: . . .
not -s format
cannot open file
internal line too long

tabspec in error
intab spec indirection illegal

EXIT CODES
0 - normal execution
1 - for any error

SEE ALSO
csplit(1), tabs(1), fspec(4).

RESTRICTIONS
Newform normally only keeps track of physical characters; how-
ever, for the -i and -o options, newform keeps track of backspaces in order to line up tabs in the appropriate logical columns.

If the -f option is used, and the last -o option specified was -o--, and was preceded by either a -o-- or a -i-, the tab specification format line is incorrect.
NEWFORM(1)

[This page left blank.]
NAME
newgrp - log in to a new group

SYNOPSIS
newgrp [- ] [ group ]

DESCRIPTION
Newgrp changes a user's group identification. The user remains
logged in and the current directory is unchanged, but calculations
of access permissions to files are performed with respect to the new
real and effective group IDs. The user is always given a new
shell, replacing the current shell, by newgrp, regardless of
whether it terminated successfully or due to an error condition
(i.e., unknown group).

Exported variables retain their values after invoking newgrp;
however, all unexported variables are either reset to their default
value or set to null. System variables (such as PS1, PS2, PATH,
MAIL, and HOME), unless exported by the system or explicitly
exported by the user, are reset to default values. For example, a
user has a primary prompt string (PS1) other than $ (default) and
has not exported PS1. After an invocation of newgrp, successful
or not, their PS1 is now set to the default prompt string $. Note
that the shell command export (see sh(1)) is the method to export
variables so that they retain their assigned value when invoking
new shells.

With no arguments, newgrp changes the group identification back
to the group specified in the user's password file entry.

If the first argument to newgrp is a -, the environment is changed
to what would be expected if the user actually logged in again.

A password is demanded if the group has a password and the user
does not, or if the group has a password and the user is not listed
in /etc/group as being a member of that group.

FILES
/etc/group     system's group file
/etc/passwd    system's password file

SEE ALSO
login(1), sh(1), group(4), passwd(4), environ(5).

RESTRICTIONS
There is no convenient way to enter a password into /etc/group.
Use of group passwords is not encouraged, because, by their very
nature, they encourage poor security practices. Group passwords
may disappear in the future.
NEWGRP(1)

[This page left blank.]
NEWS(1)

NAME
  news - print news items

SYNOPSIS
  news [ -a ] [ -n ] [ -s ] [ items ]

DESCRIPTION
  News is used to keep the user informed of current events. By conven- 
tion, these events are described by files in the directory 
/usr/news.

When invoked without arguments, news prints the contents of all 
current files in /usr/news, most recent first, with each preceded 
by an appropriate header. News stores the currency time as the 
modification date of a file named .news_time in the home directory 
of the user (the identity of this directory is determined by the 
environment variable $HOME); only files more recent than this 
currency time are considered current.

Items are specific news items that are to be printed.

If a delete is pressed during the printing of a news item, printing 
stops and the next item is started. Another delete within one 
second of the first causes the program to terminate.

OPTIONS
  If any of the options below are used, news does not change the 
stored time. The -s option may be appropriate for a user's .profile 
file or the system's /etc/profile.

- a  Prints all items, regardless of currency.

- n  Reports only the names of the current items.

- s  Reports only how many current items exist.

FILES
  /etc/profile
  /usr/news/*
  $HOME/.news_time

SEE ALSO
  profile(4), environ(5).
[This page left blank.]
NAME
   nice - run a command at low priority

SYNOPSIS
   nice [ -increment ] command [ arguments ]

DESCRIPTION
   Nice executes command with a lower CPU scheduling priority. If the
   increment argument (in the range 1-19) is given, it is used; if not, an increment of 10 is assumed.
   
   An increment larger than 19 is equivalent to 19.
   
   The super-user may run commands with priority higher than normal by using a negative increment, e.g., --10.

SEE ALSO
   nohup(1), nice(2).

DIAGNOSTICS
   Nice returns the exit status of the subject command.
[This page left blank.]
NAME
nl - line numbering filter

SYNOPSIS

DESCRIPTION
Nl reads lines from the named file or the standard input if no file is
named and reproduces the lines on the standard output. Lines are
numbered on the left in accordance with the command options in
effect.

Nl views the text it reads in terms of logical pages. Line numbering
is reset at the start of each logical page. A logical page con­sists of
a header, a body, and a footer section. Empty sections are
valid. Different line numbering options are independently avail­able
for header, body, and footer (e.g. no numbering of header and
footer lines while numbering blank lines only in the body).

The start of logical page sections are signaled by input lines con­taining
nothing but the following deIlimiter character(s):

<table>
<thead>
<tr>
<th>Line contents</th>
<th>Start of</th>
</tr>
</thead>
<tbody>
<tr>
<td>:::</td>
<td>header</td>
</tr>
<tr>
<td>::</td>
<td>body</td>
</tr>
<tr>
<td>:</td>
<td>footer</td>
</tr>
</tbody>
</table>

Unless optioned otherwise, nl assumes the text being read is in a
single logical page body.

OPTIONS
Command options may appear in any order and may be intermingled
with an optional file name. Only one file may be named. The
options are:

-btype Number the logical page body lines according to type.
Recognized types and their meaning are:

<table>
<thead>
<tr>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>number all lines</td>
</tr>
<tr>
<td>t</td>
<td>number lines with printable text only</td>
</tr>
<tr>
<td>n</td>
<td>no line numbering</td>
</tr>
<tr>
<td>pstring</td>
<td>number only lines that match the regular expression string</td>
</tr>
</tbody>
</table>

Type for logical page body defaults to t.

-htype Number the logical page header according to type (see
    -b). Default type is n.

-ftype Number the logical page footer according to type. (See
    -b). Default type is n.

-p Do not restart numbering at logical page delimiters.

-vstart# Number logical page lines with start# as the initial value.
Default start# is 1.

-incr Number logical page lines with incr as the increment value. Default incr is 1.

-ssep Separate the line number and the corresponding text line with the character sep. Default sep is a tab.

-wwidth Use width number of characters for the line number. Default width is 6.

-nformat Use format as the line numbering format. Recognized values are:

   ln left justified, leading zeroes supressed
   rn right justified, leading zeroes supressed
   rz right justified, leading zeroes kept.

Default format is rn (right justified).

-lnnum Consider num blank lines as one. For example, -l2 results in only the second adjacent blank line being numbered (if the appropriate -ha, -ba, and/or -fa option is set). Default num is 1.

-dxx Change the delimitier characters for the start of a logical page section from the default characters (\:) to two user specified characters. If only one character is entered, the second character remains the default character (:). No space should appear between the -d and the delimiter characters. To enter a backslash, use two backslashes.

EXAMPLE
The command:

    nl -v10 -i10 -d!+ file1

numbers file1 starting at line number 10 with an increment of ten. The logical page delimiters are !+.

SEE ALSO
pr(1).
NAME
nm - print name list of common object file

SYNOPSIS
nm
[ -o ] [ -x ] [ -h ] [ -v ] [ -n ] [ -e ] [ -f ] [ -u ] [ -V ] [ -T ] filenames

DESCRIPTION
The nm command displays the symbol table of each common object file filename. Filename may be a relocatable or absolute common object file; or it may be an archive of relocatable or absolute common object files. For each symbol, the following information is printed:

| Name | The name of the symbol. |
| Value | Its value expressed as an offset or an address depending on its storage class. |
| Class | Its storage class. |
| Type | Its type and derived type. If the symbol is an instance of a structure or of a union, the structure or union tag is given following the type (e.g., struct-tag). If the symbol is an array, the array dimensions are given following the type (e.g., char[n][m]). Note that the object file must have been compiled with the -g option of cc (1) for this information to be output. |
| Size | Its size in bytes, if available. Note that the object file must have been compiled with the -g option of the cc (1) command for this information to be output. |
| Line | The source line number at which it is defined, if available. Note that the object file must have been compiled with the -g option of the cc (1) command for this information to be output. |
| Section | For storage classes static and external, the object file section containing the symbol (e.g., text, data or bss). |

OPTIONS
The output of nm may be controlled using the following options. Options may be used in any order, either singly or in combination, and may appear anywhere in the command line.

-0 Print the value and size of a symbol in octal instead of decimal.
-x Print the value and size of a symbol in hexadecimal instead of decimal.
-h Suppress the output header data.
-v Sort external symbols by value before printing them.
-n Sort external symbols by name before printing them.
-e Print only static and external symbols.
-f Produce full output, including redundant symbols (.text, .data and .bss) normally suppressed.
-u Print only undefined symbols.
-V Display the version of nm command executing on standard error output.
-p produces easily parsed, terse output. Each symbol name is preceeded by its value (blanks if undefined) and one of the letters U (undefined), A (absolute), T (text segment symbol), D (data segment symbol), S (user defined segment symbol), R (register symbol), F (file symbol), or C (common symbol). If the symbol is local (non-external), the type letter appears as a lowercase letter. (This option is applicable to 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.)
-V displays the version of the nm command executing on standard error output. (This option is applicable to 5000/35 and 5000/55 Release 2.00 only.)
-T By default, nm prints the entire name of the symbols listed. Because object files can have symbols names with an arbitrary number of characters, a name that is longer than the width of the column set aside for names overflows its column, forcing every column after the name to be misaligned. The -T option causes nm to truncate every name which would otherwise overflow its column and place an asterisk as the last character in the displayed name to mark it as truncated.

EXAMPLE
This command prints the static and external symbols in file, with external symbols sorted by value:

    nm file -e -v

This command does the same:

    nm -ve file

FILES
/usr/tmp/nm??????

WARNINGS
When all the symbols are printed, they must be printed in the order they appear in the symbol table in order to preserve the scoping information. Therefore, the -v and -n options should be used only in conjunction with the -e option.

SEE ALSO
as(1), cc(1), ld(1), a.out(4), ar(4), and tmpname(3s).

DIAGNOSTICS
NAME
nohup - run a command immune to hangups and quits

SYNOPSIS
nohup command [ arguments ]

DESCRIPTION
Nohup executes command with hangups and quits ignored.
If you log off (hangup) while a command is executing in the background, the command terminates. Using nohup to execute a background command causes the command to continue execution if you log off.

EXAMPLE
It is frequently desirable to apply nohup to pipelines or lists of commands. This can be done only by placing pipelines and command lists in a single file, called a shell procedure. One can then issue:

    nohup sh file

and the nohup applies to everything in file. If the shell procedure file is to be executed often, then the need to type sh can be eliminated by giving file execute permission. Add an ampersand and the contents of file are run in the background with interrupts also ignored (see sh(1)).

    nohup file &

An example of what the contents of file could be is:

    tbl ofile | eqn | nroff > nfile

In the following command format, nohup applies only to command 1.

    nohup command1; command2

The following command format is syntactically incorrect.

    nohup (command1; command2)

If output is not redirected by the user, both standard output and standard error are sent to nohup.out. If nohup.out is not writable in the current directory, output is redirected to $HOME/nohup.out. Be careful of where standard error is redirected. On the 5000/20/30/40/50, for example, the following command may put error messages on tape making it unreadable.

    nohup cpio -o <list >/dev/rmt/0yy&

This command puts the error messages into file errors.

    nohup cpio -o <list >/dev/rmt/0yy 2>errors&

SEE ALSO
chmod(1), nice(1), sh(1), signal(2).
NAME
nroff, troff - format or typeset text

SYNOPSIS
nroff [ options ] [ files ]
troff [ options ] [ files ]

DESCRIPTION
Not on 7000 Series Systems.

Nroff formats text contained in files (standard input by default) for printing on typewriter-like devices and line printers; similarly, troff formats text for a Wang Laboratories, Inc., C/A/T phototypesetter.

OPTIONS
An argument consisting of a minus (-) is taken to be a file name corresponding to the standard input. The options, which may appear in any order, but must appear before the files, are:

-o list Print only pages whose page numbers appear in the list of numbers and ranges, separated by commas. A range N-M means pages N through M; an initial -N means from the beginning to page N; and a final N- means from N to the end. (See RESTRICTIONS below.)

-n N Number first generated page N.

-s N Stop every N pages. Nroff halts after every N pages (default N=1) to allow paper loading or changing, and resumes upon receipt of a line-feed or new-line (new-lines do not work in pipelines, e.g., with mm(1)). This option does not work if the output of nroff is piped through col(1). Troff stops the phototypesetter every N pages, produces a trailer to allow changing cassettes, and resumes when the typesetter start button is pressed. When nroff (troff) halts between pages, an ASCII BEL (in troff, the message page stop) is sent to the terminal.

-r a N Set register a (which must have a one-character name) to N.

-i Read standard input after files are exhausted.

-q Invoke the simultaneous input-output mode of the .rd request.

-z Print only messages generated by .tm (terminal message) requests.

-m name Prepend to the input files the non-compacted (ASCII text) macro file /usr/lib/tmac/tmac.name .

-c name Prepend to the input files the compacted macro files /usr/lib/macros/cmp.[nt].[dt].name and /usr/lib/macros/ucmp.[nt].name .

-k name Compact the macros used in this invocation of nroff / troff, placing the output in files [dt].name in the current directory.

Nroff only:

-T name Prepare output for specified terminal. Known names are 37 for the (default) TELETYPE® Model 37 terminal, tn300
for the GE TermiNet 300 (or any terminal without half-line
capability), 300s for the DASI 300s, 300 for the DASI 300,
450 for the DASI 450, lp for a (generic) ASCII line
printer, 382 for the DTC-382, 4000A for the Trendata
4000A, 832 for the Anderson Jacobson 832, X for a (gen-
eric) EBCDIC printer, 2631 for the Hewlett Packard 2631
line printer, 6411 for the NCR 6411 printer, 6416 for the
NCR 6416 printer, and 6455 for the NCR 6455 printer.

-e Produce equally-spaced words in adjusted lines, using the
full resolution of the particular terminal.
-h Use output tabs during horizontal spacing to speed output
and reduce output character count. Tab settings are
assumed to be every 8 nominal character widths.
-un Set the emboldening factor (number of character over-
strikes) for the third font position (bold) to n, or to zero
if n is missing.

Troff only:
-t Direct output to the standard output instead of the photo-
typesetter.
-f Refrain from feeding out paper and stopping photo-
typesetter at the end of the run.
-w Wait until phototypesetter is available, if it is currently
busy.
-b Report whether the phototypesetter is busy or available.
No text processing is done.
-a Send a printable ASCII approximation of the results to the
standard output.
-pN Print all characters in point size N while retaining all
prescribed spacings and motions, to reduce photo-
typesetter elapsed time.
-Tcat Use font-width tables for Wang CAT phototypesetter.
This device is both the default and the only choice.

FILES
/usr/lib/suftab suffix hyphenation tables
/tmp/ta$# temporary file
/usr/lib/tmac/* tmac... standard macro files and pointers
/usr/lib/macros/* standard macro files
/usr/lib/term/* terminal driving tables for nroff
/usr/lib/font/* font width tables for troff

SEE ALSO
col(1), eqn(1), greek(1), mm(1), mmt(1), tbl(1), troff(1),
mm(5).

RESTRICTIONS
Nroff / troff internally supports Eastern Standard Time; as a
result, depending on the time of the year and on your local time
zone, the date that nroff / troff generates may be off by one day
from your idea of what the date is.

When nroff / troff is used with the -olist option inside a pipeline
(e.g., with eqn(1), or tbl(1)), it may cause a harmless broken
pipe diagnostic if the last page of the document is not specified in
list.
NAME
   od - octal dump
SYNOPSIS
   od [ -bcdosx ] [ file ] [ [ + ]offset[ . ] [ b ] ]
DESCRIPTION
   Od dumps file in one or more formats as selected by the first argument. If no options are specified, -o is assumed.

   The file argument specifies which file is to be dumped. If no file argument is specified, the standard input is used.

   The offset argument specifies the offset (in octal bytes) in the file where dumping is to commence. This argument is normally interpreted as octal bytes. If . is appended, the offset is interpreted in decimal. If b is appended, the offset is interpreted in blocks of 512 bytes. If the file argument is omitted, the offset argument must be preceded by +.

   A 'w' will appear instead of the file offset, as long as the line is identical to the previous line (7000 Series only).

   Dumping continues until end-of-file.
OPTIONS
   -b   Interpret bytes in octal.
   -c   Interpret bytes in ASCII. Certain non-graphic characters appear as C escapes: null=\0, backspace=\b, form-feed=\f, new-line=\n, return=\r, tab=\t; others appear as 3-digit octal numbers.
   -d   Interpret words in unsigned decimal.
   -o   Interpret words in octal.
   -s   Interpret 16-bit words in signed decimal.
   -x   Interpret words in hex.
SEE ALSO
   dump(1).
RESTRICTIONS
   Offset argument can not exceed the value $2^{31}$ minus 1.
[This page left blank.]
NAME

pack, pcat, unpack - compress and expand files

SYNOPSIS

pack [ - ] [ -f ] name ...

pcat name ...

unpack name ...

DESCRIPTION

Pack attempts to compress the specified files. Packed files can be restored to their original form using unpack(1) or pcat(1).

Wherever possible, each input file name is replaced by a packed file name.z with the same access modes, access and modified dates, and owner as those of name. If pack is successful, name is removed. The -f option forces packing of name. This is useful for causing an entire directory to be packed even if some files do not benefit.

If the - argument is used, an internal flag is set which causes the number of times each byte is used, its relative frequency, and the code for the byte to be output on the standard output. Additional occurrences of - in place of name set and reset the internal flag.

Pack uses Huffman (minimum redundancy) codes on a byte-by-byte basis. The amount of compression obtained depends on the size of the input file and the character frequency distribution. Because a decoding tree forms the first part of each .z file, it is usually not worthwhile to pack files smaller than three blocks, unless the character frequency distribution is very skewed, which may occur with printer plots or pictures. Typically, pack reduces text files to 60-75% of their original size. Load modules, which use a larger character set and have a more uniform distribution of characters, show little compression, the packed versions being about 90% of the original size.

Pack returns the number of files that it failed to compress.

No packing occurs if:

- the file appears to be already packed;
- the file name has more than 12 characters;
- the file has links;
- the file is a directory;
- the file cannot be opened;
- no disk storage blocks are saved by packing;
- a file called name.z already exists;
- the .z file cannot be created;
- an I/O error occurred during processing.

The last segment of the file name must contain no more than 12 characters to allow space for the appended .z extension. Directories cannot be compressed.

Pcat does for packed files what cat(1) does for ordinary files, except that pcat can not be used as a filter. The specified files are unpacked and written to the standard output. Thus to view a packed file named name.z use:
pcat name.z

or just:

pcat name

To make an unpacked copy, say `nnn`, of a packed file named `name.z` (without destroying `name.z`) use the command:

```
pcat name >nnn
```

`Pcat` returns the number of files it was unable to unpack. Failure may occur if:

- the file name (exclusive of the .z) has more than 12 characters;
- the file cannot be opened;
- the file does not appear to be the output of `pack`.

`Unpack` expands files created by `pack`. For each file `name` specified in the command, a search is made for a file called `name.z` (or just `name`, if `name` ends in .z). If this file appears to be a packed file, it is replaced by its expanded version. The new file has the .z suffix stripped from its name, and has the same access modes, access and modification dates, and owner as those of the packed file.

`Unpack` returns a value that is the number of files it was unable to unpack. Failure may occur for the same reasons that it may in `pcat`, as well as for the following:

- a file with the unpacked name already exists;
- if the unpacked file cannot be created.

SEE ALSO

cat(1).
NAME
packsf, unpacksf - compress and uncompress sparse file

SYNOPSIS
packsf <input_file> compressed_file
unpacksf <compressed_file> original_file

DESCRIPTION
5000/20, 5000/30, 5000/40, and 5000/50 only.

*packsf compresses a sparse file, a file that has a large ratio of zeros to nonzero data, into a formatted file that takes less space. The compressed file can be used to recreate the original file.

If the original file is mostly nonzero, other utilities provide better compression.

The compressed file is an array of variable length records. The format of the records is:

S 2 /* Maximum size is 1024 */
struct record {
   int r_faddr; /* File address */
   int r_len; /* Length of the data */
   unsigned char r_buffer[1016];
};

Each record represents a region of nonzero data in the original file.

*r_faddr* holds the file offset, from 0, of the nonzero region and
* r_len gives the length of the region.

*r_buffer* holds the data.

If two nonzero regions are separated by less than 9 zeros the zeros are not compressed.

*unpacksf* reverses the compression and restores the original file. *unpacks* reads records from the compressed file and executes them as commands of the form:

SEEK TO r.r_faddr AND WRITE r.r_len BYTES FROM r.r_buffer

Preceeding the array of data records in the compressed file there is a magic number. The presence of the magic number permits *packsf* to reject requests to pack files that have already been packed by *packsf*.

OPTIONS
packsf Options
-l supresses magic number checking on input files.
-o supresses magic number prefixes on output files.
-0 supresses magic number processing on both input and output files.
unpacksf  Options

-i  suppresses magic number checking on input files.

SEE ALSO
pack(1).

DIAGNOSTICS
  *unpack: unexpected EOF*  Unpacksf was performed on a file that
   was not previously packed.
NAME
passwd - change login password

SYNOPSIS
passwd [ name ]

DESCRIPTION
The passwd command changes or installs a password associated with
the login name. Ordinary users may change only the pass-
word which corresponds to their login name.

Passwd prompts ordinary users for their old password, if any. It
then prompts for the new password twice. The first time the new
password is entered passwd checks to see if the old password has aged sufficiently. If aging is insufficient the new password is
rejected and passwd terminates; see passwd(4).

Assuming aging is sufficient, a check is made to insure that the
new password meets construction requirements. When the new
password is entered a second time the two copies of the new pass-
word are compared. If the two copies are not identical the cycle of
prompting for the new password is repeated for at most two more
times.

Passwords must be constructed to meet the following requirements:

Each password must have at least six characters. Only the
first eight characters are significant.

Each password must contain at least two alphabetic characters
and at least one numeric or special character. In this case,
alphabetic means upper and lower case letters.

Each password must differ from the user's login name and any
reverse or circular shift of that login name. For comparison
purposes, an upper case letter and its corresponding lower
case letter are equivalent.

New passwords must differ from the old by at least three char-
acters. For comparison purposes, an upper case letter and its
corresponding lower case letter are equivalent.

One whose effective user ID is zero is called a superuser; see
id(1), and su(1). Superusers may change any password; hence,
passwd does not prompt superusers for the old password.
Superusers are not forced to comply with password aging and pass-
word construction requirements. A superuser can create a null
password by entering a carriage return in response to the prompt
for a new password.

FILES
/etc/passwd

SEE ALSO
login(1), id(1), su(1), crypt(3C), passwd(4), dp passwd(1M).
NAME
paste - merge same lines of several files or subsequent lines of one file

SYNOPSIS
paste file1 file2 ...
paste -dlist file1 file2 ...
paste -s [-dlist] file1 file2 ...

DESCRIPTION
In the first two command forms, paste concatenates corresponding lines of the given input files file1, file2, etc. It treats each file as a column or columns of a table and pastes them together horizontally (parallel merging). Paste is the horizontal counterpart of cat(1) which concatenates vertically, i.e., one file after the other.

In the last command form above, paste combines subsequent lines of the input file (serial merging).

In all cases, lines are connected with the tab character, or with characters from an optionally specified list. Output is to the standard output, so paste can be used as the start of a pipe, or as a filter, if - is used in place of a file name.

OPTIONS
-dlist
Replace the tab character by one or more alternate characters specified in list. The list is used circularly, i.e. when exhausted, it is reused. In parallel merging (i.e. no -s option), the lines from the last file are always terminated with a new-line character, not from the list. The list may contain the special escape sequences:

\n new-line
\t tab
\\ backslash
\0 empty string, not a null character

Quoting may be necessary, if characters have special meaning to the shell (e.g. to get one backslash, use -d "\\\"").

Without this option, the new-line characters of each but the last file (or last line in case of the -s option) are replaced by a tab character.

-s Merge subsequent lines rather than one from each input file.
Use tab for concatenation, unless the -d option is used. The very last character of the file is a new-line.

-m May be used in place of any file name, to read a line from the standard input. Paste does not prompt.

NOTE
pr -t -m ... works similarly, but creates extra blanks, tabs and
new-lines for a nice page layout.

EXAMPLES

```
ls | paste -d " " list directory in one column
ls | paste - - - list directory in four columns
paste -s -d "\t\n" file combine pairs of lines into lines
```

SEE ALSO
grep(1), cut(1), pr(1).

DIAGNOSTICS

- `line too long` Output lines are restricted to 511 characters.
- `too many files` Except for `-s` option, no more than 12 input files may be specified (64 for the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00).
NAME
pcdsk - PC-DOS to UNIX file transfer

SYNOPSIS
pcdsk

DESCRIPTION
5000/20, 5000/30, 5000/35, 5000/40, 5000/50, and 5000/55 Release 2.00.00 only.

Pcdsk transfers files between a PC-DOS (or MS-DOS) floppy disk and the UNIX file system and provides directory listing functions.

Pcdsk handles 5.25 inch floppy disks formatted for PC-DOS version 2.1 which are single or double sided, have eight or nine sectors per track, and are formatted 48 tracks per inch.

The PC-DOS floppy disk must be installed in the top, left, or only floppy disk drive. The superuser must make a special file (see wd(7)) using mknod(1M) before pcdsk can be executed.

Note: The superuser must invoke pcdsk from the command line without any parameters before pcdsk can be used.

To specify a pathname for a pcdsk copy operation, use a UNIX-style pathname although either / or \ may be used to separate pathname parts. The metacharacters * and ? are recognized to have their UNIX meaning in pathnames. No escape character such as \ is recognized.

COMMANDS
cat
List a UNIX file. This command is identical to the UNIX cat(1) command; all cat options are accepted. Pcdsk passes this command to the shell for execution.

dir
List the directory of the PC-DOS floppy disk. The directory is displayed in the following format:

```
# filename other_info
```

where # is the file size in bytes, filename is the name of the file, and other_info indicates if the file is a hidden file, system file, or a directory.

exit
Terminate pcdsk.

help
Display the pcdsk commands and command descriptions.

ls
List a UNIX directory. This command is identical to the UNIX ls(1) command; all ls options are accepted. Pcdsk passes this command to the shell for execution.

mtu
Copy files from the PC-DOS floppy disk to the UNIX file system. After this command is entered, pcdsk prompts for the PC-DOS source pathname and the UNIX destination pathname. Metacharacters (wildcards) may be entered in the PC-DOS pathname,
but may not be entered in the UNIX destination pathname.

**sh** Invoke the UNIX shell. To exit the shell and return to `pcdsk`, enter a control-d.

**utm**

Copy files from the UNIX file system to the PC-DOS floppy disk. After this command is entered, `pcdsk` prompts for the UNIX source pathname and the PC-DOS destination pathname. Meta-characters (wildcards) may be entered in the UNIX pathname, but may not be entered in the PC-DOS destination pathname.

**!command**

Escape to the shell and execute `command`.

**control-d**

Terminate `pcdsk`. This is the same as the `exit` command.

**EXAMPLES**

If the PC-DOS source pathname and the UNIX destination pathname in a copy operation are specified as:

- From pcdos files: /PCSUBDIR/PCFILE
- To UNIX files: usubdir/ufile

the PCFILE file is copied to the ufile file. The PC-DOS pathname specification is from the PC-DOS root directory. The UNIX pathname specification is from the current working directory.

If the PC-DOS source pathname and the UNIX destination pathname in a copy operation are specified as:

- From pcdos files: PCSUBDIR/PCFILE
- To UNIX files: /usr/acct/thompson/usubdir/ufile

the PCFILE file is copied to the ufile file. The PC-DOS pathname specification is from the PC-DOS root directory even though a `/` is not specified. The UNIX pathname specification is from the root directory because `/` is the first character specified in the pathname.

If the PC-DOS source pathname and the UNIX destination pathname in a copy operation are specified as:

- From pcdos files: /PCSUB*/PCFILE.*
- To UNIX files: usubdir

then starting at the PC-DOS root directory, all files which are named PCFILE. with any two character file name extension in any directory which has a name starting with PCSUB are copied to the usubdir directory which is a subdirectory of the current working directory.

**SEE ALSO**

mknod(1M), wd(7).

**RESTRICTIONS**

When a file is copied to a PC-DOS floppy disk, the date and time are not put in the directory entry.

The [ and ] metacharacters do not work.
Pcdsk does not create a PC-DOS floppy disk directory nor does it format a floppy disk.

Pcdsk is compatible only with the Mass Storage Controller, the SCSI Mass Storage Controller, or the 5.25" Disk Controller subsystems. (This restriction applies to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.)
NAME
pg - file perusal filter for screen terminals

SYNOPSIS
pg [-number] [-p string ] [-cefns] [+linenumber] [/+pattern/]
[files ...]

DESCRIPTION
The pg command is a filter which allows the examination of files one
screenful at a time on a terminal. The file name - or no file name
indicates that pg should read from the standard input. Each
screenful is followed by a prompt. If the user types a carriage
return, another page is displayed; other possibilities are
enumerated below.

This command is different from previous paginators in that it allows
you to back up and review something that has already passed. The
method for doing this is explained below.

OPTIONS
The command line options are:

- number
  The number of lines in the window that pg is to use instead of
  the default number. On a terminal containing 24 lines, the
default window size is 23.

- p string
  Use string as the prompt. If the prompt string contains a "%d",
  the first occurrence of "%d" in the prompt is replaced by the
current page number when the prompt is issued. The default
prompt string is " : ".

- c Clear the screen and home the cursor before displaying each
  page. This option is ignored if clear_screen is not defined for
  this terminal type in the terminfo (4) data base.

- e Do not pause at the end of each file.

- f Do not split lines. Normally, pg splits lines longer than the
  screen width, but some sequences of characters in the text
  being displayed (e.g., escape sequences for underlining) gen-
  erate undesirable results.

- n Cause an automatic end of command as soon as a command letter
  is entered. Normally, commands must be terminated by a new-
  line.

- r does not permit shell execution (see the /command command).

- s Print all messages and prompts in standout mode (usually
  inverse video).

+ linenumber
  Start up at linenumber .

+/ pattern /
  Start up at the first line containing the regular expression pat-
  tern.

COMMANDS
At any time the prompt is displayed, the user may enter one of the
pg commands. When output is being sent to the terminal, the user
can press the quit key (normally control-\) or the interrupt
(break) key to cause pg to stop sending output and display the prompt. Some output is lost when this is done because characters in the terminal output queue are flushed when the quit signal occurs.

The commands that may be entered when pg pauses can be divided into three categories: those causing further perusal, those that search, and those that modify the perusal environment.

Further Perusal Commands
Commands which cause further perusal normally take a preceding address, an optionally signed number indicating the point from which further text should be displayed. This address is interpreted in either pages or lines depending on the command. A signed address specifies a point relative to the current page or line, and an unsigned address specifies an address relative to the beginning of the file. Each command has a default address that is used if none is provided. The perusal commands and their defaults are:

(+1)<newline> or <blank>
Display one page. The address is specified in pages.

(+1) l
Scroll the screen, forward or backward, the number of lines specified if the address is signed. Print a screenful beginning at the specified line if the address is unsigned.

(+1) d or ^D
Scroll half a screen forward or backward.

The following perusal commands take no address.

Redisplay the current page of text.
$ Display the last windowful in the file. Use with caution when the input is a pipe.

Search Commands
The following commands are available for searching for text patterns in the text. The regular expressions described in ed (1) are available. They must always be terminated by a <newline> even if the -n option is specified.

i/pattern/
Search forward for the i th (default i =1) occurrence of pattern. Searching begins immediately after the current page and continues to the end of the current file without wrap-around.

i^pattern^t
Search backwards for the i th (default i =1) occurrence of pattern. Searching begins immediately before the current page and continues to the beginning of the current file without wrap-around. The ^ notation is useful for ADDS 100 terminals which do not properly handle the ?.

After searching, pg normally displays the line found at the top of the screen. This can be modified by appending m or b to the search command to leave the line found in the middle or at the
bottom of the window from now on. The suffix t can be used to restore the original situation.

**Modify Environment Commands**

Modify the environment of perusal with the following commands:

- **Skips**
  - **if** Skip i screenfuls and display a screenful of lines.
  - **in** Begin perusing the i th next file in the command line. The i is an unsigned number with a default value of 1.
  - **ip** Begin perusing the i th previous file in the command line. The i is an unsigned number with a default value of 1.
  - **iw** Display another window of text. If i is present, set the window size to i.
  - **iz** Same as pressing a newline/return except that i, if present, becomes the new window size.

- **Save**
  - **s filename** Save the input in the named file. Only the current file being perused is saved. The white space between the s and filename is optional. This command must always be terminated by a newl

- **Help**
  - **h** Help by displaying an abbreviated summary of available commands.

- **Quit**
  - **q or Q** Quit pg.

- **Command**
  - **command** Command is passed to the shell whose name is taken from the SHELL environment variable. If this is not available, the default shell is used. This command must always be terminated by a newline even if the -n option is specified.

**EXAMPLE**

A sample usage of pg in reading system news is:

```
news | pg -p "(Page %d):"
```

**NOTES**

While waiting for terminal input, pg responds to BREAK, DEL, and by terminating execution. Between prompts, however, these signals interrupt pg's current task and place the user in prompt mode. These should be used with caution when input is being read from a pipe because an interrupt is likely to terminate the other commands in the pipeline.

Users of more (1) will find that the z and f commands are available, and that the terminal /, , or ? may be omitted from the searching commands.

In order to determine terminal attributes, pg scans the terminfo (4) data base for the terminal type specified by the environment variable TERM. If TERM is not defined, the terminal type dumb is assumed.

If the standard output is not a terminal, then pg acts just like cat (1), except that a header is printed before each file (if there is more than one).
FILES
/usr/lib/terminfo/*  terminal information data base
/tmp/pg*  temporary file if pipe input

SEE ALSO
  crypt(1), ed(1), grep(1), more(1), pr(1), terminfo(4).

RESTRICTIONS
  If terminal tabs are not set every eight positions, undesirable results may occur.
  When using pg as a filter with another command that changes the terminal I/O options (e.g., crypt(1)), terminal settings may not be restored correctly.
NAME
pr - print files

SYNOPSIS
pr [ options ] [ files ]

DESCRIPTION
Pr prints the named files separating the listing into pages. Each page is headed by the page number, a date and time, and the name of the file, unless options specify otherwise. Pr is typically used to paginate files for output to a printer. Pr prints the named files on the standard output. If the standard output is associated with a terminal, error messages are withheld until pr has completed printing. If file is -, or if no files are specified, the standard input is assumed.

Columns are of equal width separated by at least one space; lines which do not fit are truncated, unless the -s option is used.

The width of an output line is 72 character positions for equal width multi-column output unless the -w option is used.

The length of an output page is 66 lines unless the -l option is used.

Pr advances to a new page using a sequence of line-feeds unless the -f option is used.

OPTIONS
The options may appear singly or be combined in any order.

+k Begin printing with page k; the default is 1.
-k Produce k-column output; the default is 1. The options -e and -i are assumed for multi-column output.
-a Print multi-column output across the page. The -k must be specified for more than one column.
-d Double-space the output.
-eck
Expand input tabs to character positions k+1, 2*k+1, 3*k+1, etc. If k is 0 or is omitted, pr assumes tab settings at every eighth position. Pr expands tab characters in the input into the appropriate number of spaces. The c (any non-digit character) designates the input tab character. If c is not specified, the tab character is used.

-f Use the form-feed character for new pages; the default is to use a sequence of line feeds. Pause before beginning the first page if the standard output is associated with a terminal.
-h Use the next argument as the header to be printed instead of the file name.

-ick
Replace white space in output wherever possible by inserting tabs to character positions k+1, 2*k+1, 3*k+1, etc. If k is 0 or is omitted, pr assumes tab settings at every eighth position. The c (any non-digit character) designates the output tab character. If c is not specified, the tab character is used.

-lk Set the length of a page to k lines; the default is 66.
-m Merge and print all files simultaneously, one per column. This option overrides the -k and -a options.

-nc k
  Provide k-digit line numbering; the default for k is 5. The number occupies the first k+1 character positions of each column of normal output or each line of -m output. The c (any non-digit character) separates the line number from whatever follows. If c is not specified, a tab character is used.

-ok
  Offset each line by k character positions. The number of character positions per line is the sum of the width and offset.

-p Pause before beginning each page if the output is directed to a terminal. Pr rings the bell at the terminal and waits for a carriage return.

-r Print no diagnostic reports on failure to open files.

-sc Separate columns by the single character c instead of by the appropriate number of spaces. The default for c is a tab. Do not truncate lines.

-t Print neither the five line identifying header nor the five line trailer normally supplied for each page. Quit printing after the last line of each file without spacing to the end of the page.

-wk
  Set the width of a line to k character positions; the default is 72 for equal-width multi-column output, no limit otherwise. The -k should be specified as something other than 1. If used with the -m option, the specified width must be greater than the number of files to be merged; for example, merging three files requires a minimum width of 4. If the -w option causes truncation, the -s option may not work.

EXAMPLES
  Print file1 and file2 as a double-spaced, three-column listing headed by file list and pipe the listing to a printer:

  pr -3dh "file list" file1 file2 | print

  Write file1 on file2, expanding tabs to columns 10, 19, 28, 37, . . .

  pr -e9 -t <file1 >file2

FILES
  /dev/tty** to suspend messages

SEE ALSO
  cat(1), pg(1).
NAME
print, lpr - line printer spooler

SYNOPSIS
print [ options ] files
lpr [ options ] files

DESCRIPTION
This command is applicable to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only. The print or lpr command spools the named file(s) for printing. The command creates an entry in the spool queue for each file listed on the input command line. The command also places a copy of the file to be printed in the spooler directory under a unique file name. After completion of the copy process, the command informs the shell to start the despooler which prints the file. The file created by the spooler for printing is deleted from the system upon completion of the print sequence.

The command does not control pagination, headers, or any other part of the output. All control characters and formatting data must be placed in the file prior to invoking print or lpr.

If no files are named, the command reads from standard input.

OPTIONS
The options may appear in any order.

-b Do not print a banner page consisting of user name, date, filename and the contents of /etc/lpmsg. The default is to print the banner.

-cp nnn
Print nnn copies of this file. The range for nnn is 1 through 999. The default is 1.

-fm xxxxx
Use xxxxx as the forms name; the forms name is any 1-5 alphanumeric characters. The forms name specified here must agree with the forms name of the form specified as installed in the printer for printing to occur. See spool(1). The default is the standard print forms name 00000.

-ln nn
Print nn number of lines per inch. Standard values are 6 and 8 lines per inch. The default is 6 lines per inch. This data is used by the printer operator to setup for special forms control and is not automatically used by the spooler.

-pr nn
Set nn as the priority for this print request. The range for nn is 0 through 15.

-pt lpnn
Redirect print output to the specified line printer. This option overrides the default terminal/printer routing. The range for nn is 00 through the maximum number of printers allowed on the system.

EXAMPLES
To paginate and print report on the printer designated as lp01 without a banner, enter
To print a previously formatted file of paychecks on a form designated as paych, enter

```
pr report | print -b -pt lp01
```

If the payck form is not designated as installed on the printer, the payfile is held in a wait state. After the payck form is designated as installed by using `spool(1)`, payfile is printed.

**MAPPING**

*Print* provides mapping for up to ten device classes. Device class 0 maps tabs to appropriate spaces, etc. Device class 1 maps one to one. Device classes 2 through 9 may be defined by the user. The user must be the superuser, an analyst, or C programmer. See `/usr/spool/lpd/oemdir/README` for information on defining device classes 2 through 9.

**FILES**

```
/usr/spool/lpd* spool area
/usr/spool/lpd/lpd despooler
/bin/print spooler
/bin/lpr spooler
/bin/spool spool queue manager
/usr/spool/lpd/spooldev spool device table manager
/usr/spool/lpd/??spldev spool device table
/usr/spool/lpd/oemdir user defined printer mapping
/usr/spool/lpd/??splque spool queue
/usr/spool/lpd/sf* spooled files
/etc/lpmsg line printer message file
```

**SEE ALSO**

*spooldev(1M)*, *spool(1)*.

**RESTRICTION**

The *print* command queues a file and informs the shell to start the despooler. If no despooler is active and if the shell is terminated before the despooler is started, the files queued remain on the queue in a wait state. The files are printed during the next run of the despooler (i.e. the next print or spool -start command).

Files which contain formatting functions or various font styles may need to be sent through a preprocessor before being sent to the line printer spooler.
NAME
prof - display profile data

SYNOPSIS
prof [-tcan] [-ox] [-g] [-z] [-h] [-s] [-m mdata] [prog]

DESCRIPTION
Prof interprets a profile file produced by the monitor(3C) function. The symbol table in the object file prog (a.out by default) is read and correlated with a profile file (mon.out by default). For each external text symbol the percentage of time spent executing between the address of that symbol and the address of the next is printed, together with the number of times that function was called and the average number of milliseconds per call.

A program creates a profile file if it has been loaded with the -p option of cc(1). This option to the cc command arranges for calls to monitor(3C) at the beginning and end of execution. It is the call to monitor at the end of execution that causes a profile file to be written. The number of calls to a function is tallied if the -p option was used when the file containing the function was compiled.

The name of the file created by a profiled program is controlled by the environment variable PROFDIR. If PROFDIR does not exist, mon.out is produced in the directory current when the program terminates. If PROFDIR = string, "string/pid.progname" is produced, where progname consists of argv[0] with any path prefix removed, and pid is the program process id. If PROFDIR = nothing, no profiling output is produced.

A single function may be split into subfunctions for profiling by means of the MARK macro (see prof(5)).

OPTIONS
The mutually exclusive options t, c, a, and n determine the type of sorting of the output lines:
-t Sort by decreasing percentage of total time (default).
-c Sort by decreasing number of calls.
-a Sort by increasing symbol address.
-n Sort lexically by symbol name.

The mutually exclusive options o and x specify the printing of the address of each symbol monitored:
-o Print each symbol address in octal along with the symbol name.
-x Print each symbol address in hexadecimal along with the symbol name.

The following options may be used in any combination:
-g Include non-global symbols (static functions).
-z Include all symbols in the profile range (see monitor(3C)), even if associated with zero number of calls and zero time.
-h Suppress the heading normally printed on the report. This is useful if the report is to be processed further.
-s Print a summary of several of the monitoring parameters and statistics on the standard error output.

-m mdata
   Use file mdata instead of mon.out as the input profile file.

FILES
   mon.out for profile
   a.out   for namelist

SEE ALSO
   cc(1), exit(2), profil(2), monitor(3C), prof(5).

WARNING
   The times reported in successive identical runs may show variances of 20% or more because of varying cache-hit ratios due to sharing of the cache with other processes. Even if a program seems to be the only one using the machine, hidden background or asynchronous processes may affect the data. In rare cases, the clock ticks initiating recording of the program counter may be in rhythm with loops in a program, grossly distorting measurements. Call counts are always recorded precisely, however.

RESTRICTIONS
   Only programs that call exit(2) or return from main cause a profile file to be produced unless a final call to monitor is explicitly coded.

   The use of the -p option of cc(1) to invoke profiling imposes a limit of 600 functions that may have call counters established during program execution. For more counters you must call monitor(3C) directly. If this limit is exceeded, other data is overwritten and the mon.out file is corrupted. The number of call counters used is reported automatically by the prof command whenever the number exceeds 5/6 of the maximum.
NAME
  prs - print an SCCS file

SYNOPSIS
   prs [-d[dataspec]] [-r[SID]] [-e] [-l] [-c[date-time]] [-a] files

DESCRIPTION
   Prs prints, on the standard output, parts or all of an SCCS file
   (see sccsfile(4)) in a user-supplied format. If a directory is
   named, prs behaves as though each file in the directory were
   specified as a named file, except that non-SCCS files (last com-
   ponent of the path name does not begin with s.), and unreadable
   files are silently ignored. If a name of - is given, the standard
   input is read; each line of the standard input is taken to be the
   name of an SCCS file or directory to be processed; non-SCCS files
   and unreadable files are silently ignored.

   Arguments to prs, which may appear in any order, consist of
   options and file names.

OPTIONS
   All the described options apply independently to each named file:
     -d[dataspec]
       Used to specify the output data specification. The
       dataspec is a string consisting of SCCS file
data keywords (see DATA KEYWORDS) inter-
       spersed with optional user supplied text.
     -r[SID]
       Used to specify the SCCS ID entification (SID)
       string of a delta for which information is desired.
       If no SID is specified, the SID of the most
       recently created delta is assumed.
     -e
       Requests information for all deltas created earlier
       than and including the delta designated via the -r
       option or the date given by the -c option.
     -l
       Requests information for all deltas created later
       than and including the delta designated via the -r
       option or the date given by the -c option.
     -c[date-time]
       Cutoff date-time, in the form:
         YY[MM[DD[HH[MM[SS]]]]]
       Units omitted from the date-time default to their
       maximum possible values; that is, -c7502 is
       equivalent to -c750228235959. Any number of
       non-numeric characters may separate the various
       two-digit pieces of the cutoff date in the form:
     -a
       Requests printing of information for both
       removed, i.e., delta type = R, (see rmdel(1))
and existing, i.e., delta type = D, deltas. If the
-a option is not specified, information for exist-
ing deltas only is provided.

DATA KEYWORDS
Data keywords specify which parts of an SCCS file are to be
retrieved and output. All parts of an SCCS file (see sccsfile(4))
have an associated data keyword. There is no limit on the number
of times a data keyword may appear in a dataspec.

The information printed by prs consists of: (1) the user-supplied
text and (2) appropriate values (extracted from the SCCS file)
substituted for the recognized data keywords in the order of
appearance in the dataspec. The format of a data keyword value is
either Simple (S), in which keyword substitution is direct, or
Multi-line (M), in which keyword substitution is followed by a car-
riage return.

User-supplied text is any text other than recognized data key-
words. A tab is specified by \t and carriage return/new-line is
specified by \n. The default data keywords are:

":Dt:\t:DL:\nMRs:\n:MR:COMMENTS:\n:C:"

### TABLE 1. SCCS Files Data Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Data Item</th>
<th>File Section</th>
<th>Value Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>:Dt:</td>
<td>Delta information</td>
<td>Delta Table</td>
<td>See below*</td>
</tr>
<tr>
<td>:DL:</td>
<td>Delta line statistics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>:Li:</td>
<td>Lines inserted by Delta</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:Ld:</td>
<td>Lines deleted by Delta</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:Lu:</td>
<td>Lines unchanged by Delta</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:DT:</td>
<td>Delta type</td>
<td></td>
<td>D or R</td>
</tr>
<tr>
<td>:R:</td>
<td>Release number</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:L:</td>
<td>Level number</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:S:</td>
<td>Sequence number</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:D:</td>
<td>Date Delta created</td>
<td></td>
<td>:Dy: :Dm: :Dd:</td>
</tr>
<tr>
<td>:Dy:</td>
<td>Year Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:Dm:</td>
<td>Month Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:Dd:</td>
<td>Day Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:T:</td>
<td>Time Delta created</td>
<td></td>
<td>:Th: :Tm: :Ts:</td>
</tr>
<tr>
<td>:Th:</td>
<td>Hour Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:Tm:</td>
<td>Minutes Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:Ts:</td>
<td>Seconds Delta created</td>
<td></td>
<td>nn</td>
</tr>
<tr>
<td>:P:</td>
<td>Programmer who created Delta</td>
<td></td>
<td>logname</td>
</tr>
<tr>
<td>:DS:</td>
<td>Delta sequence number</td>
<td></td>
<td>nnnnn</td>
</tr>
<tr>
<td>:DP:</td>
<td>Predecessor Delta seq-no.</td>
<td></td>
<td>nnnn</td>
</tr>
<tr>
<td>:DI:</td>
<td>Seq-no. of deltas incl., excl., ignored</td>
<td></td>
<td>:Dn: :Dx: :Dg:</td>
</tr>
<tr>
<td>:Dn:</td>
<td>Deltas included (seq #)</td>
<td></td>
<td>:DS: :DS: :DS:</td>
</tr>
<tr>
<td>:Dx:</td>
<td>Deltas excluded (seq #)</td>
<td></td>
<td>:DS: :DS: :DS:</td>
</tr>
<tr>
<td>:Dg:</td>
<td>Deltas ignored (seq #)</td>
<td></td>
<td>:DS: :DS: :DS:</td>
</tr>
<tr>
<td>:MR:</td>
<td>MR numbers for delta</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:C:</td>
<td>Comments for delta</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:UN:</td>
<td>User names</td>
<td>User Names</td>
<td>text</td>
</tr>
<tr>
<td>:FL:</td>
<td>Flag list</td>
<td>Flags</td>
<td>text</td>
</tr>
<tr>
<td>:Y:</td>
<td>Module type flag</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:MF:</td>
<td>MR validation flag</td>
<td></td>
<td>yes or no</td>
</tr>
<tr>
<td>:MP:</td>
<td>MR validation pgm name</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:KF:</td>
<td>Keyword error/warning flag</td>
<td></td>
<td>yes or no</td>
</tr>
<tr>
<td>:KV:</td>
<td>Keyword validation string</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:BF:</td>
<td>Branch flag</td>
<td></td>
<td>yes or no</td>
</tr>
<tr>
<td>:J:</td>
<td>Joint edit flag</td>
<td></td>
<td>yes or no</td>
</tr>
<tr>
<td>:LK:</td>
<td>Locked releases</td>
<td></td>
<td>:R: ...</td>
</tr>
<tr>
<td>:Q:</td>
<td>User defined keyword</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:M:</td>
<td>Module name</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:FB:</td>
<td>Floor boundary</td>
<td></td>
<td>:R:</td>
</tr>
<tr>
<td>:CB:</td>
<td>Ceiling boundary</td>
<td></td>
<td>:R:</td>
</tr>
<tr>
<td>:Ds:</td>
<td>Default SID</td>
<td></td>
<td>:I:</td>
</tr>
<tr>
<td>:Nd:</td>
<td>Null delta flag</td>
<td></td>
<td>yes or no</td>
</tr>
<tr>
<td>:FD:</td>
<td>File descriptive text</td>
<td>Comments</td>
<td>text</td>
</tr>
<tr>
<td>:BD:</td>
<td>Body</td>
<td>Body</td>
<td>text</td>
</tr>
<tr>
<td>:GB:</td>
<td>Gotten body</td>
<td></td>
<td>text</td>
</tr>
<tr>
<td>:k:</td>
<td>A form of what(1) string</td>
<td>N/A</td>
<td>:Z: :M: \t: I:</td>
</tr>
<tr>
<td>:z:</td>
<td>what(1) string delimiter</td>
<td>N/A</td>
<td>@(#)</td>
</tr>
<tr>
<td>:F:</td>
<td>SCCS file name</td>
<td>N/A</td>
<td>text</td>
</tr>
<tr>
<td>:PN:</td>
<td>SCCS file path name</td>
<td>N/A</td>
<td>text</td>
</tr>
</tbody>
</table>

EXAMPLES

prs -d"Users and/or user IDs for :F: are:\n:UN:" s.file

may produce on the standard output:

Users and/or user IDs for s.file are:

xyz
131
abc

prs -d"Newest delta for pgm :M:: :I: Created :D: By :P:" -r
s.file

may produce on the standard output:

Newest delta for pgm main.c: 3.7 Created 77/12/1 By cas

As a special case:

prs s.file

may produce on the standard output:

D 1.1 77/12/1 00:00:00 cas 1 000000/00000/00000
MRs:
bl78-12345
bl79-54321
COMMENTS:

this is the comment line for s.file initial delta

for each delta table entry of the "D" type. The only option allowed
to be used with the special case is the -a option.

FILES

/tmp/pr?????

SEE ALSO

admin(1), delta(1), get(1), help(1), sccsfile(4).

Source Code Control System User Guide in the Support Tools
Guide.

DIAGNOSTICS

Use help(1) for explanations.
NAME
    ps - report process status
SYNOPSIS
    ps [ options ]
DESCRIPTION
    Ps prints certain information about active processes. Without
    options, information is printed about processes associated with the
    current terminal. The output consists of a short listing containing
    only the process ID, terminal identifier, cumulative execution
time, and the command name. Otherwise, the information that is
displayed is controlled by the selection of options.
OPTIONS
    Options using lists as arguments can have the list specified in one
    of two forms: a list of identifiers separated from one another by a
    comma, or a list of identifiers enclosed in double quotes and
    separated from one another by a comma and/or one or more spaces.
The options are:
    -e   Print information about all processes.
    -d   Print information about all processes, except process
group leaders.
    -a   Print information about all processes, except process
group leaders and processes not associated with a termi-
nal.
    -f   Generate a full listing. (See below for meaning of
columns in a full listing).
    -l   Generate a long listing. See below.
    -c   corefile Use the file corefile in place of /dev/mem. (Not available
         on 5000/20/30/40/50.)
    -s   swapdev
         Use the file swapdev in place of /dev/swap. This is use-
         ful when examining a corefile; a swapdev of /dev/null
         causes the user block to be zeroed out.
    -n   namelist
         The argument is taken as the name of an alternate sys-
tem namelist file in place of /unix or /syst.
    -t   termlist Restrict listing to data about the processes associated
         with the terminals given in termlist. Terminal identif-
         iers may be specified in one of two forms: the device file
         name (e.g., tty04) or if the device file name starts with
         tty, just the digit identifier (e.g., 04).
    -p   proclist
         Restrict listing to data about processes whose process
         ID numbers are given in proclist.
    -u   uidlist
         Restrict listing to data about processes whose user ID
         numbers or login names are given in uidlist. In the list-
ing, the numerical user ID is printed unless the -f option
         is used, in which case the login name is printed.
    -g   grplist
         Restrict listing to data about processes whose process
group leaders are given in grplist.
OUTPUT DESCRIPTION

The column headings and the meaning of the columns in a `ps` listing are given below; the letters f and l indicate the option (full or long) that causes the corresponding heading to appear; all means that the heading always appears. Note that these two options determine only what information is provided for a process; they do not determine which processes are listed.

5000 Series Systems

<table>
<thead>
<tr>
<th>F</th>
<th>(l)</th>
<th>Flags (octal and additive) associated with the process:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 swapped;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 in core;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 system process;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 locked-in core (e.g., for physical I/O);</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 being swapped;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20 being traced by another process;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40 another tracing flag;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100 text pointer valid;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3B 20 computer: swapin segment expansion;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>200 3B 20 computer: process is child (during fork swap);</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VAX-11/780: process is partially swapped.</td>
</tr>
</tbody>
</table>

7000 Series Systems

<table>
<thead>
<tr>
<th>F</th>
<th>(l)</th>
<th>Flags (hexadecimal and additive) associated with the process:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1 in core;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 swapper or pager process;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 process being swapped out;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 save area flag;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 process is being traced;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20 another tracing flag;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40 user settable lock in core;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>80 process in page wait state;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100 another flag to prevent swap out;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>200 delayed unlock of pages;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>400 working on exiting;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>800 doing physical I/O;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000 process resulted from vfork();</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2000 another vfork() flag;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4000 no vm, parent in a vfork();</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8000 init data space on demand from inode;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10000 system detected anomalous vm behaviour;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20000 user warned of anomalous vm behaviour;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40000 timing out during sleep;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>80000 detached inherited by init;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100000 using old signal mechanism;</td>
</tr>
</tbody>
</table>
5000 Series Systems

S (1) The state of the process:
   0 non-existent;
   S sleeping;
   W waiting;
   R running;
   I intermediate;
   Z terminated;
   T stopped;
   X growing.

7000 Series System

STAT (f,l) state of the process:
   the state is given by a sequence of three letters, e.g. "RWN". The first letter indicates the runnabil-
   ity of the process: R for runnable processes, r for processes running on Slave (6/32MP only), T for
   stopped processes, P for processes in page wait, D for those in disk (or other short term) waits, S for
   those sleeping for less than about 20 seconds, and I for idle (sleeping longer than about 20 seconds)
   processes. The second letter indicates whether a process is swapped out, showing W if it is, or a blank
   if it is loaded (in-core); a process which has speci-
   fied a soft limit on memory requirements and which is
   exceeding that limit shows >; such a process is
   (necessarily) not swapped. The third letter
   indicates whether a process is running with altered
   CPU scheduling priority (nice); if the process priority is
   reduced, an N is shown, if the process priority has
   been artificially raised then a '<' is shown; processes
   running without special treatment have just a blank.

(Unless otherwise noted, the following features apply to the
5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 and the
7000 Series only.)

UID (f,l)
   The user ID number of the process owner; the login name is
   printed under the -f option.

PID (all)
   The process ID of the process; it is possible to kill a process if
   you know this datum.

PPID (f,l)
   The process ID of the parent process.

C (f,l)
   Processor utilization for scheduling.

PRI (l)
   The priority of the process; higher numbers mean lower prior-
   ity.
NI  (1)  
   Nice value; used in priority computation.
ADDR  (1)  
   The memory address of the process if resident; otherwise, the
   disk address.
SZ  (1)  
   The size in blocks of the core image of the process.
RSS  (1)  
   The real memory (resident set) size in blocks of the core image
   of the process. Not all machines display this column. (Available
   with the 7000 Series only.)
WCHAN  (1)  
   The event for which the process is waiting or sleeping; if
   blank, the process is running.
STIME  (f)  
   Starting time of the process.
TTY  (all)  
   The controlling terminal for the process.
TIME  (all)  
   The cumulative execution time for the process.
CMD  (all)  
   The command name; the full command name and its arguments
   are printed under the -f option.

A process that has exited and has a parent, but has not yet been
waited for by the parent, is marked <defunct>.

Under the -f option, ps tries to determine the command name and
arguments given when the process was created by examining
memory or the swap area. Failing this, the command name, as it
would appear without the -f option, is printed in square brackets.

FILES
   /unix          system namelist
   /syst          system namelist (5000/60/80/90 only)
   /dev/mem       memory
   /dev/swap      the default swap device
   /etc/passwd    supplies UID information
   /etc/ps_data   internal data structure
   /dev           searched to find terminal (tty) names

SEE ALSO
   acctcom(1), kill(1), nice(1).

RESTRICTIONS
   Things can change while ps is running; the picture it gives is only
   a close approximation to reality. Some data printed for defunct
   processes are irrelevant.

If the /etc/ps_data/ file is not current (i.e. after a kernel
remake), ps gives invalid results. To make /etc/ps_data current,
remove it and run ps again. (This paragraph is applicable to the
5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.)
NAME
ptx - permuted index

SYNOPSIS
ptx [ options ] [ input [ output ] ]

DESCRIPTION

Ptx generates the file output that can be processed with a text formatter to produce a permuted index of file input (standard input and output default). Ptx has three phases:

1. Do the permutation generating one line for each keyword in an input line.
2. Rotate the keyword to the front and sort the permuted file.
3. Rotate the sorted lines so the keyword comes at the middle of each line.

Ptx output is in the form:

.xx "tail" "before keyword" "keyword and after" "head"

where .xx is assumed to be an nroff(1) or troff(1) macro provided by the user or provided by the mptx(5) macro package. The before keyword and keyword and after fields incorporate as much of the line as fits around the keyword when it is printed. Tail and head, at least one of which is always the empty string, are wrapped-around pieces small enough to fit in the unused space at the opposite end of the line.

OPTIONS

If the -I and -o options are missing, ptx uses /usr/lib/elgn as the ignore file.

-f Fold upper and lower case letters for sorting.
-t Prepare the output for the phototypesetter.
-w n Use n as the length of the output line. The default line length is 72 characters for nroff and 100 for troff.
-g n Use n as the number of characters that ptx reserves in its calculations for each gap among the four parts of the line as finally printed. The default gap is 3.
-o file Use as keywords only the words given in file.
-i file Do not use as keywords any words given in file.
-b file Use the characters in file to separate words. Tab, newline, and space characters are always used as break characters.
-r Assume any leading non-blank characters of each input line to be a reference identifier (as to a page or chapter) separate from the text of the line. Attach that identifier as a fifth field on each output line.

FILES
/bin/sort
/usr/lib/eign
/usr/lib/tmac/tmac.ptx

SEE ALSO
nroff(1), troff(1), mm(5), mptx(5).

RESTRICTIONS
Line length counts do not account for overstriking or proportional spacing.
Lines that contain tildes (~) are not processed correctly because ptx uses that character internally.
NAME
  pwd - working directory name

SYNOPSIS
  pwd

DESCRIPTION
  *pwd* prints the path name of the working (current) directory.

SEE ALSO
  cd(1).

DIAGNOSTICS
  The messages
    Cannot open ..
    Read error in ..
  indicate possible file system trouble and should be referred to the
  system administrator.
NAME
ratfor - rational Fortran dialect

SYNOPSIS
ratfor [ options ] [ files ]

DESCRIPTION
Ratfor converts a rational dialect of Fortran into ordinary Fortran. Ratfor provides control flow constructs essentially identical to those in C:

statement grouping:
{ statement; statement; statement }

decision-making:
if (condition) statement [ else statement ]
switch (integer value) {
    case integer: statement
    ... [ default: ] statement
}

loops:
while (condition) statement
for (expression; condition; expression) statement
do limits statement
repeat statement [ until (condition) ]
break
next

and some syntactic features to make programs easier to read and write:

free form input:
multiple statements/line; automatic continuation

comments:
# this is a comment.

translation of relationals:
>, >=, etc., become .GT., .GE., etc.

return expression to caller from function:
return (expression)

define:
define name replacement

include:
include file

Ratfor is best used with f77(1).

OPTIONS
-h Turns quoted strings into 27H constructs.
-C Copies comments to the output and attempts to format them neatly.
-6x
Specifications x as the continuation character and places it in column
6. Ratfor normally marks continuation lines with an & in column 1.

SEE ALSO

efl(1), f77(1).

*Ratfor* in the Programming Guide.
NAME
  regcmp - regular expression compile

SYNOPSIS
  regcmp [ - ] files

DESCRIPTION
  Regcmp, in most cases, precludes the need for calling regcmp(3X)
  from C programs. This reduces both execution time and program
  size.

  The command regcmp compiles the regular expressions in file and
  places the output in file.i. If the - option is used, regcmp places
  the output in file.c. The format of entries in file is a name (C vari-
  able) followed by one or more blanks followed by a regular expres-
  sion enclosed in double quotes. The output of regcmp is C source
  code. Compiled regular expressions are represented as extern
  char vectors. File.i files may thus be included into C programs, or
  file.c files may be compiled and later loaded. In the C program
  which uses the regcmp output, regex(abc, line) applies the regu-
  lar expression named abc to line.

EXAMPLES
  name  "((A-Za-z)[A-Za-z0-9]*)$0"
  telno "\(\{0,1\}\(\{2-9\}[01][1-9]\)\$0\)\{0,1\} *
         "\([2-9][0-9]{2}\)\$1[ -]\{0,1\}"
         "\([0-9]{4}\)\$2"

  In the C program that uses the regcmp output,
      regex(telno, line, area, exch, rest)
  applies the regular expression named telno to line.

SEE ALSO
  regcmp(3X).
NAME
   rev - reverse lines of a file

SYNOPSIS
   rev [ file ] ...

DESCRIPTION
   Rev copies the named files to the standard output, reversing the order of characters in every line. If no file is specified, the standard input is copied.
[This page left blank.]
NAME
  rm, rmdir - remove files or directories

SYNOPSIS
  rm [ -fri ] file ...
  rmdir dir ...

DESCRIPTION
  `rm` removes the entries for one or more files from a directory. If an
  entry was the last link to the file, the file is destroyed. Removal of
  a file requires write permission in its directory, but neither read
  nor write permission on the file itself.

  If a file has no write permission and the standard input is a termi­
  nal, the file permissions permissions are printed and a line is read
  from the standard input. If that line begins with y the file is
  deleted, otherwise the file remains. `Rm` or `rmdir` does not take this
  precaution if the `-f` option is given or if the standard input is not a
  terminal.

  If a designated file is a directory, an error message is printed
  unless the option `-r` is used.

  `Rmdir` removes entries for the named directories which must be
  empty.

OPTIONS
  -f Remove files without verification from standard input (see
    DESCRIPTION above).
  -r Recursively delete the entire contents of the specified direc­
    tory and the directory itself.
  -i Inquire whether to delete each file and, if the `-r` option is also
    used, whether to examine each directory.

EXAMPLES
  To remove all files in the current directory which have file names
  beginning with ch, enter
    rm ch*
  Be careful using metacharacters (wildcards) so that you do not
  destroy files you mean to keep.
  To remove the oldstuff directory and all files and subdirectories
  contained in the oldstuff directory, enter
    rm -r oldstuff
  To remove the oldstuff directory and all files and subdirectories
  contained in the oldstuff directory verifying each file removal, enter
    rm -ir oldstuff

SEE ALSO
  unlink(2).

NOTE
  The file .. (dotdot) cannot be removed.
NAME
rmdel - remove a delta from an SCCS file

SYNOPSIS
rmdel -rSID files

DESCRIPTION
Rmdel removes the delta specified by the SID from each named SCCS file. The delta to be removed must be the newest (most recent) delta in its branch in the delta chain of each named SCCS file.

In addition, the delta specified must not be that of a version being edited for the purpose of making a delta; that is, if a p-file (see get(1)) exists for the named SCCS file, the delta specified must not appear in any entry of the p-file.

If a directory is named, rmdel behaves as though each file in the directory were specified as a named file, except that non-SCCS files (last component of the path name does not begin with s.) and unreadable files are silently ignored. If a name of - is given, rmdel reads the standard input; each line of the standard input is taken to be the name of an SCCS file to be processed; non-SCCS files and unreadable files are silently ignored.

The exact permissions necessary to remove a delta are documented in the Source Code Control System User Guide. Simply stated, they are
- if you make a delta you can remove it
- if you own the file and directory you can remove a delta

FILES
x.file See delta(1)
z.file See delta(1)

SEE ALSO
delta(1), get(1), help(1), prs(1), sccsfile(4).

DIAGNOSTICS
Use help(1) for explanations.
[This page left blank.]
NAME
rz, rb - XMODEM, YMODEM, ZMODEM (batch) file receive

SYNOPSIS
rz [ - +labDpqtuv ]
rb [ - +labDqtuv ]
rz [ - 1abcqtuv ] file
[ - ] [ v ] rzCOMMAND

DESCRIPTION
This command is applicable to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.

The rz and the rb commands uses the XMODEM, YMODEM, or ZMODEM error correcting protocol to receive files over a serial port from a variety of programs running under PC-DOS, CP/M, UNIX, and other operating systems.

The first form of rz (Receive ZMODEM) receives files with the ZMODEM batch protocol. If the sending program does not support ZMODEM, rz steps down to YMODEM protocol after 50 seconds. This delay can be eliminated by calling the program as rb.

When receiving with XMODEM or YMODEM, rz accepts either standard 128 byte sectors or 1024 byte sectors. The user should determine when the longer block length actually improves throughput without causing problems.

If extended file information (file length, etc.) is received, the file length controls the number of bytes written to the output dataset (YMODEM only), and the modify time and file mode (if non zero) are set accordingly.

If no extended file information is received, slashes in the pathname are changed to underscore, and any trailing period in the pathname is eliminated. This conversion is useful for files received from CP/M systems. With YMODEM, each file name is converted to lower case unless it contains one or more lower case letters.

The second form of rz receives a single file with XMODEM protocol. The user must supply the file name to both sending and receiving programs.

The third form of rz is invoked as rzCOMMAND (with an optional leading - as generated by login(1)). For each received file, rz pipes the file to "COMMAND filename" where filename is the name of the transmitted file with the file contents as standard input.

Each file transfer is acknowledged when COMMAND exits with 0 status. A non zero exit status terminates the transfers.

A typical use for this form is rzrmall which calls rmall(1) to post mail to the user specified by the transmitted file name. For example, sending the file "caf" from a PC-DOS system to rzrmall on a UNIX system would result in the contents of the DOS file "caf" being mailed to user "caf".
On some UNIX systems, the login directory must contain a link to COMMAND as login sets SHELL=rsh which does not permit absolute pathnames. If invoked with a leading v, rz reports progress to /tmps/rzlog. The following entry in /etc/passwd works for UNIX systems:

rzrmail::100:100::/usr/spool/uucppublic:/usr/bin/rzrmail

If the SHELL environment variable includes rsh or rksh (restricted shell), rz does not accept absolute pathnames or references to a parent directory, does not modify an existing file, and removes any files received in error.

If rz is invoked with stdout and stderr to different datasets, verbose is set to 2, causing frame by frame progress reports to stderr. This may be disabled with the q option.

ZMODEM CAPABILITIES
Rz supports incoming ZMODEM binary (-b), ASCII (-a), protect (-p), and append (+) requests, and ZMODEM command execution.

OPTIONS
+ Append received data to any existing file of the same name.
1 Use file descriptor 1 for ioctl and reads (UNIX only). By default, file descriptor 0 is used. This option permits rz to be used with the cu -? command. If the calling program has spawned a separate process to read characters from the modem, that process must be disabled for rz to operate properly.
a Convert files to UNIX conventions by stripping carriage returns and all characters beginning with the first Control Z (CP/M end of file).
b Binary file transfer override. Do not convert to ASCII.
c Request 16 bit CRC. XMODEM file transfers default to 8 bit checksum. YMODEM and ZMODEM normally use 16 bit CRC.
D (ZMODEM) Output file data to /dev/null; for testing.
p (ZMODEM) Protect: skip file if destination file exists.
q Quiet suppresses verbosity.
tim Change timeout to tim tenths of seconds.
u Make received pathnames lower case.
v Verbose causes a list of file names to be appended to /tmp/rzlog. More v's generate more output.

EXAMPLES
To download a file via XMODEM protocol while logged onto a bulletin board using cu, first issue the bulletin board command to begin the download then enter:

```
~?rz -1b filename
```

This sequence is the equivalent of the built-in cu command:

```
~&nml filename
```

To receive a file by way of XMODEM protocol while logged onto a remote system which has cu, enter:
sz -X -fqy filename
  ~?rz -1b filename
This sequence is the equivalent of the built-in cu command:
  ~%takez filename
To receive any number of files via ZMODEM protocol while logged onto a remote system which has cu, enter:
  sz -fqy files ....
  ~?rz -1b
This sequence is the equivalent of the built-in cu command:
  ~%takez filename

FILES
/tmp/rzlog stores debugging output generated with -vv option.

NOTES
The UNIX "ulimit" parameter must be set high enough to permit large file transfers.
The TTY input buffering on some systems may not allow long blocks or streaming input, especially at high baud rates.

SEE ALSO
cu(1c), sz(1), ulimit(2).

RESTRICTIONS
Pathnames are restricted to 127 characters. In XMODEM single file mode, the pathname given on the command line is still processed as described above. The ASCII option's CR/LF to NL translation merely deletes CR's.
Some versions of UNIX cu(1) do not operate properly with this program.
Improperly specified options and failing file transfers may leave the terminal in an unpredictable state.

ZMODEM CAPABILITIES
Rz supports incoming ZMODEM binary (-b), ASCII (-a), protect (-p), and append (-+) requests, and ZMODEM command execution.
[This page left blank.]
NAME
sact - print current SCCS file editing activity

SYNOPSIS
sact files

DESCRIPTION
Sact informs the user of any impending deltas to a named SCCS file. This situation occurs when get(1) with the -e option has been previously executed without a subsequent execution of delta(1). If file is a directory, sact behaves as though each file in the directory were specified as a named file, except that non-SCCS files and unreadable files are silently ignored. If a name of - is given, sact reads the standard input assuming each line is the name of an SCCS file to be processed.

The output for each named file consists of five fields separated by spaces.

Field 1 the SID of a delta that currently exists in the SCCS file to which changes will be made to make the new delta.
Field 2 the SID of the new delta to be created.
Field 3 the logname of the user who will make the delta (i.e. the user who executed a get for editing).
Field 4 the date that get -e was executed.
Field 5 the time that get -e was executed.

SEE ALSO
delta(1), get(1), unget(1).

DIAGNOSTICS
Use help(1) for explanations.
Page 2
NAME
sag - system activity graph

SYNOPSIS
sag [ options ]

DESCRIPTION
Not on 7000/40.
Sag graphically displays the system activity data stored in a binary data file by a previous sar(1) run. Any of the sar data items may be plotted singly, or in combination; as cross plots, or versus time. Simple arithmetic combinations of data may be specified. Sag invokes sar and finds the desired data by string-matching the data column header (run sar to see what data is available).

OPTIONS
These options are passed through to sar:
-s time Select data later than time in the form hh [:mm]. Default is 08:00.
-e time Select data up to time. Default is 18:00.
-i sec Select data at intervals as close as possible to sec seconds.
-f file Use file as the data source for sar. Default is the current daily data file /usr/adm/sa/sadd.

Other options:
-T term
Produce output suitable for terminal term. See tplot(1G) for known terminals. If term is vpr, output is processed by vpr -p and queued to a Versatec printer. Default for term is $TERM.

-x spec
x axis specification with spec in the form described under AXIS SPECIFICATION.
Sag permits a single spec for the x axis. If unspecified, time is used.

-y spec
y axis specification with spec in the form described under AXIS SPECIFICATION. Up to 5 specs separated by ; may be given for the y axis. The -y default is:

-y "%usr 0 100; %usr + %sys 0 100; %usr + %sys + %wio 0 100"

AXIS SPECIFICATION
An axis specification is in the form
"name [op name] ... [lo hi]"
Name is either an integer value, or a string that matches a column header in the sar report, with an optional device name in square brackets, e.g., r+w/s[dsk-1].
Op is + - * or / surrounded by blanks. Up to five names may be specified. Parentheses are not recognized.
Contrary to custom, + and - have precedence over * and /. Evaluation is left to right. Thus sag evaluates \( \frac{A}{A+B} \times 100 \) as \( \frac{A}{(A+B)} \times 100 \), and \( \frac{A+B}{C+D} \) as \( \frac{A+B}{(C+D)} \).

Lo and hi are optional numeric scale limits. If unspecified, they are deduced from the data.

Enclose the -x and -y arguments in double quotes if blanks or `<CR>` are included in the specification.

EXAMPLES
To see CPU utilization for today:

```
  sag
```

To see activity over 15 minutes of all disk drives:

```
TS='date +%H:%M'
sar -o tempfile 60 15
TE='date +%H:%M'
sag -f tempfile -s $TS -e $TE -y "r+w/s[dsk]"
```

FILES
/usr/adm/sa/sadd daily data file for day dd.

SEE ALSO
sar(1), tplot(1G).
NAME
sar - system activity reporter

SYNOPSIS
sar [ -ubdycwaqmA ] [ -o file ] t [ n ]
sar [ -ubdycwaqmA ] [ -s time ] [ -e time ] [ -i sec ] [ -f file ]

DESCRIPTION
Sar, using the first syntax, samples cumulative activity counters in the operating system at n intervals of t seconds. The default value of n is 1. If the -o option is specified, sar save the samples in file in binary format.

Using the second syntax, with no sampling interval specified, sar extracts data from a previously recorded file, either the one specified aby the -f option or, if -f is not used, the standard system activity daily data file /usr/adm/SA/sadd for the current date dd. The starting and ending times of the report can be bounded via the -s and -e time arguments of the form hh[:mm[:ss]]. The -i option selects records a t sec second intervals. Otherwise, all intervals found in the data file are reported.

OPTIONS
The following options, valid in both usages of sar, specify the subsets of data to be printed. Listed under each option are column meanings. If no options are specified, the -u option is assumed.

-u Report CPU utilization:

%usr - Portion of time running is user mode,
%wio - Portion of time idle with some process waiting for block I/O.
%idle - Portion of time otherwise idle.

-b Report buffer activity:

bread/s, brwrit/s - Transfers per second of data between system buffers and disk or other block devices.
bread/s, lwrit/s - Accesses of system buffers.
%rcache, %wcache - Cache match ratios, e.g., 1 - bread/bread.
pread/s, pwrit/s - Transfers via raw device mechanism.

-d Report activity for each block device, e.g., disk or tape drive:

%busy - Portion of time device was busy servicing a transfer request. (5000/60/80/90 only, not available on 5000/20/30/40/50.)
avque - Average number of requests outstanding during %busy. (5000/60/80/90 only, not available on 5000/20/30/40/50.)
$r^+w/s$ - Number of data transfers to or from a device.

$blk/s$ - Number of bytes transferred to or from a device in 512-byte units.

$avrwait$ - Average time is ms. that transfer requests wait idly on queue. (5000/60/80/90 only, not available on 5000/20/30/40/50.)

$avserv$ - Average time to be serviced (which for disks includes seek, rotational latency and data transfer times). (5000/60/80/90 only, not available on 5000/20/30/40/50.)

-y Report TTY device activity:

$fawch/s$ - Input character rate.

$canch/s$ - Input character rate processed by canon.

$outch/s$ - Output character rate.

$rcvin/s$ - Receive interrupt rates.

$xmtin/s$ - Transmit interrupt rates.

$mdmin/s$ - Modem interrupt rates.

-c Report system calls:

$scall/s$ - System calls of all types.

$sread/s, swrit/s, fork/s, exec/s$ - Specific system calls.

$rchar/s, wchar/s$ - Characters transferred by read and write system calls.

-w Report system swapping and switching activity:

$swpin/s, swpot/s$ - Number of transfers for swapins (including initial loading of some programs) and swapouts.

$bswin/s, bswor/s$ - Number of 512-byte units transferred for swapins (including initial loading of some programs) and swapouts.

$pswch/s$ - Process switches.

-a Report use of file access system routines: how many times per second the $iget/s, namei/s, dirblk/s$ routines were called.

-q Report average queue length while occupied, and percent (%) of time occupied:

$runq-sz, %runocc$ - Run queue of executable processes in memory.
**swpq-sz, %swpocc** - Swap queue of processes swapped out but ready to run.

- **v** Report status of text, process, inode and file tables:
  - `text-sz, proc-sz, indo-sz` - Entries/size for each table, evaluated one at sampling point.
  - `text-ov, proc-ov, file-ov` - Overflows occurring between sampling points.

- **m** Report message and semaphore activities:
  - `msg/s, sea/s` - Primitives per second.

- **A** Report all data. Equivalent to `-udqbwcayvm`.

**EXAMPLES**

To see CPU activity so far for today, enter:

```
sar
```

To watch CPU activity evolve for 10 minutes and save data, enter:

```
sar -o temp 60 10
```

**FILES**

```
/usr/adm/sa/sadd daily data file
```

**RESTRICTIONS**

The `-d` option does not work.

**SEE ALSO**

`sag(1G), sar(1M)`
[This page left blank.]
NAME
sccsdiff - compare two versions of an SCCS file

SYNOPSIS
sccsdiff -rSID1 -rSID2 [-p] [-sn] files

DESCRIPTION
Sccsdiff compares two versions of an SCCS file and generates the
differences between the two versions. Any number of SCCS files
may be specified, but options apply to all files.

OPTIONS
-rSID1 -rSID2
  Specifies SID1 and SID2 as the deltas of an SCCS file that are to
  be compared. Sccsdiff passes versions to bdiff(1) in the order
given.

-p  Pipes output for each file through pr(1).

-sn
  Denotes n as the file segment size that bdiff passes to diff(1).
  This is useful when diff fails due to a high system load.

FILES
/tmp/get????? Temporary files

SEE ALSO
bdiff(1), get(1), help(1), pr(1).

DIAGNOSTICS
The message

  file: No differences

is printed if the two versions are the same.
Use help(1) for explanations.
SCCSDIFF(1)

[This page left blank.]
NAME
script - make typescript of terminal session

SYNOPSIS
script [ -a ] [ file]

DESCRIPTION
7000 Series System only.

Script makes a typescript of everything printed on your terminal. The typescript is written to file, or appended to file if the -a option is given. It can be sent to the line printer later with lpr. If no file name is given, the typescript is saved in the file typescript.

The script ends when the forked shell exists.

This program is useful when using a CRT and a hard-copy record of the dialog is desired, as for a student handing in a program that was developed on a CRT when hard-copy terminals are in short supply. Script counts on the existence of pseudo terminals: dev/ptyp?. If they do not exist, use MAKEDEV in /dev to create them: MAKEDEV pty?.

RESTRICTIONS
Script places everything in the log file. This is not what the naive user expects.
[This page left blank.]
NAME
sdb - symbolic debugger

SYNOPSIS
sdb [-w] [-W] [ objfil [ corfil [ directory-list ] ] ]

DESCRIPTION
Sdb is a symbolic debugger that can be used with C and F77 programs. It may be used to examine their object files and core files and to provide a controlled environment for their execution.

Objfil is normally an executable program file which has been compiled with the -g (debug) option; if it has not been compiled with the -g option, or if it is not an executable file, the symbolic capabilities of sdb are limited, but the file can still be examined and the program debugged. The default for objfil is a.out. Corfil is assumed to be a core image file produced after executing objfil; the default for corfil is core. The core file need not be present. A - in place of corfil forces sdb to ignore any core image file. The colon separated list of directories (directory-list) is used to locate the source files used to build objfil.

It is useful to know that at any time there is a current line and current file. If corfil exists then they are initially set to the line and file containing the source statement at which the process terminated. Otherwise, they are set to the first line in main(). The current line and file may be changed with the source file examination commands.

By default, warnings are provided if the source files used in producing objfil cannot be found or are newer than objfil. This checking feature and the accompanying warnings may be disabled by the use of the -W option.

Names of variables are written just as they are in C or F77. Note that names in C are of arbitrary length; sdb does not truncate names. Variables local to a procedure may be accessed using the form procedure:variable. If no procedure name is given, the procedure containing the current line is used by default.

It is also possible to refer to structure members as variable.member, pointers to structure members as variable->member, and array elements as variable[number]. Pointers may be dereferenced by using the form pointer[0]. Combinations of these forms may also be used. F77 common variables may be referenced by using the name of the common block instead of the structure name. Blank common variables may be named by the form .variable. A number may be used in place of a structure variable name, in which case the number is viewed as the address of the structure, and the template used for the structure is that of the last structure referenced by sdb. An unqualified structure variable may also be used with various commands. Generally, sdb interprets a structure as a set of variables. Thus, sdb displays the values of all the elements of a structure when it is requested to display a structure. An exception to this interpretation occurs when displaying variable addresses. An entire structure does
have an address, and it is this value sdb displays, not the addresses of individual elements.

Elements of a multidimensional array may be referenced as variable[number]/number or as variable[number,number,...]. In place of number, the form number;number may be used to indicate a range of values, * may be used to indicate all valid values for that subscript, or subscripts may be omitted entirely if they are the last subscripts and the full range of values is desired. As with structures, sdb displays all the values of an array or of the section of an array if trailing subscripts are omitted. It displays only the address of the array itself or of the section specified by the user if subscripts are omitted. A multidimensional parameter in an F77 program cannot be displayed as an array, but it is actually a pointer, whose value is the location of the array. The array itself can be accessed symbolically from the calling function.

A particular instance of a variable on the stack may be referenced by using the form procedure:variable,number. All the variations mentioned in naming variables may be used. Number is the occurrence of the specified procedure on the stack, counting the top, or most current, as the first. If no procedure is specified, the procedure currently executing is used by default.

It is also possible to specify a variable by its address. All forms of integer constants which are valid in C may be used, so that addresses may be input in decimal, octal or hexadecimal.

Line numbers in the source program are referred to as filename:;number or procedure:;number. In either case the number is relative to the beginning of the file. If no procedure or file name is given, the current file is used by default. If no number is given, the first line of the named procedure or file is used.

While a process is running under sdb, all addresses refer to the executing program; otherwise they refer to objfil or corfil. An initial argument of -w permits overwriting locations in objfil.

Individual processor general registers may be named instead of variables by using the register name with a % prepended. The x command displays the current values of all the general registers. The contents of these registers can be displayed or modified.

Note that the hardware floating point registers of the 68881 math coprocessor are also available to be used as the 68020 general registers when the 68881 math coprocessor installed. These registers are named %fp0 through %fp7.

OPTIONS
-Permit overwriting locations in objfil.
-W Disable warnings.

ADDRESSES
The address in a file associated with a written address is determined by a mapping associated with that file. Each mapping is represented by two triples (b1, e1, f1) and (b2, e2, f2) and the
file address corresponding to a written address is calculated as follows:

\[ b1 = \text{file address} = \text{address} + f1 \cdot b1 \]

otherwise

\[ b2 = \text{file address} = \text{address} + f2 \cdot b2 \]

otherwise, the requested address is not valid. In some cases (e.g., for programs with separated I and D space) the two segments for a file may overlap.

The initial setting of both mappings is suitable for normal a.out and core files. If either file is not of the kind expected then, for that file, \( b1 \) is set to 0, \( e1 \) is set to the maximum file size, and \( f1 \) is set to 0; in this way the whole file can be examined with no address translation.

In order for \( sdb \) to be used on large files, all appropriate values are kept as signed 32-bit integers. The M command can be used to display or change the current values for the address maps.

COMMANDS
The commands for examining data in the program are:

- \( t \) Print a stack trace of the terminated or halted program.
- \( T \) Print the top line of the stack trace.
- \( \text{variable/clm} \) Print the value of variable according to length \( l \) and format \( m \).
  A numeric count \( c \) indicates that a region of memory, beginning at the address implied by variable, is to be displayed. The length specifiers are:
  - \( b \) one byte
  - \( h \) two bytes (half word)
  - \( l \) four bytes (long word)

  Valid values for \( m \) are:
  - \( c \) Character
  - \( d \) Decimal
  - \( u \) Decimal, unsigned
  - \( o \) Octal
  - \( x \) Hexadecimal
  - \( f \) 32-bit single precision floating point
  - \( g \) 64-bit double precision floating point
  - \( s \) Assume variable is a string pointer and print characters starting at the address pointed to by the variable.
  - \( a \) Print characters starting at the variable address. This format may not be used with register variables.
  - \( p \) Pointer to procedure
  - \( i \) Disassemble machine-language instruction with addresses printed numerically and symbolically.
  - \( I \) Disassemble machine-language instructions with addresses just printed numerically.

The length specifiers are only effective with the formats \( c, d, u, o \) and \( x \). Any of the specifiers, \( c, l, \) and \( m \), may be omitted.
If all are omitted, *sdb* chooses a length and a format suitable for the variable type as declared in the program. If \( m \) is specified, then this format is used for displaying the variable. A length specifier determines the output length of the value to be displayed, sometimes resulting in truncation. A count specifier \( c \) tells *sdb* to display that many units of memory, beginning at the address of \( \text{variable} \). The number of bytes in one such unit of memory is determined by the length specifier \( l \), or if no length is given, by the size associated with the \( \text{variable} \). If a count specifier is used for the \( s \) or a command, then that many characters are printed. Otherwise successive characters are printed until either a null byte is reached or 128 characters are printed. The last variable may be redisplayed with the command \( ./ \).

The *sh(1)* metacharacters \*\ and ? may be used within procedure and variable names, providing a limited form of pattern matching. If no procedure name is given, variables local to the current procedure and global variables are matched; if a procedure name is specified then only variables local to that procedure are matched. To match only global variables, the form \( \text{variable}\!\!:\text{pattern} \) is used.

\[\text{linenumber}\!\!:\text{lm}\]
\[\text{variable}\!\!:\text{lm}\]

Print the value at the address from a.out or I space given by \( \text{linenumber} \) or \( \text{variable} \) (procedure name), according to the format \( \text{lm} \). The default format is "i".

\[\text{variable}=\text{lm}\]
\[\text{linenumber}=\text{lm}\]
\[\text{number}=\text{lm}\]

Print the address of \( \text{variable} \) or \( \text{linenumber} \), or the value of \( \text{number} \), in the format specified by \( \text{lm} \). If no format is given, then \( \text{Ix} \) is used. The last variant of this command provides a convenient way to convert between decimal, octal and hexadecimal.

\[\text{variable}\!\!:\text{value}\]

Set \( \text{variable} \) to the given \( \text{value} \). The value may be a number, a character constant or a variable. The value must be well defined; expressions which produce more than one value, such as structures, are not allowed. Character constants are denoted 'character'. Numbers are viewed as integers unless a decimal point or exponent is used. In this case, they are treated as having the type double. Registers are viewed as integers. The \( \text{variable} \) may be an expression which indicates more than one variable, such as an array or structure name. If the address of a variable is given, it is regarded as the address of a variable of type \( \text{int} \). C conventions are used in any type conversions necessary to perform the indicated assignment.

\[\text{x}\]

Print the machine registers and the current machine-language instruction.
X  Print the current machine-language instruction.

The commands for examining source files are:

`fp?`
Print floating point register where ? is 0-7 (5000/90 only).

`e procedure`
`e file-name`
`e directory/`
`e directory file-name`
The first two forms set the current file to the file containing
procedure or to file-name. The current line is set to the first
line in the named procedure or file. Source files are assumed to
be in directory. The default is the current working directory.
The latter two forms change the value of directory. If no pro-
cedure, file name, or directory is given, the current procedure
name and file name are reported.

`/regular expression/`
Search forward from the current line for a line containing a
string matching regular expression as in ed(1). The trailing /
may be deleted.

`?regular expression?`
Search backward from the current line for a line containing a
string matching regular expression as in ed(1). The trailing ?
may be deleted.

`p  Print the current line.`
`z  Print the current line followed by the next 9 lines. Set the
current line to the last line printed.`
`w  Window. Print the 10 lines around the current line.
`number`
Set the current line to the given line number. Print the new
current line.

`count+`
Advance the current line by count lines. Print the new current
line.

`count-`
Retreat the current line by count lines. Print the new current
line.

The commands for controlling the execution of the source program
are:

`count r args`
`count R`
Run the program with the given arguments. The r command
with no arguments reuses the previous arguments to the pro-
gram while the R command runs the program with no argu-
ments. An argument beginning with < or > causes redirection
for the standard input or output, respectively. If count is
given, it specifies the number of breakpoints to be ignored.

`linenumber c count`
`linenumber C count`
Continue after a breakpoint or interrupt. If `count` is given, it specifies the breakpoint at which to stop after ignoring `count - 1` breakpoints. C continues with the signal which caused the program to stop reactivated and c ignores it. If a line number is specified then a temporary breakpoint is placed at the line and execution is continued. The breakpoint is deleted when the command finishes.

`linenumber g count`
Continue after a breakpoint with execution resumed at the given line. If `count` is given, it specifies the number of breakpoints to be ignored.

`s count`
`S count`
Single step the program through `count` lines. If no count is given then the program is run for one line. S is equivalent to s except it steps through procedure calls.

`i`
`I` Single step by one machine-language instruction. I steps with the signal which caused the program to stop reactivated and i ignores it.

`variable$m count`
`address:m count`
Single step (as with s) until the specified location is modified with a new value. If `count` is omitted, it is effectively infinity. Variable must be accessible from the current procedure. Because this command is done by software, it can be very slow.

`level v`
Toggle verbose mode, for use when single stepping with S, s or m. If `level` is omitted, then just the current source file and/or subroutine name is printed when either changes. If `level` is 1 or greater, each C source line is printed before it is executed; if `level` is 2 or greater, each assembler statement is also printed. A v turns verbose mode off if it is on for any level.

`k` Kill the program being debugged.

`procedure(arg1, arg2, ...)`
`procedure(arg1, arg2, ...) /m`
Execute the named procedure with the given arguments. Arguments can be integer, character or string constants or names of variables accessible from the current procedure. The second form causes the value returned by the procedure to be printed according to format m. If no format is given, it defaults to d. Note that when the procedure completes, control is returned to sdb by a breakpoint.

`linenumber b commands`
Set a breakpoint at the given line. If a procedure name without
a line number is given (e.g., "proc:"), a breakpoint is placed at the first line in the procedure even if it was not compiled with the -g option. If no linenumber is given, a breakpoint is placed at the current line. If no commands are given, execution stops just before the breakpoint and control is returned to sdb. Otherwise the commands are executed when the breakpoint is encountered and execution continues. Multiple commands are specified by separating them with semicolons. If k is used as a command to execute at a breakpoint, control returns to sdb, instead of continuing execution.

B Print a list of the currently active breakpoints.

linenumber d
Delete a breakpoint at the given line. If no linenumber is given then the breakpoints are deleted interactively. Each breakpoint location is printed and a line is read from the standard input. If the line begins with a y or d then the breakpoint is deleted.

D Delete all breakpoints.

l Print the last executed line.

linenumber a
Announce. If linenumber is of the form proc: number, the command effectively does a linenumber b 1. If linenumber is of the form proc:, the command effectively does a proc: b T.

Miscellaneous commands:

!command
The command is interpreted by sh(1).

new-line
If the previous command printed a source line then advance the current line by one line and print the new current line. If the previous command displayed a memory location, then display the next memory location.

c control-d
Scroll. Print the next 10 lines of instructions, source or data depending on which was printed last.

< filename
Read commands from filename until the end of file is reached, and then continue to accept commands from standard input. When sdb is told to display a variable by a command in such a file, the variable name is displayed along with the value. This command may not be nested; < may not appear as a command in a file.

M Print the address maps.

M [?/] [*] b e f
Record new values for the address map. The arguments ? and / specify the text and data maps, respectively. The first segment, (bl, el, fl), is changed unless * is specified, in which case the second segment (bl, el, fl), of the mapping is
changed. If fewer than three values are given, the remaining map parameters are left unchanged.

"string"
Print the given string. The C escape sequences of the form \
\character are recognized, where character is a nonnumeric character.

q Exit the debugger.
The following commands also exist and are intended only for debugging the debugger:
V Print the version number.
Q Print a list of procedures and files being debugged.
Y Toggle debug output.

FILES
a.out
core

SEE ALSO
cc(1), f77(1), sh(1), a.out(4), core(4).
Symbolic Debugging Program - "sdb" in the Programming Guide.

WARNINGS
When sdb prints the value of an external variable for which there is no debugging information, a warning is printed before the value. The value is assumed to be int (integer).

Data which are stored in text sections are indistinguishable from functions.

Line number information in optimized functions is unreliable, and some information may be missing. Note that cc(1) disables optimization for modules compiled with the -g option.

RESTRICTIONS
If a procedure is called when the program is not stopped at a breakpoint (such as when a core image is being debugged), all variables are initialized before the procedure is started. This makes it impossible to use a procedure which formats data from a core image.

The default type for printing F77 parameters is incorrect. Their address is printed instead of their value.

Tracebacks containing F77 subprograms with multiple entry points may print too many arguments in the wrong order, but their values are correct.

The range of an F77 array subscript is assumed to be 1 to n, where n is the dimension corresponding to that subscript. This is only significant when the user omits a subscript or uses * to indicate the full range. There is no problem in general with arrays having subscripts whose lower bounds are not 1.
the -g option. If no linenumber is given, a breakpoint is placed at the current line. If no commands are given, execution stops just before the breakpoint and control is returned to sdb. Otherwise the commands are executed when the breakpoint is encountered and execution continues. Multiple commands are specified by separating them with semicolons. If k is used as a command to execute at a breakpoint, control returns to sdb, instead of continuing execution.

B Print a list of the currently active breakpoints.

linenumber d
Delete a breakpoint at the given line. If no linenumber is given then the breakpoints are deleted interactively. Each breakpoint location is printed and a line is read from the standard input. If the line begins with a y or d then the breakpoint is deleted.

D Delete all breakpoints.

l Print the last executed line.

linenumber a
Announce. If linenumber is of the form proc:number, the command effectively does a linenumber b 1. If linenumber is of the form proc:, the command effectively does a proc: b T.

Miscellaneous commands:

!command
The command is interpreted by sh(1).

new-line
If the previous command printed a source line then advance the current line by one line and print the new current line. If the previous command displayed a memory location, then display the next memory location.

control-d
Scroll. Print the next 10 lines of instructions, source or data depending on which was printed last.

< filename
Read commands from filename until the end of file is reached, and then continue to accept commands from standard input. When sdb is told to display a variable by a command in such a file, the variable name is displayed along with the value. This command may not be nested; < may not appear as a command in a file.

M Print the address maps.

M [?][*] b e f
Record new values for the address map. The arguments ? and / specify the text and data maps, respectively. The first segment, (bl, el, fl), is changed unless * is specified, in which case the second segment (bl, el, fl), of the mapping is changed. If fewer than three values are given, the remaining map parameters are left unchanged.
"string"
   Print the given string. The C escape sequences of the form \character are recognized, where character is a nonnumeric character.

q Exit the debugger.

The following commands also exist and are intended only for debugging the debugger:

V Print the version number.
Q Print a list of procedures and files being debugged.
Y Toggle debug output.

FILES
   a.out
   core

SEE ALSO
   cc(1), f77(1), sh(1), a.out(4), core(4).

Symbolic Debugging Program - "sdb" in the Programming Guide.

WARNINGS
   When sdb prints the value of an external variable for which there is no debugging information, a warning is printed before the value.
   The value is assumed to be int (integer).

   Data which are stored in text sections are indistinguishable from functions.

   Line number information in optimized functions is unreliable, and some information may be missing. Note that cc(1) disables optimization for modules compiled with the -g option.

RESTRICTIONS
   If a procedure is called when the program is not stopped at a breakpoint (such as when a core image is being debugged), all variables are initialized before the procedure is started. This makes it impossible to use a procedure which formats data from a core image.

   The default type for printing F77 parameters is incorrect. Their address is printed instead of their value.

   Tracebacks containing F77 subprograms with multiple entry points may print too many arguments in the wrong order, but their values are correct.

   The range of an F77 array subscript is assumed to be 1 to n, where n is the dimension corresponding to that subscript. This is only significant when the user omits a subscript or uses * to indicate the full range. There is no problem in general with arrays having subscripts whose lower bounds are not 1.
NAME
  sdiff - side-by-side difference program

SYNOPSIS
  sdiff [ options ... ] file1 file2

DESCRIPTION
  Sdiff uses the output of diff(1) to produce a side-by-side listing of
two files indicating those lines that are different. Each line of the
two files is printed with a blank gutter between them if the lines
are identical, a < in the gutter if the line only exists in file1, a > in
the gutter if the line only exists in file2, and a | for lines that are
different.

For example:

```
x | y
a a
b <
c <
d d
> c
```

OPTIONS
  -w n  Use the next argument, n, as the width of the output
        line. The default line length is 130 characters.
  -l    Only print the left side of any lines that are identical.
  -s    Do not print identical lines.
  -o output
        Use the next argument, output, as the name of a third file
        that is created as a user controlled merging of file1 and
        file2.

Sdiff copies identical lines of file1 and file2 to output. Sdiff prints sets of differences, as produced by diff(1); where a set of differences share a common gutter charac-
ter.

After printing each set of differences, sdiff prompts the
user with a % and waits for one of the following user-
typed commands:

1    append the left column to the output file
r    append the right column to the output file
s    turn on silent mode; do not print identical lines
v    turn off silent mode
e l   call the editor with the left column
e r   call the editor with the right column
e b   call the editor with the concatenation of left and
      right
e    call the editor with a zero length file
q   exit from the program

On exit from the editor, the resulting file is concatenated on the end of the output file.

SEE ALSO
diff(1), ed(1).
NAME
sed - stream editor

SYNOPSIS
sed [-n] [-e script] [-f sfile] [ files ]

DESCRIPTION
Sed copies the named files (standard input default) to the standard output, edited according to a script of commands.

A script consists of editing commands, one per line, of the following form:

[ address [, address ] ] function [ arguments ]

Address, function, and argument are described below.

In normal operation, sed cyclically copies a line of input into a pattern space (an internal edit buffer) and applies in sequence all commands whose addresses apply to the lines within the edit buffer. Editing commands can be applied only to non-selected pattern spaces by use of the negation function ! (below). At the end of the script, sed copies the pattern space to the standard output and deletes the pattern space. Some of the commands use an internal temporary buffer to save all or part of the edit buffer for subsequent retrieval.

OPTIONS
The following options can be specified several times.

-t sfile Takes the script from sfile.
-e script Specifies script as the script. If there is just one -e option and no -f option, the -e may be omitted. Script should be surrounded by quotes to isolate it from the shell.
-n Suppresses the default output.

ADDRESS
An address is either a decimal number that counts input lines cumulatively across files, a $ that addresses the last line of input, or a context address. A context address is a regular expression / in the style of ed(1), with the following differences:

The construction \?regular expression?, where ? is any character, is identical to /regular expression/. Note that in the context address \xabc\xdefx, the second x stands for itself, so that the regular expression is abcdef.

The escape sequence \n matches a new-line embedded in the pattern space.

A period . matches any character except the terminal new-line of the pattern space.

A command line with no addresses selects every pattern space.

A command line with one address selects each pattern space that matches the address.
A command line with two addresses selects the inclusive range from the first pattern space that matches the first address through the next pattern space that matches the second. If the second address is a number less than or equal to the line number first selected, only one line is selected. Thereafter the process is repeated, looking again for the first address.

FUNCTION
In the following list of functions the maximum number of permissible addresses for each function is indicated in the parentheses preceding each function.

(1) a\text  Append. Place text on the output before reading the next input line (see text below).

(2) b label  Branch to the : command bearing the label. If label is not specified, branch to the end of the script.

(2) c\text  Change. Delete the edit buffer. With 0 or 1 address or at the end of a 2-address range, place text on the output (see text below). Start the next cycle.

(2) d  Delete the edit buffer. Start the next cycle.

(2) D  Delete the initial segment of the edit buffer through the first new-line. Start the next cycle; do not copy a new line into the pattern space if any lines remain in the pattern space.

(2) g  Replace the contents of the edit buffer by the contents of the temporary buffer.

(2) G  Append the contents of the temporary buffer to the edit buffer.

(2) h  Replace the contents of the temporary buffer by the contents of the edit buffer.

(2) H  Append the contents of the edit buffer to the temporary buffer.

(1) i\text  Insert. Place text on the standard output. (see text below)

(2) l  List the edit buffer on the standard output in an unambiguous form. Spell non-printing characters in two-digit ASCII and fold long lines.

(2) n  Copy the edit buffer to the standard output. Replace the edit buffer with the next line of input.

(2) N  Append the next line of input to the edit buffer with an embedded new-line. (The current line number changes.)

(2) p  Print. Copy the edit buffer to the standard output.
(2) P  Copy the initial segment of the edit buffer through the first new-line to the standard output.

(1) q  Quit. Branch to the end of the script. Do not start a new cycle.

(2) r rfile  Read the contents of rfile. Place the contents of rfile on the output before reading the next input line. (see rfile below)

(2) s/regular expression /replacement /flags  Substitute the replacement string for instances of the regular expression in the edit buffer. Any character may be used instead of /. For a fuller description see ed(1). Flags is zero or more of:

  n  n = 1-512. Substitute for just the n-th occurrence of the regular expression.

  g  Global. Substitute for all nonoverlapping instances of the regular expression rather than just the first one.

  p  Print the edit buffer if a replacement was made.

  w wfile  Write. Append the edit buffer to wfile if a replacement was made. (see wfile below)

(2) t label  Test. Branch to the : command bearing the label if any substitutions have been made since the most recent reading of an input line or execution of a t. If label is empty, branch to the end of the script.

(2) w wfile  Write. Append the edit buffer to wfile. (see wfile below)

(2) x  Exchange the contents of the pattern and temporary buffers.

(2) y/string1 /string2 /  Transform. Replace all occurrences of characters in string1 with the corresponding character in string2. The lengths of string1 and string2 must be equal.

(2)! function  Apply the function (or group, if function is { }) only to lines not selected by the address(es).

(0) : label  Specify a label to which b and t commands may branch.

(1) =  Place the current line number on the standard output as a line.

(2) {  Execute the following commands through a matching } only when the edit buffer is selected.
An empty command is ignored.

If a # is the first character on the first line of a script file, the entire line is treated as a comment unless the character following the # is an n. If an n follows the #, the default output is suppressed and the rest of the line after the #n is ignored. A script file must contain at least one non-comment line.

ARGUMENT

text One or more lines of text; each line must end with a backslash to escape the new-line, except the last line. Backslashes within text are escape characters, and may be used to retain initial tabs and blanks that sed usually strips from every script line.

rfile Read file; this argument must be the last one in the command line and exactly one blank must separate it from its function.

wfile Write file; this argument must be the last one in the command line and exactly one blank must separate it from its function. Sed creates each wfile (up to ten distinct wfiles) before processing.

SEE ALSO

awk(1), ed(1), grep(1).
NAME
    set_tape - change the logical tape size for tape device

SYNOPSIS
    set_tape [ -s ] tape_device bytes

DESCRIPTION
    5000/30 and 5000/50 only.

    Set_tape changes the logical size, in bytes, for tape_device to
bytes. This is used when tape cartridges that have a lower or
higher capacity than the default value of 40960000 bytes are used.
The default value is that of a DC450A cartridge tape.

OPTION
    -s   Do not send output to standard output (silent)

EXAMPLE
    To set the logical size of /dev/rtp to 13.56 megabytes (DC150A car-
tridge tape), insert a tape in the cartridge tape drive and use the
command:

        set_tape /dev/rtp 13650000

    The output will be:

        Old tape capacity = 4096000
        New tape capacity = 13650000

SEE ALSO
    tape_size(1)

RESTRICTIONS
    Set_tape works only when a tape cartridge is mounted in the drive.
The logical size is set to the default, 40960000 bytes, when the sys-
tem is booted.
[This page left blank.]
NAME
setalign - set / unset alignment emulation

SYNOPSIS
setalign [ -yn ] [ -f ffile ] arg ...

DESCRIPTION
7000 Series Systems only.

Setalign will set or unset the alignment emulation capability in an executable file.

The -y option will set the alignment emulation by changing the magic number that is stored in the header of the file. When this condition is set, once the file is executed and encountered any alignment fault, the operating system will handle it correctly as the alignment fault does not exist.

The -n option will unset the alignment emulation. In this case, the process will get an illegal instruction and generate a core dump when the file encountered an alignment fault.

The -f option is given, the next argument is taken to be a file containing the names of the files to be examined.

Setalign will return the alignment emulation status if -y or -n is not specified.

SEE ALSO
ld(1).
NAME
/local/bin/setlp - set parameters for a line printer type device (parallel)

SYNOPSIS
setlp [ - options ]

DESCRIPTION
5000/60, 5000/80, and 5000/90 only.
setlp sets line printer options for the standard input device. Without arguments, it reports the current settings of the device. With the -g option, the current settings are reported in a form (ascii) that can be used as an argument to another setlp command.

Options:
  -i number  indent every line on the printout 'number' characters.
  -c number  print 'number' columns truncating anything left.
  -d number  wait for an acknowledge signal for 'number' units ( 1 units equals approximately 10-20 microseconds).
              The default is 100 units.
  -l number  print 'number' lines per page.
  nocr (-nocr) do not map (-map) NL to NL-CR on output.
  cap (-cap) map (do not map) lower case alphabetics to uppercase alphabetics on output.

EXAMPLE
To change the indent for the parallel port 0 printer from the default of 4 to 0.

setlp -i0 < /dev/gcp/lp/c0

FILES
/dev/gcp/lp/

SEE ALSO
lp(7) ioctl(2)
NAME
   set_tape - change the logical tape size for tape device

SYNOPSIS
   set_tape [ -s ] tape_device bytes

DESCRIPTION
   5000/30, 5000/35, 5000/50, and 5000/55 only.
   Set_tape changes the logical size, in bytes, for tape_device to
   bytes. This is used when tape cartridges that have a lower or
   higher capacity than the default value of 40960000 bytes are used.
   The default value is that of a DC450A cartridge tape.

OPTION
   - s   Do not send output to standard output (silent)

EXAMPLE
   To set the logical size of /dev/rtp for a DC150A cartridge tape,
   insert a tape in the cartridge tape drive and use the command:

   set_tape /dev/rtp 13200000

   The output will be:

   Old tape capacity = 40960000
   New tape capacity = 13200000

SEE ALSO
   tape_size(1)

RESTRICTIONS
   Set_tape works only when a tape cartridge is mounted in the drive.
   The logical size is set to the default, 40960000 bytes, when the sys-
   tem is booted.
SET_TAPE(1)

[This page left blank.]
NAME
   setulimit - set a user file size limit

SYNOPSIS
   setulimit size
   exec setulimit size

DESCRIPTION
   Setulimit allows the user to change the file size limit. Size is the
   number of 1024-byte blocks.

   Using setulimit without exec puts the user into a new shell in which
   the new size limit operates. The user may use Control-D to return
   to the login shell, where the standard default limit of two mega-
   bytes applies. Using setulimit with exec changes the limit and
   overlays the existing shell; only one process will be used, instead
   of two. Any errors that cause setulimit to exit will cause the pro-
   gram to logout.

   Care should be taken when using setulimit within the shell script.
   If setulimit without exec is used in a shell script, the entries follow-
   ing the setulimit are not processed until the new shell is exited. If
   the setulimit with exec is used in a shell script, the entries follow-
   ing the setulimit command are never processed.

   The maximum number of 1024-byte blocks allowed by setulimit is
determined by the value contained in the file /etc/SETULIMITMAX.
   Only the superuser can modify this file.

EXAMPLE
   To change the file size limit to 10240 blocks, and to enter a new
   shell, enter:

   setulimit 10240

FILES
   /etc/SETULIMITMAX  Maximum number of 1024-byte blocks
   allowed by setulimit(1). Must be greater
   than 1. If /etc/SETULIMITMAX does not
   exist or is not readable, the default is
   2048-byte block size.

DIAGNOSTICS
   The following message indicates an incorrect size value:

   Usage: setulimit NNN - where NNN is greater than 1 and less
   than or equal to [system maximum set by superuser]

   The following message indicates the ulimit call failed:

   Size change failed, setuid incorrect

   The following message indicates an invalid value for
   /etc/SETULIMITMAX:

   Invalid value() in /etc/SETULIMITMAX.
WARNING
Using *setulimit* without *exec* causes local variables defined in the parent not to be defined in the new shell.

Care should be taken when using *setulimit* within the shell script. If *setulimit* without *exec* is used in a shell script, the entries following the *setulimit* are not processed until the new shell is exited. If the *setulimit* with *exec* is used in a shell script, the entries following the *setulimit* command are never processed.
NAME
sh, rsh - shell, the standard/restricted command programming language

SYNOPSIS
sh [ -acefhiknrstuvx ] [ args ]
rsh [ -acefhiknrstuvx ] [ args ]

DESCRIPTION
Sh is a command programming language that executes commands read from a terminal or a file. Rsh is a restricted version of the standard command interpreter sh; it is used to set up login names and execution environments whose capabilities are more controlled than those of the standard shell. See Invocation below for the meaning of arguments to the shell.

Definitions
A blank is a tab or a space. A name is a sequence of letters, digits, or underscores beginning with a letter or underscore. A parameter is a name, a digit, or any of the characters *, @, #, ?, -, $, and !.

Commands
A simple-command is a sequence of non-blank 'Words separated by blanks. The first word specifies the name of the command to be executed. Except as specified below, the remaining words are passed as arguments to the invoked command. The command name is passed as argument 0 (see exec(2)). The value of a simple-command is its exit status if it terminates normally, or (octal) 200+status if it terminates abnormally (see signal(2) for a list of status values).

A pipeline is a sequence of one or more commands separated by ; (or, for historical compatibility, by `). The standard output of each command but the last is connected by a pipe(2) to the standard input of the next command. Each command is run as a separate process; the shell waits for the last command to terminate. The exit status of a pipeline is the exit status of the last command.

A list is a sequence of one or more pipelines separated by ;, &, &&, or ; ;, and optionally terminated by ; or &. Of these four symbols, ; and & have equal precedence, which is lower than that of && and ; ;. The symbols && and ; ; also have equal precedence. An arbitrary number of new-lines may appear in a list, instead of semicolons, to delimit commands.

The separators affect execution as follows:

;  Sequentially execute the preceding pipeline.
&  Asynchronously execute the preceding pipeline; that is, the shell does not wait for the pipeline to finish execution before proceeding to execute the next command.
&& Execute the following list only if the preceding pipeline returns a zero exit status.
Execute the following list only if the preceding pipeline returns a non-zero exit status.

A command is either a simple-command or one of the following. Unless otherwise stated, the value returned by a command is that of the last simple-command executed in the command.

```
for name [ in word ] do list done
```

Each time a for command is executed, name is set to the next word taken from the in word list. If in word is omitted, then the for command executes the do list once for each positional parameter that is set (see Parameter Substitution below). Execution ends when there are no more words in the list.

```
case word in [ pattern [ ; pattern ] . . . ) list ;; ] . . . esac
```

A case command executes the list associated with the first pattern that matches word. The form of the patterns is the same as that used for file-name generation (see File Name Generation) except that a slash, a leading dot, or a dot immediately following a slash need not be matched explicitly.

```
if list then list [ elif list then list ] . . . [ else list ] fi
```

The list following if is executed and, if it returns a zero exit status, the list following the first then is executed. Otherwise, the list following elif is executed and, if its value is zero, the list following the next then is executed. Failing that, the else list is executed. If no else list or then list is executed, then the if command returns a zero exit status.

```
while list do list done
```

A while command repeatedly executes the while list and, if the exit status of the last command in the list is zero, executes the do list; otherwise the loop terminates. If no commands in the do list are executed, then the while command returns a zero exit status; until may be used in place of while to negate the loop termination test.

```
(list)
   Execute list in a sub-shell.
{list;}
   Simply execute list.
name ( ) {list;}
   Define a function which is referenced by name. The body of the function is the list of commands between { and }. Execution of functions is described below (see Execution).
```

The following words are only recognized as the first word of a command and when not quoted:

```
if then else elif fi case esac for while until do done { }
```

Comments
A word beginning with # causes that word and all the following characters up to a new-line to be ignored.

Command Substitution
The standard output from a command enclosed in a pair of grave accents (` ``) may be used as part or all of a word; trailing new-lines are removed.
Parameter Substitution

The character $ is used to introduce substitutable parameters. There are two types of parameters, positional and keyword. If parameter is a digit, it is a positional parameter. Positional parameters may be assigned values by set. Keyword parameters (also known as variables) may be assigned values by:

\[ name=value [ name=value ] . . . \]

Pattern-matching is not performed on value. There cannot be a function and a variable with the same name.

\${parameter}\}

The value, if any, of the parameter is substituted. The braces are required only when parameter is followed by a letter, digit, or underscore that is not to be interpreted as part of its name. If parameter is * or @, all the positional parameters, starting with $1, are substituted (separated by spaces). Parameter $0 is set from argument zero when the shell is invoked.

\${parameter}:-word\}

If parameter is set and is non-null, substitute its value; otherwise substitute word.

\${parameter}:=word\}

If parameter is not set or is null set it to word; the value of the parameter is substituted. Positional parameters may not be assigned to in this way.

\${parameter}?:word\}

If parameter is set and is non-null, substitute its value; otherwise, print word and exit from the shell. If word is omitted, the message parameter null or not set is printed.

\${parameter}:+word\}

If parameter is set and is non-null, substitute word; otherwise substitute nothing.

In the above, word is not evaluated unless it is to be used as the substituted string, so that, in the following example, pwd is executed only if d is not set or is null:

\[ echo \${d:-'pwd'} \]

If the colon ( : ) is omitted from the above expressions, the shell only checks whether parameter is set or not.

The following parameters are automatically set by the shell:

- # The number of positional parameters in decimal.
- - Options supplied to the shell on invocation or by the set command.
- ? The decimal value returned by the last synchronously executed command.
- $ The process number of this shell.
- ! The process number of the last background command invoked.

The following parameters are used by the shell:

HOME
The default argument (home directory) for the cd(1)
command.

PATH
The search path for commands (see Execution below). The user may not change PATH if executing under rsh.

CDPATH
The search path for the cd(1) command.

MAIL
If this parameter is set to the name of a mail file and the MAILPATH parameter is not set, the shell informs the user of the arrival of mail in the specified file.

MAILCHECK
This parameter specifies how often (in seconds) the shell checks for the arrival of mail in the files specified by the MAILPATH or MAIL parameters. The default value is 600 seconds (10 minutes). If set to 0, the shell checks before each prompt.

MAILPATH
A colon (:) separated list of file names. If this parameter is set, the shell informs the user of the arrival of mail in any of the specified files. Each file name can be followed by % and a message that will be printed when the modification time changes. The default message is you have mail.

PS1
Primary prompt string, by default $.

PS2
Secondary prompt string, by default >.

IFS
Internal field separators, normally space, tab, and newline.

SHACCT
If this parameter is set to the name of a file writable by the user, the shell writes an accounting record in the file for each shell procedure executed. Accounting routines such as acctcom(1) and acctcms(1M) can be used to analyze the data collected.

SHELL
When the shell is invoked, it scans the environment (see Environment below) for this name. If it is found and there is an r in the file name part of its value, the shell becomes a restricted shell.

The shell gives default values to PATH, PS1, PS2, MAILCHECK and IFS. HOME and MAIL are set by login(1).

Blank Interpretation
After parameter and command substitution, the results of substitution are scanned for internal field separator characters (those found in IFS) and split into distinct arguments where such characters are found. Explicit null arguments ("" or ") are retained. Implicit null arguments (those resulting from parameters that have no values) are removed.
File Name Generation

Following substitution, each command word is scanned for the characters *, ?, and [. If one of these characters appears the word is regarded as a pattern. The word is replaced with alphabetically sorted file names that match the pattern. If no file name is found that matches the pattern, the word is left unchanged.

The character . at the start of a file name or immediately following a /, as well as the character / itself, must be matched explicitly.

* Matches any string, including the null string.
? Matches any single character.
[ ... ] Matches any one of the enclosed characters. A pair of characters separated by - matches any character lexically between the pair, inclusive. If the first character following the opening [ is a ! any character not enclosed is matched.

Quoting

The following characters have a special meaning to the shell and cause termination of a word unless quoted:

; & ( ) | ^ < > new-line space tab

A character may be quoted (i.e., made to stand for itself) by preceding it with a \. The pair \new-line is ignored. All characters enclosed between a pair of single quote marks (''), except a single quote, are quoted. Inside double quote marks (""), parameter and command substitution occurs and \ quotes the characters \, ' , " , and $. "$*" is equivalent to "$1 $2 . . . "$, whereas "$@" is equivalent to "$1 "$2 . . . .

Prompting

When used interactively, the shell prompts with the value of PS1 before reading a command. If at any time a new-line is typed and further input is needed to complete a command, the secondary prompt, the value of PS2, is issued.

Input/Output

Before a command is executed, its input and output may be redirected using a special notation interpreted by the shell. The following may appear anywhere in a simple-command or may precede or follow a command and are not passed on to the invoked command; substitution occurs before word or digit is used:

<word Use file word as standard input (file descriptor 0).
>word Use file word as standard output (file descriptor 1). If the file does not exist it is created; otherwise, it is truncated to zero length.
>>word Use file word as standard output. If the file exists output is appended to it after first seeking to the end-of-file; otherwise, the file is created.
<< [ - ]word Use the shell input (read up to a line that is the same as word, or to an end-of-file) as the standard input. If any character of word is quoted, no interpretation is placed upon the input characters; otherwise,
parameter and command substitution occurs, (unescaped) \new-line is ignored, and \ must be used to quote the characters \\
, $, ', and the first character of word. If - is appended to <<, all leading tabs are stripped from word and from shell input.

<&digit Use the file associated with file descriptor digit as standard input. Similarly for the standard output using >&digit.

<&- Close the standard input. Similarly for the standard output using >&-.

If any of the above is preceded by a digit, the file descriptor which is associated with the file is that specified by the digit (instead of the default 0 or 1). For example:

... 2>&1

associates file descriptor 2 with the file currently associated with file descriptor 1.

The order in which redirections are specified is significant. The shell evaluates redirections left-to-right. For example:

... 1>&xxx 2>&1

first associates file descriptor 1 with file xxx. It associates file descriptor 2 with the file associated with file descriptor 1 (i.e. xxx). If the order of redirections were reversed, file descriptor 2 would be associated with the terminal (assuming file descriptor 1 had been) and file descriptor 1 would be associated with file xxx.

If a command is followed by & the default standard input for the command is the empty file /dev/null. Otherwise, the environment for the execution of a command contains the file descriptors of the invoking shell as modified by input/output specifications.

Redirection of output is not allowed in the restricted shell.

Environment

The environment (see environ(5)) is a list of name-value pairs that is passed to an executed program in the same way as a normal argument list. The shell interacts with the environment in several ways. On invocation, the shell scans the environment and creates a parameter for each name found giving it the corresponding value. If the user modifies the value of any of these parameters or creates new parameters, none of these affects the environment unless the export command is used to bind the shell parameter to the environment (see also Invocation -a). A parameter may be removed from the environment with the unset command. The environment seen by any executed command is thus composed of any unmodified name-value pairs originally inherited by the shell, minus any pairs removed by unset, plus any modifications or additions, all of which must be noted in export commands.

The environment for any simple-command may be augmented by prefixing it with one or more assignments to parameters. Thus:
TERM=450 cmd and 
(export TERM; TERM=450; cmd)

are equivalent as far as the execution of cmd is concerned.

If the -k option is set, all keyword arguments are placed in the environment, even if they occur after the command name. The following first prints a=b c but then only c after the set -k:

echo a=b c
set -k
echo a=b c

Signals
The INTERRUPT and QUIT signals for an invoked command are ignored if the command is followed by &; otherwise signals have the values inherited by the shell from its parent, with the exception of signal 11 (memory fault) (but see also the trap command below).

Execution
Each time a command is executed, the above substitutions are carried out. If the command name matches one of the Special Commands listed below, it is executed in the shell process. If the command name does not match a Special Command, but matches the name of a defined function, the function is executed in the shell process (note how this differs from the execution of shell procedures). The positional parameters $1, $2, . . . are set to the arguments of the function. If the command name matches neither a Special Command nor the name of a defined function, a new process is created and an attempt is made to execute the command via exec(2).

The shell parameter PATH defines the search path for the directory containing the command. Alternative directory names are separated by a colon ( : ). The default path is :/bin:/usr/bin specifying the current directory, /bin, and /usr/bin, in that order. Note that the current directory is specified by a null path name, which can appear immediately after the equal sign or between the colon delimiters anywhere else in the path list. If the command name contains a / the search path is not used; such commands are not executed by the restricted shell. Otherwise, each directory in the path is searched for an executable file. If the file has execute permission but is not an a.out file, it is assumed to be a file containing shell commands. A sub-shell is spawned to read it. A parenthesized command is also executed in a sub-shell.

The location in the search path where a command was found is remembered by the shell to help avoid unnecessary execs later. If the command was found in a relative directory, its location must be re-determined whenever the current directory changes. The shell forgets all remembered locations whenever the PATH variable is changed or the hash -r command is executed (see below).

Special Commands
Input/output redirection is permitted for these commands. File descriptor 1 is the default output location.
read and execute commands from file and return. The search path specified by PATH is used to find the directory containing file.

break [n] Exit from the enclosing for or while loop, if any. If n is specified, break n levels.

continue [n] Resume the next iteration of the enclosing for or while loop. If n is specified, resume at the n-th enclosing loop.

cd [arg] Change the current directory to arg. The shell parameter HOME is the default arg. The shell parameter CDPATH defines the search path for the directory containing arg. Alternative directory names are separated by a colon (:). The default path is <null>, specifying the current directory. Note that the current directory is specified by a null path name, which can appear immediately after the equal sign or between the colon delimiters anywhere else in the path list. If arg begins with a /, the search path is not used. Otherwise, each directory in the path is searched for arg. The cd command may not be executed by rsh.

echo [arg . ..] Echo arguments. See echo(1) for usage and description.

eval [arg . ..] Read the arguments as input to the shell and execute the resulting command(s).

exec [arg . ..] Execute the command specified by the arguments in place of this shell without creating a new process. Input/output arguments may appear and, if no other arguments are given, cause the shell input/output to be modified.

exit [n] Exit a shell with the exit status specified by n. If n is omitted, the exit status is that of the last command executed (an end-of-file also causes the shell to exit.)

export [name . ..] Mark the given names for automatic export to the environment of subsequently-executed commands. If no arguments are given, a list of all names that are exported in this shell is printed. Function names may not be exported.

hash [-r] [name . ..] For each name, determine and remember the location in the search path of the command specified by name. The -r option causes the shell to forget all remembered locations. If no arguments are given, information about remembered commands is presented. Hits is the number of times a command has been invoked by the shell process. Cost is a measure of the work required to locate a command in the search path. There are
certain situations which require that the stored location of a command be recalculated. Commands for which this is done are indicated by an asterisk (*) adjacent to the hits information. Cost is incremented when the recalculation is done.

newgrp [ arg . . . ]
Equivalent to exec newgrp arg . . . . See newgrp(1) for usage and description.

pwd
Print the current working directory. See pwd(1) for usage and description.

read [ name . . . ]
Read one line from the standard input and assign the first word to the first name, the second word to the second name, etc., with leftover words assigned to the last name. The return code is 0 unless an end-of-file is encountered.

readonly [ name . . . ]
Mark the given names readonly; the values of the these names may not be changed by subsequent assignment. If no arguments are given, a list of all readonly names is printed.

return [ n ]
Exit a function with the return value specified by n. If n is omitted, the return status is that of the last command executed.

set [--aefhkntuvx [ arg . . . ] ]
Set options and positional parameters. See Invocation for a description of the options. Using + rather than - causes these options to be turned off. The current set of options may be found in $-. The remaining arguments are positional parameters and are assigned, in order, to $1, $2, . . . . If no arguments are given the values of all names are printed.

shift [ n ]
The positional parameters from $n+1 . . . are renamed $1 . . . . If n is not given, it is assumed to be 1.

test
Evaluate conditional expressions. See test(1) for usage and description.

times
Print the accumulated user and system times for processes run from the shell.

trap [ arg ] [ n ] . . .
Read the command arg and execute when signal(s) n is received. Note that arg is scanned once when the trap is set and once when the trap is taken. Trap commands are executed in order of signal number. Any attempt to set a trap on a signal that was ignored on entry to the current shell is ineffective. An attempt to trap on signal 11 (memory fault) produces an error. If arg is absent all trap(s) n are reset to their original values. If arg is the null string this signal is ignored by the shell and by the commands it invokes. If n is 0 the command arg is executed on exit from the shell. The trap command with no arguments prints a list of commands associated with each signal number.
type [ name . . . ]
For each name, indicate how it would be interpreted if used as a command name.

ulimit [ n ]
Impose a size limit of \( n \) blocks on files written by child processes (files of any size may be read). With no argument, the current limit is printed.

umask [ nnn ]
Set the user file-creation mask to \( nnn \) (see umask(2)). If \( nnn \) is omitted, the current value of the mask is printed.

unset [ name . . . ]
For each name, remove the corresponding variable or function. The variables PATH, PS1, PS2, MAILCHECK and IFS cannot be unset.

wait [ n ]
Wait for the specified process and report its termination status. If \( n \) is not given all currently active child processes are waited for and the return code is zero.

Invocation
If the shell is invoked through exec(2) and the first character of argument zero is '-', commands are initially read from /etc/profile and from $HOME/.profile, if such files exist. Thereafter, commands are read as described below, which is also the case when the shell is invoked as /bin/sh. The options below are interpreted by the shell on invocation. Note that unless the -c or -s option is specified, the first argument is assumed to be the name of a file containing commands, and the remaining arguments are passed as positional parameters to that command file.

-a Mark variables which are modified or created for export.
-c string
Read commands from string.
-e Exit immediately if a command exits with a non-zero exit status.
-f Disable file name generation.
-h Locate and remember function commands as functions are defined; function commands are normally located when the function is executed.
-i If the -i option is present or if the shell input and output are attached to a terminal, make this shell interactive. In this case TERMINATE is ignored (so that kill 0 does not kill an interactive shell) and INTERRUPT is caught and ignored (so that wait is interruptible). In all cases, QUIT is ignored by the shell.
-k Place all keyword arguments in the environment for a command, not just those that precede the command name.
-n Read commands but do not execute them.
-r Make the shell a restricted shell.
-s If the -s option is present or if no arguments remain, read commands from the standard input. Any remaining arguments specify the positional parameters. Shell output except for Special Commands is written to file descriptor
2.

-\(t\) Exit after reading and executing one command.
-\(u\) Treat unset variables as an error when substituting.
-\(v\) Print shell input lines as they are read.
-\(x\) Print commands and their arguments as they are executed.
-- Do not change any of the options; useful in setting $1 to -. 

Rsh Only

\(Rsh\) is used to set up login names and execution environments whose capabilities are more controlled than those of the standard shell. The actions of \(rsh\) are identical to those of \(sh\), except that the following are disallowed:
- changing directory (see cd(1)),
- setting the value of $PATH,
- specifying path or command names containing /,
- redirecting output (> and >>).

The restrictions above are enforced after .profile is interpreted.

When a command to be executed is found to be a shell procedure, \(rsh\) invokes \(sh\) to execute it. Thus, it is possible to provide to the end-user shell procedures that have access to the full power of the standard shell, while imposing a limited menu of commands; this scheme assumes that the end-user does not have write and execute permissions in the same directory. 

The net effect of these rules is that the writer of the .profile has complete control over user actions, by performing guaranteed setup actions and leaving the user in an appropriate directory (probably not the login directory).

The superuser often sets up a directory of commands (i.e., /usr/rbin) that can be safely invoked by \(rsh\). Some superusers also supply \(rsh\) users with the restricted editor red.

EXIT STATUS

Errors detected by the shell, such as syntax errors, cause the shell to return a non-zero exit status. If the shell is being used non-interactively execution of the shell file is abandoned. Otherwise, the shell returns the exit status of the last command executed (see also the exit command above).

FILES

/etc/profile
$HOME/.profile
/tmp/sh
/dev/null

SEE ALSO

acctcom(1), cd(1), echo(1), env(1), login(1), newgrp(1), pwd(1), test(1), umask(1), acctcms(1M), dup(2), exec(2), fork(2), pipe(2), signal(2), ulimit(2), umask(2), wait(2), a.out(4), profile(4), environ(5).

Shell Tutorial in the User Guide.
Shell Introduction, Using Shell Commands, Shell Programming,
and Examples of Shell Procedures in the Programming Guide.

WARNINGS
If a command is executed, and a command with the same name is installed in a directory in the search path before the directory where the original command was found, the shell continues to exec(2) the original command. Use the hash command to correct this situation.

If you move the current directory or one above it, pwd may not give the correct response. Use the cd command with a full path name to correct this situation.
NAME
shl - shell layer manager

SYNOPSIS
shl

DESCRIPTION
Shl allows a user to interact with more than one shell from a single terminal. The user controls these shells, known as layers, using the commands described below.

The current layer is the layer which can receive input from the keyboard. Other layers attempting to read from the keyboard are blocked. Output from multiple layers is multiplexed onto the terminal. The output of a layer may be blocked when the layer is not current by setting the loblk option of stty(1) or by issuing the block command of shl.

The stty control-character swtch (set to ^Z if NUL) is used to switch control to shl from a layer. Shl has its own prompt, >>>, to help distinguish it from a layer.

A layer is a shell which has been bound to a virtual tty device (/dev/sxt???). The virtual device can be manipulated like a real tty device using stty(1) and ioctl(2). Each layer has its own process group id.

A name is a sequence of characters delimited by a blank, tab or new-line. Only the first eight characters are significant. The names (1) through (7) cannot be used when creating a layer. They are used by shl when no name is supplied. These names may be abbreviated to just the digit.

COMMANDS
The following commands may be issued from the shl prompt level. Any unique prefix is accepted.

create [ name ]
Create a layer called name and make it the current layer. If no argument is given, a layer is created with a name of the form (#) where # is the last digit of the virtual device bound to the layer. The shell prompt variable PS1 is set to the name of the layer followed by a space. A maximum of seven layers can be created.

block name [ name ... ]
For each name, block the output of the corresponding layer when it is not the current layer. This is equivalent to setting the stty option loblk within the layer.

delete name [ name ... ]
For each name, delete the corresponding layer. All processes in the process group of the layer are sent the SIGHUP signal (see signal(2)).

help (or ?)
Print the syntax of the shl commands.

layers [-L] [ name ... ]
For each name, list the layer name and its process group. The
-l option produces a ps(1)-like listing. If no arguments are given, information is presented for all existing layers.

**resume [ name ]**
Make the layer referenced by name the current layer. If no argument is given, the last existing current layer is resumed.

**toggle**
Resume the layer that was current before the last current layer.

**unblock name [ name ... ]**
For each name, do not block the output of the corresponding layer when it is not the current layer. This is equivalent to setting the stty option -loblk within the layer.

**quit**
Exit shl. All layers are sent the SIGHUP signal.

**name**
Make the layer referenced by name the current layer.

**FILES**
/dev/sxt???
Virtual tty devices

$SHELL
Variable containing path name of the shell to use
(default is /bin/sh).

**SEE ALSO**
sh(1), stty(1), ioctl(2), signal(2), sxt(7).

**DIAGNOSTICS**
When the layers command is issued, shl attempts to determine the state of the existing layers.

If a layer has only performed a shell process since it was last exited, shl assumes that the process is either currently performing some shell command or is waiting for input. The status displayed is:

executing or awaiting input

If a layer has a process other than its shell process performing, shl can not determine if the process is performing or waiting for keyboard input but the status displayed is also:

executing or awaiting input

If the current process is a process other than a shell process and the process exits, shl assumes that the layer's shell process is now waiting for input and therefore the status displayed is:

blocked on input

**WARNINGS**
Pressing the Rubout or equivalent key while at the shl level prompt, may cause the entire line, including the shell prompt >>>, to be erased and positions the cursor on the next line. The shell prompt is not displayed but a command can be entered on the blank line. Pressing the Return or equivalent key causes the shell
prompt >>> to be displayed.

RESTRICTIONS

Shl can not be used as a login shell.

Shl can only be invoked when the system is in the multi-user mode. If shl is invoked while the system is in the single-user mode, the message 'Multiplex failed (errno = xx)' is displayed.

Shl can not be invoked from a previously created shell layer. If shl is invoked from a previously created shell layer, the error message 'No control channels available (errno = xx)' is displayed.
NAME
   show - display current hardware configuration

SYNOPSIS
   show [ -l | -list ] [ -h | -help ] [ pathname ]

DESCRIPTION
   This command is applicable to the 5000/30, 5000/35, 5000/50, and
   5000/55 Release 2.00.00 only.

   The show command is used to display the current hardware configu-
   ration. Invoking the show command without any options produces a report that displays the current hardware configuration.

   Because the show command uses the UNIX environment, the appropriate device driver(s) must be configured in the kernel so
   that all hardware components can be retrieved.

OPTIONS
   -l | -list
      The -l option lists the hardware configuration in a data format
      that can be used from a shellscript. Optionally, -list can be
      entered.

   -h | -help
      The -h option displays HELP information. The help information
      includes a description of all multibus boards, their correspond-
      ing abbreviations, and their physical addresses (hexadecimal).
      Optionally, -help can be entered.

   pathname
      The pathname option indicates the "full" pathname of the
      operating system (kernel); the default pathname is /unix.

   In addition, the -h option lists all disk device names (block and
   special/raw) and their corresponding major and minor numbers
   which is helpful if a device node must be created using the
   mknod command.

   EXAMPLES
      show
      show -h
      show -l
      show -l /unix.orig

SEE ALSO
   mknod(1M)
NAME
  size  - print section sizes of common object files

SYNOPSIS
  size [-o] [-x] [-V] files

DESCRIPTION
  The size command produces section size information for each sec­
  tion in the common object files. Size prints the size of the text,
  data and bss (uninitialized data) sections along with the total size
  of the object file. If an archive file is input to the size command the
  information for all archive members is displayed.
  
  Size prints numbers in decimal unless either the -o or the -x option
  is used.

OPTIONS
  -o  prints numbers in octal
  -x  prints numbers in hexadecimal
  -V  prints the version information

NOTE
  Both the Release 2R1 and Release 1R1 archive formats are sup­
  ported permitting transparent use with archive libraries from
  Release 1R1 on the 5000/20/40/50.

SEE ALSO
  as(1), cc(1), ld(1), a.out(4), ar(4).

DIAGNOSTICS
  size: name: cannot open
    name cannot be read.
  size: name: bad magic
    name is not an appropriate common object file.
NAME
sleep - suspend execution for an interval

SYNOPSIS
sleep time

DESCRIPTION
Sleep suspends execution for \textit{time} seconds. \textit{Sleep} executes a command after a certain amount of time:

\begin{verbatim}
(sleep 105; command)&
\end{verbatim}

or executes a command every so often:

\begin{verbatim}
while true
do
    command
    sleep 37
done
\end{verbatim}

SEE ALSO
alarm(2), sleep(3C).

RESTRICTIONS
\textit{Time} must be less than 65536 seconds. It is also recommended that \textit{time} be greater than 1.
NAME
  sln - link files symbolically

SYNOPSIS
  sln file1 [ file2 ...] target
  sln dir target

DESCRIPTION
  5000/20, 5000/30, 5000/35, 5000/40, 5000/50, and 5000/55 only.

  File1 is linked symbolically to target. Under no circumstance can
  file1 and target be the same (take care when using sh (1) meta­
  characters). If target is a directory, then one or more files are linked
  to that directory.

  If the sln command is invoked by the super-user then file1 may be a
  directory.

  Slm is similar to the ln(1) command, except for the following:
  - File1 and target can exist on different file systems.
  - If the target is removed after the sln(1) command has been
    issued, then any file access attempts will result in an error.
  - The files file1, file2, and dir must be full path names.

  The rm (1) command must be used on all the files that were
  remotely linked to target.

SEE ALSO
  ln(1), rm(1), unlink(1M), slink(2), link(1M).

RESTRICTIONS
  If file1 is a directory or has subsequently been removed then the
  unlink (1M) command must be invoked to remove the symbolic link.
  Due to changes in path when operating on directories, linked direc­
  tories may cause utilities such as find to return a bad status.
NAME
sno - SNOBOL interpreter

SYNOPSIS
sno [ files ]

DESCRIPTION
Sno is a compiler and interpreter very similar to SNOBOL. The slight differences are listed under DIFFERENCES.

Input to sno is the concatenation of the named files and the standard input. Sno considers input statements, up to and including the label end, a program, and compiles those statements. All other statements are available to syspit.

DIFFERENCES
Sno has no unanchored searches. To get the same effect:

a *** b
a *x* b = x c

Sno has no back referencing.

x = "abc"

Sno declares functions at compile time using the (non-unique) label define. Execution of a function call begins at the statement following the define. Functions cannot be defined at run time, and sno preempts the use of the name define. Sno does not provide for automatic variables other than parameters. Examples:

define f( )
define f(a, b, c)

All labels except define (even end) must have a non-empty statement.

Labels, functions and variables must all have distinct names. In particular, the non-empty statement on end cannot merely name a label.

If start is a label in the program, sno starts program execution there. If not, sno begins execution with the first executable statement; define is not an executable statement.

Sno has no built-in functions.

Sno does not need parentheses for arithmetic; normal precedence applies. Because of this, spaces must set off the arithmetic operators / and *.

The right side of assignments must be non-empty.

Either ' or " may be used for literal quotes.

The pseudo-variable sysppt is not available.

SEE ALSO
awk(1).
[This page left blank.]
NAME
sort - sort and/or merge files

SYNOPSIS
sort [-emu] [ooutput] [-ykmem] [-zrecsz] [-dfMNr] [-btX]
[+pos1 [-pos2]] [files]

DESCRIPTION
Sort sorts lines of all the named files together and writes the result
on the standard output. The standard input is read if - is used as
a file name or no input files are named.

Comparisons are based on one or more sort keys extracted from
each line of input. By default, there is one sort key, the entire
input line, and ordering is lexicographic by bytes in machine col-
lating sequence.

OPTION
The following options alter the default behavior:
-c Check that the input file is sorted according to the ordering
rules; give no output unless the file is out of sort.
-m Merge only, the input files are already sorted.
-u Unique: suppress all but one in each set of lines having equal
keys.
-ooutput
Use the argument given as the name of an output file instead of
the standard output. This file may be the same as one of the
inputs. There may be optional blanks between -o and output.

-ykmem
The amount of main memory used by the sort has a large impact
on its performance. Sorting a small file in a large amount of
memory is a waste. If this option is omitted, sort begins using
a system default memory size, and continues to use more space
as needed. If this option is presented with a value, kmem,
sort starts using that number of kilobytes of memory, unless
the administrative minimum or maximum is violated, in which
case the corresponding extremum is used. Thus, -y0 is
guaranteed to start with minimum memory. By convention, -y
(with no argument) starts with maximum memory.

-zrecsz
The size of the longest line read is recorded in the sort phase
so buffers can be allocated during the merge phase. If the
sort phase is omitted via the -c or -m option, a popular system
default size is used. Lines longer than the buffer size cause
sort to terminate abnormally. Supplying the actual number of
bytes in the longest line to be merged (or some larger value)
prevents abnormal termination.

The following options override the default ordering rules.
-d Dictionary order: only letters, digits and blanks (spaces and
tabs) are significant in comparisons.
-f Fold lower case letters into upper case.

-i Ignore characters outside the ASCII range 040-0176 in non-numeric comparisons.

-M Compare as months. The first three non-blank characters of the field are folded to upper case and compared so that JAN < FEB < ... < DEC. Invalid fields compare low to JAN. The -M option implies the -b option (see below).

-n An initial numeric string, consisting of optional blanks, optional minus sign, and zero or more digits with optional decimal point, is sorted by arithmetic value. The -n option implies the -b option (see below). Note that the -b option is only effective when restricted sort key specifications are in effect.

-r Reverse the sense of comparisons.

When ordering options appear before restricted sort key specifications, the requested ordering rules are applied globally to all sort keys. When attached to a specific sort key (described below), the specified ordering options override all global ordering options for that key.

The notation +pos1 -pos2 restricts a sort key to one beginning at pos1 and ending at pos2. The characters at positions pos1 and pos2 are included in the sort key provided that pos2 does not precede pos1. A missing -pos2 means the end of the line.

Specifying pos1 and pos2 involves the notion of a field, a minimal sequence of characters followed by a field separator or a new-line. By default, the first blank (space or tab) of a sequence of blanks acts as the field separator. All blanks in a sequence of blanks are considered to be part of the next field; for example, all blanks at the beginning of a line are considered to be part of the first field. The treatment of field separators can be altered using the options.

-tx Use x as the field separator character; x is not considered to be part of a field although it may be included in a sort key. Each occurrence of x is significant (e.g., xx delimits an empty field).

-b Ignore leading blanks when determining the starting and ending positions of a restricted sort key. If the -b option is specified before the first +pos1 argument, it is applied to all +pos1 arguments. Otherwise, the b option may be attached independently to each +pos1 or -pos2 argument (see below).

Pos1 and pos2 each have the form m.n optionally followed by one or more of the options bdfinr. A starting position specified by +m .n is interpreted to mean the n+1st character in the m+1st field. A missing .n means .0, indicating the first character of the m+1st field. If the b option is in effect n is counted from the first non-blank in the m+1st field; +m .0 b refers to the first non-blank character in the m+1st field.
A last position specified by \(-m.n\) is interpreted to mean the \(n\)th character including separators after the last character of the \(m\)th field. A missing \(n\) means .0, indicating the last character of the \(m\)th field. If the \(b\) option is in effect \(n\) is counted from the last leading blank in the \(m+1\)st field; \(-m.1\ b\) refers to the first non-blank in the \(m+1\)st field.

When there are multiple sort keys, later keys are compared only after all earlier keys compare equal. Lines that otherwise compare equal are ordered with all bytes significant.

**EXAMPLES**

Sort the contents of `infile` with the second field as the sort key:

```
sort +1 -2 infile
```

Sort, in reverse order, the contents of `infile1` and `infile2`, placing the output in `outfile` and using the first character of the second field as the sort key:

```
sort -r -o outfile +1.0 -1.2 infile1 infile2
```

Sort, in reverse order, the contents of `infile1` and `infile2` using the first non-blank character of the second field as the sort key:

```
sort -r +1.0b -1.1b infile1 infile2
```

Print the password file (`passwd (4)`) sorted by the numeric user ID (the third colon-separated field):

```
sort -t: +2n -3 /etc/passwd
```

Print the lines of the already sorted file `infile`, suppressing all but the first occurrence of lines having the same third field (the options `-um` with just one input file make the choice of a unique representative from a set of equal lines predictable):

```
sort -um +2 -3 infile
```

**FILES**

```
/usr/tmp/stm???
```

**SEE ALSO**

`comm(1)`, `join(1)`, `uniq(1)`.

**DIAGNOSTICS**

Comments and exits with non-zero status for various trouble conditions (e.g., when input lines are too long), and for disorder discovered under the `-c` option.

When the last line of an input file is missing a new-line character, `sort` appends one, prints a warning message, and continues.

**RESTRICTIONS**

`Sort` outputs files without truncation if the file has up to 46,380 lines of up to 1024 characters per line. `Sort` outputs files with truncation (and displays an error message) if the file exceeds 12,488 characters or 1561 lines and the line length exceed 1024 characters.
[This page left blank.]
NAME
spell, hashmake, spellin, hashcheck - find spelling errors

SYNOPSIS
spell [ -v ] [ -b ] [ -x ] [ -l ] [ -i ] [ +local_file ] [ files ]
/usr/lib/spell/compress (Not on the 5000/30 or 5000/50)
/usr/lib/spell/hashmake
/usr/lib/spell/spellin n
/usr/lib/spell/hashcheck spelling_list

DESCRIPTION
Spell collects words from the named files and looks them up in a
spelling list. Spell prints words on the standard output that do not
occur in the spelling list, and cannot be derived from words in that
list by applying certain inflections, prefixes, and/or suffixes. If
no files are named, spell collects words from the standard input.
Spell ignores most troff(1), tbl(1), and eqn(1) constructions.
Spell follows chains of included files (.so and .nx troff(1)
requests), unless the names of such included files begin with
/usr/lib.
The spelling list is based on many sources, and while it is less com-
plete than an ordinary dictionary in some ways, it is also more effec-
tive with respect to proper names and popular technical
words. Coverage of the specialized vocabularies of biology, medi-
cine, and chemistry is light.

Pertinent auxiliary files may be specified by name arguments indi-
cated below with their default settings (see FILES). Spell accumu-
lates copies of all output in the history file. Since identical entries
are often made in the history file because the same word is
misspelled during different executions of spell, compress should
be used to remove redundant entries. The compressed history file
will be smaller and easier to analyze. The stop list filters out
misspellings (e.g., thier=thy.y+ier) that would otherwise pass.

Three routines help maintain and check the hash lists used by
spell:

hashmake Reads a list of words from the standard input and
writes the corresponding nine-digit hash code on the
standard output.

spellin n Reads n sorted hash codes from the standard input and
writes a compressed spelling list on the standard out-
put. Information about the hash coding is printed on
standard error.

hashcheck Reads a compressed spelling_list and recreates the
nine-digit hash codes for all the words in it; hashcheck
writes these codes on the standard output.

OPTIONS
SPELL(1)

-v Print all words not literally in the spelling list, with plausible derivations from the words in the spelling list indicated.

-b Check British spelling, preferring centre, colour, programme, speciality, travelled, etc. This option also assumes that all ize suffixes should be spelled ise; this assumption is incorrect.

-x Print every plausible stem with = for each word.

-l Follow the chains of all files including those that begin with /usr/lib.

-i Ignore all chains of included files.

+local_file
Remove words found in local_file from the output of spell. Local_file is the name of a user-provided file that contains a sorted list of words, one per line. This option permits the user to specify a set of words that are correct spellings, in addition to the spelling list used by spell.

EXAMPLE
The following example creates the hashed spell list hlist and checks the result by comparing the two temporary files; they should be equal.

cat goodwds /usr/lib/spell/hashmake sort -u >tmp1
cat tmp1 /usr/lib/spell/spellin 'cat tmp1 wc -l' >hlist
cat hlist /usr/lib/spell/hashcheck >tmp2
diff tmpl tmp2

FILES
D_SPELL=/usr/lib/spell/hlist[ab] hashed spelling lists, American and British
S_SPELL=/usr/lib/spell/hstop hashed stop list
H_SPELL=/usr/lib/spell/spellhist history file
/usr/lib/spell/spellprog program

SEE ALSO
deroff(1), eqn(1), sed(1), sort(1), tbl(1), tee(1), troff(1).

RESTRICTIONS
The coverage of the spelling list is uneven; new installations will probably wish to monitor the output for several months to gather local additions; typically, these additions are kept in a separate local file that is added to the hashed spelling_list by spellin.
NAME
spline - interpolate smooth curve

SYNOPSIS
spline [ options ]

DESCRIPTION
Not on 7000/40.

Spline takes pairs of numbers from the standard input as abscissas and ordinates of a function. It produces a similar set, which is approximately equally spaced and includes the input set, on the standard output. The cubic spline output (R. W. Hamming, Numerical Methods for Scientists and Engineers, 2nd ed., pp. 349ff) has two continuous derivatives, and sufficiently many points to look smooth when plotted, for example by graph(1G).

OPTIONS
-a Supply abscissas automatically (they are missing from the input); spacing is given by the next argument, or is assumed to be 1 if next argument is not a number.
-k k Set constant k to k (default k is 0) for use in the boundary value computation:

\[ y_0'' = ky_1, \quad y_n'' = ky_{n-1} \]

-n n Space output points so that approximately n intervals occur between the lower and upper x limits (default n = 100).
-p Make output periodic, i.e., match derivatives at ends. First and last input values should normally agree.
-x low [ high ]
Denotes low as the lower x limit; denotes high (if given) as the upper x limit. Normally, these limits are calculated from the data. Automatic abscissas start at lower limit (default 0).

SEE ALSO
graph(1G).

DIAGNOSTICS
When data is not strictly monotone in x, spline reproduces the input without interpolating extra points.

RESTRICTIONS
A limit of 1,000 input points is enforced silently.
SPLINE(1G)

[This page left blank.]
NAME
split - split a file into pieces

SYNOPSIS
split [ -n ] [ -b ] [ file [ name ] ]

DESCRIPTION
Split reads file and writes it into a set of output files. The name of the first output file is name with aa appended, and so on lexicographically, up to zz, a maximum of 676 files. Name cannot be longer than 12 characters. If no output name is given, x is the default.

If no input file is given, or if - is given in its place, the standard input file is used.

OPTIONS
- n Number of pieces in each output file; the default is 1000.
- b Pieces are 512-byte blocks; the default is that pieces are lines.
   (5000/20/30/40/50 only)

EXAMPLE
To split the toobig file into 500 line files named smallaa, smallab, smallac, etc., enter
   split -n 500 toobig small

SEE ALSO
bfs(1), csplit(1).
[This page left blank.]
NAME
spool - spool queue manager

SYNOPSIS
spool command-option secondary-option ...

DESCRIPTION
Spool controls the spool queue after a file has been placed into the spool queue by print(1) or lpr for printing. Spool:
• Controls printing through the ability to purge, hold, restart, start, and stop the printer spooler system.
• Allows the user to query and modify the file entries in the spool queue.
• Allows update of certain control data by accessing the spool queue.

Only the originator of the file or the superuser may change data in the spool queue or view the spooled file.

COMMAND SYNTAX
The following table lists the spool command options and secondary options.

<table>
<thead>
<tr>
<th>Command</th>
<th>Command Options</th>
<th>Secondary Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>spool</td>
<td>-help</td>
<td></td>
</tr>
<tr>
<td>spool</td>
<td>-query</td>
<td></td>
</tr>
<tr>
<td>spool</td>
<td>-change</td>
<td></td>
</tr>
<tr>
<td>spool</td>
<td>-purge</td>
<td>-sp spool_id [-pt lpnn] [-pr xx] [-fm xxxxxx]</td>
</tr>
<tr>
<td>spool</td>
<td>-hold</td>
<td>-sp spool_id [-ws tnn] [-pt lpnn] [-who xxxxxxxx]</td>
</tr>
<tr>
<td>spool</td>
<td>-release</td>
<td>-sp spool_id [-ws tnn] [-pt lpnn] [-who xxxxxxxx]</td>
</tr>
<tr>
<td>spool</td>
<td>-start</td>
<td>-pt lpnn spool</td>
</tr>
<tr>
<td>spool</td>
<td>-look</td>
<td>-sp spool_id</td>
</tr>
<tr>
<td>spool</td>
<td>-restart</td>
<td>-pt lpnn [-pg nnn]</td>
</tr>
</tbody>
</table>

SECONDARY OPTIONS
Secondary options have the following meanings and values:
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-sp spool-id</td>
<td>Spool identification: spool-id is sf followed by six digits</td>
</tr>
<tr>
<td>-cp xxx</td>
<td>Number of copies: xxx is 1-999</td>
</tr>
<tr>
<td>-fm xxxxx</td>
<td>Form name: xxxxx is 1-5 alphanumeric characters</td>
</tr>
<tr>
<td>-pg nnnn</td>
<td>Page number: nnnn is 1-9999</td>
</tr>
<tr>
<td>-pr xx</td>
<td>Priority: xx is 0-15</td>
</tr>
<tr>
<td>-pt lpnn</td>
<td>Printer number: nn is 00-maximum number of printers allowed</td>
</tr>
<tr>
<td>-who xxxxxxxx</td>
<td>User name: xxxxxxxx is 1-8 characters</td>
</tr>
<tr>
<td>-ws tnn</td>
<td>Terminal number: nn is 00-15</td>
</tr>
</tbody>
</table>

COMMAND OPTIONS

-help
The help option displays the list of commands on the screen. No secondary options are used.

-query
The query option displays the current spool queue status. This display is screen oriented and is available to all users. No secondary options are used.

-change
The change option allows the user to change specified spooled file values. The entries in the spool queue can be modified by the superuser or the originating user. The various forms of this command are listed below:

    spool -change -sp spool_id -pt lp02
    Change the destination printer to lp02 for the file specified by spool_id.

    spool -change -sp spool_id -pr nn
    Change the print priority to nn for the file specified by spool_id.

    spool -change -sp spool_id -cp nn
    Change the number of copies to nn for the file specified by spool_id.

    spool -change -sp spool_id -fm xxxxx
    Change the forms name to xxxxx for the file specified by the spool_id.

-purge
The purge option allows the user to delete a spooled file from the spooler subsystem. This command functions only for the superuser or the originating user. The various forms of this command command are listed below:
spool -purge -sp spool-id
Purge the file specified by spool-id.

spool -purge -ws t01
Purge all files submitted from work station t01.

spool -purge -pt lp01
Purge all files queued to lp01.

spool -purge -who smith
Purge all files submitted by user smith.

-start
The start option starts printing on the specified printer. If the despooler is already writing to the printer, this command is ignored. The secondary option -pt indicates the printer. For example,

spool -start -pt lp01
starts printing with the first file on the spool queue for lp01.

-stop
The stop option stops printing on the specified printer and terminates the running despooler for the specified print device. The file which was being printed remains on the print queue and is resumed when an explicit spool start command is issued or a new print request is generated by the print or lpr command. The secondary option -pt indicates the printer. For example,

spool -stop -pt lp02
stops printing the file presently being printed on lp02.

-restart
The restart option restarts the printing of a file which is currently being printed at any page within the print file. If the specified page number is greater than the number of pages in the file, printing starts with page 1. The secondary option -pt indicates the printer. For example,

spool -restart -pt lp02
restarts printing at page 1 of the currently printing file on lp02.
The secondary option -pg indicates the page number. For example,

spool -restart -pt lp02 -pg 33
restarts printing at page 33 of the currently printing file on lp02.

-hold
The hold option allows a specific print file to be placed on hold by specifying the spool_id for the print file as follows:
spool -hold -sp spool_id

Also, a queue of files may be placed on hold by specifying the terminal work station, the print device, or the originating user name for the print file as shown in the following examples.

spool -hold -ws t02
Holds all files submitted from work station t02.

spool -hold -pt lp01
Holds all print files queued to be printed on lp01.

spool -hold -who smith
Holds all files originated by user smith.

-release
The release option allows a specific file to be released from hold and placed in a wait state by specifying the spool_id for the print file as follows:

spool -release -sp spool_id

Also, a group of print files may be released from hold and placed in a wait state by specifying the terminal work station, the destination print device, or the originator of the print file. The following examples define these options.

spool -release -ws t01
Releases all print files on hold for device t01.

spool -release -pt lp01
Releases all print files on hold for lp01.

spool -release -who smith
Releases all files on hold for user smith.

To print the released files, the start option of the spool command must be performed.

-look
The look option displays a spooled file to a terminal if the requesting user is the originator of the spooled file or the superuser. The secondary option -sp designates the spooled file.

FILES
/usr/spool/lpd/* spool area
/usr/spool/lpd/lpd despooler
/bin/print spooler
/bin/lpr spooler
/bin/spool spool queue manager
/usr/spool/lpd/spooldev spool device table manager
/usr/spool/lpd/??spldev spool device table
/usr/spool/lpd/??spilque spool queue
`/usr/spool/lpd/sf*` spooled files

**SEE ALSO**

`print(1), spooldev(1M)`.
[This page left blank.]
NAME
ssp - make output single spaced

SYNOPSIS
ssp [ name ... ]

DESCRIPTION
5000 Series only.
Ssp removes extra blank lines and causes all output to be single spaced. Ssp can be used directly, or as a filter after nroff or other text formatting operations.

EXAMPLE
nroff -ms filea fileb | ssp >> filec
prepares filea and fileb with the -ms macro package, then single spaces the output and directs it to filec.
[This page left blank.]
NAME
starter - information about the UNIX system for beginning users

SYNOPSIS
[ help ] starter

DESCRIPTION
5000/60, 5000/80, and 5000/90 only.

The UNIX system Help Facility command starter provides five categories of information about the UNIX system to assist new users.

The five categories are:
- commands a new user should learn first
- UNIX system documents important for beginners
- education centers offering UNIX system courses
- local environment information
- on-line teaching aids installed on the UNIX system

The user may choose one of the above categories by entering its corresponding letter (given in the menu), or may exit to the shell by typing q (for "quit"). When a category is chosen, the user will receive one or more pages of information pertaining to it.

From any screen in the Help Facility, a user may execute a command via the shell (sh(1)) by typing a ! and the command to be executed. The screen will be redrawn if the command that was executed was entered at a first level prompt. If entered at any other prompt level, only the prompt will be redrawn.

By default, the Help Facility scrolls the data that is presented to the user. If you prefer to have the screen clear before printing the data (non-scrolling), the shell variable SCROLL must be set to no and exported so it will become part of your environment. This is done by adding the following line to your .profile file (see profile(4)): "export SCROLL; SCROLL=no". If you later decide that scrolling is desired, SCROLL must be set to yes.

Information on each of the Help Facility commands (starter, locate, usage, glossary, and help) is located on their respective manual pages.

SEE ALSO
glossary(1), help(1), locate(1), sh(1), usage(1).

WARNINGS
If the shell variable TERM (see sh(1)) is not set in the user’s profile file, then TERM will default to the terminal value type 450 (a hard-copy terminal). For a list of valid terminal types, refer to term(5).
NAME
stat - statistical network useful with graphical commands

SYNOPSIS
node-name [options] [files]

DESCRIPTION
Not on 7000/40.

Stat is a collection of command level functions (nodes) that can be interconnected using sh(1) to form a statistical network. The nodes reside in /usr/bin/graf (see graphics(1G)). Data is passed through the network as sequences of numbers (vectors), where a number is of the form:

[sign] (digits) (. digits) [e [sign] digits]

evaluated in the usual way. Brackets and parentheses surround fields. All fields are optional, but at least one of the fields surrounded by parentheses must be present. Any character input to a node that is not part of a number is taken as a delimiter.

Stat nodes are divided into four classes.

Transformers, which map input vector elements into output vector elements;
Summarizers, which calculate statistics of a vector;
Translators, which convert among formats; and
Generators, which are sources of definable vectors.

Below is a list of synopses for stat nodes. Most nodes accept options indicated by a leading minus (-). In general, an option is specified by a character followed by a value, such as c5. This is interpreted as c = 5 (c is assigned 5). The following keys are used to designate the expected type of the value:

- c characters,
- i integer,
- f floating point or integer,
- file file name, and
- string string of characters, surrounded by quotes to include a shell argument delimiter.

Options without keys are flags. All nodes except generators accept files as input, hence it is not indicated in the synopses.

Transformers:

- abs [-ci] - absolute value
- columns (similarly for -c options that follow)
- af [-ci tv] - arithmetic function
titled output, verbose
- cell [-ci] - round up to next integer
- cusum [-ci] - cumulative sum
STAT(1G)

exp [-ci] - exponential
floor [-ci] - round down to next integer
gamma [-ci] - gamma
list [-ci dstring] - list vector elements
delimiter(s)
log [-ci bf] - logarithm
base
mod [-ci mf] - modulus
modulus
pair [-ci Ffile xi] - pair elements
File containing base vector, x group size
power [-ci pf] - raise to a power
power
root [-ci rf] - take a root
root
round [-ci pi si] - round to nearest integer, .5 rounds to 1
places after decimal point, significant digits
slline [-ci if nisf] - generate a line given slope and intercept
intercept, number of positive integers, slope
sin [-ci] - sine
subset [ -ff ni pf s ] - point from empirical cumulative density function
Summarizers:
bucket [-ai ci Ffile hf ii If ni] - break into buckets
average size, File containing bucket boundaries,
high, interval, low, number
Input data should be sorted
cor [-Ffile] - correlation coefficient
File containing base vector
hilo [-h l o ox oy] - find high and low values
high only, low only, option form, option form with x prepended, option form with y prepended
lreg [-Ffile io s] - linear regression
File containing base vector, intercept only, option form for slline, slope only
mean [-ff ni pf] - (trimmed) arithmetic mean
fraction, number, percent
point [-ff ni pf s] - point from empirical cumulative density function
fraction, number, percent, sorted input

prod - internal product
qsort [ -ci ] - quick sort
rank - vector rank
total - sum total
var - variance

Translators:

bar [ -a b f g ri wi xf xa yf ya ylf yhf ] - build a bar chart
suppress axes, bold, suppress frame, suppress grid,
region, width in percent, x origin, suppress x-axis label, y origin, suppress y-axis label, y-axis lower bound, y-axis high bound
Data is rounded off to integers.

hist [ -a b f g ri xf xa yf ya ylf yhf ] - build a histogram
suppress axes, bold, suppress frame, suppress grid,
region, x origin, suppress x-axis label, y origin, suppress y-axis label, y-axis lower bound, y-axis high bound

label [ -b c File h p ri x xu y yr ] - label the axis of a GPS file
bar chart input, retain case, label File, histogram input, plot input, rotation, x-axis, upper x-axis, y-axis, right y-axis

pie [ -b o p t p i ri v x i y i ] - build a pie chart
bold, values outside pie, value as percentage(=100),
value as percentage(=i), draw percent of pie,
region, no values, x origin, y origin
Unlike other nodes, input is lines of the form
[< i e f cc >] value [label]
ignore (do not draw) slice, explode slice, fill
slice, color slice c=( black, red, green, blue)

plot [ -a b cstring d f File g m ri xf xa xif xhf xlf xni xt yf ya yif yhf ylf yni yt ] - plot a graph
suppress axes, bold, plotting characters, dis­connected, suppress frame, File containing x vector,
suppress grid, mark points, region, x origin, suppress x-axis label, x interval, x high bound, x low bound, number of ticks on x-axis, suppress x-axis title, y origin, suppress y-axis label, y interval, y high bound, y low bound, number of ticks on y-axis, suppress y-axis title

title [ -b c lstring vstring ustring ] - title a vector or a GPS title bold, retain case, lower title, upper title, vector title
Generators:

- **gas** [ -ci if ni sf tf ] - generate additive sequence
  interval, number, start, terminate

- **prime** [ -ci hi li ni ] - generate prime numbers
  high, low, number

- **rand** [ -ci hf lf mf ni si ] - generate random sequence
  high, low, multiplier, number, seed

**RESTRICTIONS**

Some nodes have a limit on the size of the input vector.

**SEE ALSO**

- graphics(1G), gps(4).
NAME
   strings - find the printable strings in an object, or other binary, file

SYNOPSIS
   strings [ - ] [ -o ] [ -n u m b e r ] file . .

DESCRIPTION
   Strings looks for ascii strings in a binary file. A string is any
   sequence of 4 or more printing characters ending with a newline or
   a null. Strings only looks in the initialized data space of object
   files.

   Strings is useful for identifying random object files.

OPTIONS
   - examine uninitialized data space as well as initialized data space
   -o precede each string by its offset in the file (in octal)
   -n use n as the minimum string length, rather than 4

SEE ALSO
   od(1)

WARNING
   The algorithm for identifying strings is extremely primitive
NAME
strip - strip symbol and line number information from object file

SYNOPSIS
strip [-l] [-x] [-r] [-V] filename

DESCRIPTION
The strip command strips the symbol table and line number information from common object files, including archives. Once a file has been stripped, no symbolic debugging access is available for that file; therefore, strip is normally run only on production modules that have been debugged and tested. The purpose of this command is to reduce the file storage overhead taken by the object file.

If there are any relocation entries in the object file and any symbol table information is to be stripped, strip prints an error message and terminates without stripping filename unless the -r option is used.

If the strip command is executed on a common archive file (see ar(4)) the archive symbol table is removed. The archive symbol table must be restored by executing the ar(1) command with the -s option before the archive can be link edited by the ld(1) command. Strip(1) instructs the user with appropriate warning messages when this situation arises.

OPTIONS
The amount of information stripped from the symbol table can be controlled by using any of the following options:
- Strip line number information only; do not strip any symbol table information.
- x Do not strip static or external symbol information.
- r Reset the relocation indexes into the symbol table.
- V Print on the standard error output the version of the strip command which is executing.

NOTE
Both the Release 2R1 and Release 1R1 archive formats are supported permitting transparent use of archive libraries from Release 1R1 on the 5000/20/40/50.

FILES
/usr/tmp/strp??????

SEE ALSO
ar(1), as(1), cc(1), ld(1), ar(4), a.out(4).

DIAGNOSTICS
If filename cannot be read, strip prints
strip: name: cannot open

If filename is not an appropriate common object file, strip prints
strip: name: bad magic
If *filename* contains relocation entries and the `-r` option is not used, the symbol table information cannot be stripped; *strip* prints

```
strip: name: relocation entries present; cannot strip
```
NAME
stty - set the options for a terminal

SYNOPSIS
stty [ -a ] [ -g ] [ options ]

DESCRIPTION
Stty sets certain terminal I/O options for the device that is the current standard input; without arguments, it reports the settings of certain options; with the -a option, it reports all of the option settings; with the -g option, it reports current settings in a form that can be used as an argument to another stty command. Detailed information about the modes listed in the first five groups below may be found in termio(7). Options in the last group are implemented using options in the previous groups. Note that many combinations of options make no sense, but no sanity checking is performed. The options are selected from the following:

CONTROL MODES

parenb (-parenb) enable (disable) parity generation and detection.
parodd (-parodd) select odd (even) parity.

CS MODES

cs5 cs6 cs7 cs8 select character size (see termio(7)).
0 hang up phone line immediately.
50 75 110 134 150 200
300 600 1200 1800
2400 4800 9600 exta extb

set terminal baud rate to the number given, if possible. (All speeds are not supported by all hardware interfaces.)

hupcl (-hupcl) hang up (do not hang up) DATA-PHONE® connection on last close.
hup (-hup) same as hupcl (-hupcl).

cstopb (-cstopb) use two (one) stop bits per character.
cread (-cread) enable (disable) the receiver.
clocal (-clocal) assume a line without (with) modem control.
loblk (-loblk) block (do not block) output from a non-current layer.

INPUT MODES

ignbrk (-ignbrk) ignore (do not ignore) break on input.
brkint (-brkint) signal (do not signal) INTR on break.
ignpar (-ignpar) ignore (do not ignore) parity errors.
parmrk (-parmrk) mark (do not mark) parity errors (see termio(7)).

inpack (-inpack) enable (disable) input parity checking.
istrip (-istrip) strip (do not strip) input characters to seven bits.

inlcr (-inlcr) map (do not map) NL to CR on input.
igncr (-igncr) ignore (do not ignore) CR on input.
icrnl (-icrnl) map (do not map) CR to NL on input.
iucr (-iucr) map (do not map) upper-case alphabets to lower case on input.
STTY(1)

ixon (-ixon)  enable (disable) START/STOP output control. Output is stopped by sending an ASCII DC3 (CTRL-s) and started by sending an ASCII DC1 (CTRL-q).

ixany (-ixany)  allow any character (only DC1) to restart output.

ixoff (-ixoff)  request that the system send (not send) START/STOP characters when the input queue is nearly empty/full.

OUTPUT MODES

opost (-opost)  post-process output (do not post-process output; ignore all other output modes).

olcuc (-olcuc)  map (do not map) lower-case alphabetics to upper case on output.

onlcr (-onlcr)  map (do not map) NL to CR-NL on output.

ocrnl (-ocrnl)  map (do not map) CR to NL on output.

onocr (-onocr)  do not (do) output CRs at column zero.

onlret (-onlret)  on the terminal NL performs (does not perform) the CR function.

ofill (-ofill)  use fill characters (use timing) for delays.

ofdel (-ofdel)  fill characters are DELs (NULs).

cr0 cr1 cr2 cr3  select style of delay for carriage returns (see termio(7)).

nl0 nl1  select style of delay for line-feeds (see termio(7)).

tab0 tab1 tab2 tab3  select style of delay for horizontal tabs (see termio(7)).

bs0 bs1  select style of delay for backspaces (see termio(7)).

ff0 ff1  select style of delay for form-feeds (see termio(7)).

vt0 vt1  select style of delay for vertical tabs (see termio(7)).

LOCAL MODES

isig (-isig)  enable (disable) the checking of characters against the special control characters INTR, QUIT, and SWTCH.

icanon (-icanon)  enable (disable) canonical input (ERASE and KILL processing).

xcase (-xcase)  canonical (unprocessed) upper/lower-case presentation.

echo (-echo)  echo back (do not echo back) every character typed.

echoe (-echoe)  echo (do not echo) ERASE character as a backspace-space-backspace string. Note: this mode erases the ERASEd character on many CRT terminals; however, it does not keep track of column position and, as a result, may be confusing on escaped characters, tabs, and backspaces.
echok (-echok)
echo (do not echo) NL after KILL character.

lfkc (-lfkc)
the same as echok (-echok); obsolete.

echonl (-echonl)
echo (do not echo) NL.

nofish (-nofish)
disable (enable) flush after INTR, QUIT, or SWTCH.

stwrap (-stwrap)
disable (enable) truncation of lines longer than 79 characters on a synchronous line.

stflush (-stflush)
enable (disable) flush on a synchronous line after every write(2).

stappl (-stappl)
use application mode (use line mode) on a synchronous line.

CONTROL ASSIGNMENTS

control-character
set control-character to c, where control-character is erase, kill, intr, quit, swtch, eof, ctab, min, or time (ctab is used with -stappl; (min and time are used with -icanon (see termio(7)). If c is preceded by an (escaped from the shell) caret (^), then the value used is the corresponding CTRL character (e.g., ^d is a CTRL-d); ^? is interpreted as DEL and ^- is interpreted as undefined.

line i
set line discipline to i (0 < i < 127).

tdcd s
set connect timer to s seconds. Setting s to zero disables the timer. (5000/90 only)

tact s
set inactivity timer to s seconds. Setting s to zero disables the timer. (5000/90 only)

COMBINATION MODES

evenp or parity
enable parenb and cs7.

oddp
enable parenb, cs7, and parodd.

-parity, -evenp, or -oddp
disable parenb, and set cs8.

raw (-raw or cooked)
enable (disable) raw input and output (no ERASE, KILL, INTR, QUIT, SWTCH, EOT, or output post processing).

nl (-nl)
unset (set) icrnl, onlcr. In addition -nl unsets inlcr, igncr, ocrln, and onlret.

lcase (-lcase)
set (unset) xcase, iuclc, and ocuc.

LCASE (-LCASE)
same as lcase (-lcase).

tabs (-tabs or tab3)
preserve (expand to spaces) tabs when printing.

ek
reset ERASE and KILL characters back to normal # and @.

sane
resets all modes to some reasonable values;
term

set all modes suitable for the terminal type

term Traditionaly, term has been one of tty33, tty37, vt05, tn300, ti700, or tak.

EXAMPLE
To display options, stty

to display options for tty01

stty </dev/tty01

SEE ALSO
tabs(1), ioctl(2), termio(7).
NAME
su - become superuser or another user

SYNOPSIS
su [ - ] [ name [ arg . . . ] ]

DESCRIPTION
Su allows one to become another user without logging off. The
default user name is root (i.e., superuser).

To use su, the appropriate password must be supplied (unless one
is already root). If the password is correct, su executes a new
shell with the real and effective user ID set to that of the specified
user. The new shell is the optional program named in the shell
field of the specified user password file entry (see passwd(4)), or
/bin/sh if none is specified (see sh(1)). To restore normal user ID
privileges, type an EOF (cntrl-d) to the new shell.

Any additional arguments given on the command line are passed to
the program invoked as the shell. When using programs like
sh(1), an arg of the form -c string executes string via the shell
and an arg of -r gives the user a restricted shell.

The following statements are true only if the optional program
named in the shell field of the specified user password file entry is
like sh(1). If the first argument to su is a -, the environment is
changed to what would be expected if the user actually logged in as
the specified user. This is done by invoking the program used as
the shell with an arg0 value whose first character is -, thus caus­
ing first the system profile (/etc/profile) and then the specified
user profile (.profile in the new HOME directory) to be executed.
Otherwise, the environment is passed along with the possible
exception of $PATH, which is set to /bin:/etc:/usr/bin for root.

Note that if the optional program used as the shell is /bin/sh, the
user .profile can check arg0 for -sh or -su to determine
if it was invoked by login(1) or su(1), respectively. If the user program is
other than /bin/sh, then .profile is invoked with an arg0 of -pro­
gram by both login(1) and su(1).

All attempts to become another user using su are logged in the log
file /usr/adm/sulog.

EXAMPLES
To become user bin while retaining your previously exported
environment, execute:

    su bin

To become user bin but change the environment to what would be
expected if bin had originally logged in, execute:

    su - bin

To execute command with the temporary environment and permis­
sions of user bin, execute:
su - bin -c "command args"

FILES
/etc/passwd system password file
/etc/profile system profile
$HOME/.profile user profile
/usr/adm/sulog log file

SEE ALSO
env(1), login(1), sh(1), passwd(4), profile(4), environ(5).
NAME
sum - print checksum and block count of a file

SYNOPSIS
sum [-r] file

DESCRIPTION
Sum calculates and prints a 16-bit checksum for the named file, and also prints the number of blocks in the file. Sum is typically used to look for bad sections of a file, or to validate a file communicated over some transmission line.

OPTIONS
-r use an alternate algorithm in computing the checksum.

SEE ALSO
wc(1).

DIAGNOSTICS
Read error
means end of file for most devices; check the block count to determine if an actual read error occurred.
SUM(1)

[This page left blank.]
NAME
sync - update the super block

SYNOPSIS
sync

DESCRIPTION
Sync executes the sync system primitive.
If the system is to be stopped, sync must be called to insure file
system integrity. Sync flushes all previously unwritten system
buffers out to disk, thus assuring that all file modifications up to
that point will be saved. See sync(2) for details.

SEE ALSO
sync(2).
[This page left blank.]
NAME
sz, sb - XMODEM, YMODEM, ZMODEM batch file send

SYNOPSIS
sz [ - +labdefklNnopqTtuvy ] file ...
sb [ -iadfkqtuv ] file ...
sz -X [ - 1kqtuv ] file
sz [ - loqtv ] -c COMMAND
sz [ - loqtv ] -i COMMAND

DESCRIPTION
This command is available with the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00 and greater.

Sz uses the ZMODEM, YMODEM or XMODEM error correcting protocol to send one or more files using the cu(1C) command over a serial port to a variety of programs running under PC-DOS, CP/M, UNIX, VMS, and other operating systems.

The first form of sz sends one or more files with ZMODEM or YMODEM batch protocol. Normally, only the file name part of the pathname is transmitted. On UNIX systems, additional information about the file is transmitted. If the receiving program uses this information, the transmitted file length controls the exact number of bytes written to the output dataset, and the modify time and file mode are set accordingly.

When logged in through cu, output from another program may be piped to sz for transmission by denoting standard input by "-":
   ps -ef | sz -
   ~?rz -1

The program output is transmitted with the filename sPID.sz where PID is the process ID of the sz program. If the environment variable ONAME is set, that is used instead. In this case, the UNIX commands:
   ONAME=output.file
   export ONAME
   ps -ef|sz -y -
   ~?rz -1

sends the file named output.file, which contains the output of the ps command, to the local system. The -y option instructs the receiver to open the file for writing unconditionally.

As an alias of the sz command, UNIX sb supports YMODEM-g with "cbreak" tty mode, XON/XOFF flow control, and the interrupt character set to CAN. YMODEM-g increases throughput over error free channels (direct connection, X,PC, etc.) by not acknowledging each transmitted sector.

The second form of sz uses the -X flag to send a single file with XMODEM or XMODEM-1k protocol. The user must supply the file name to both sending and receiving programs.
If `sz` is invoked with `$SHELL` set and if that variable contains the string `rsh` or `rksh` (restricted shell), `sz` operates in restricted mode. Restricted mode restricts pathnames to the current directory and `PUBDIR` (usually `/usr/spool/uucppublic`) and/or subdirectories thereof.

The third form sends a single `COMMAND` to the receiver for execution. `Sz` exits with the `COMMAND` return value. If `COMMAND` includes spaces or characters special to the shell, it must be quoted.

The fourth form sends a single `COMMAND` to the receiver for execution. `Sz` exits as soon as the receiver has correctly received the command, before it is performed.

If `sz` is invoked with `stdout` and `stderr` to different datasets, `verbose` is set to 2, causing frame by frame progress reports to `stderr`. This may be disabled with the `q` option.

**OPTIONS**

+ Instruct the receiver to append transmitted data to an existing file (ZMODEM only).
  
1 Use file descriptor 1 for `ioctl`s and `read`s (UNIX only). By default, file descriptor 0 is used. This option permits `sz` to be used with the `cu` "? command. `Cu` disables the separate process which reads characters from the modem, so that `rz` operates properly.

a Convert NL characters in the transmitted file to CR/LF. This is done by the sender for XMODEM and YMODEM, by the receiver for ZMODEM.

b (ZMODEM) Binary override: transfer file without any translation.

c  **COMMANDs+1**
   
   (ZMODEM) Send `COMMAND` to the receiver for execution, return with exit status for `COMMAND`.

   d (YMODEM/ZMODEM) Change all instances of "." to "/" in the transmitted pathname. Thus, `C/fileB0000` (which is unacceptable to MSDOS or CP/M) is transmitted as `C/fileB0000`. If the resultant filename has more than 8 characters in the stem, a "." is inserted to permit a total of eleven.

   e (ZMODEM) Escape all control characters; normally XON, XOFF, CR-@-CR, and Ctrl-X are escaped.

   f (YMODEM/ZMODEM) Send full pathname. Normally directory prefixes are stripped from the transmitted filename.

   i **COMMAND**
      
      (ZMODEM) Send `COMMAND` to the receiver for execution, return immediately upon the receiving program's successful reception of the command.

   k (YMODEM) Send files using 1024 byte blocks rather than the default 128 byte blocks (XMODEM/YMODEM). 1024 byte packets speed file transfers at high bit rates. (ZMODEM streams the data for the best possible throughput.)

   LN
      
      Use ZMODEM sub-packets of length N. A larger N (32 <= N <=
1024) gives slightly higher throughput, a smaller N speeds error recovery. The default is 128 below 300 baud, 256 above 300 baud, or 1024 above 2400 baud.

I N (ZMODEM) Wait for the receiver to acknowledge correct data every N (32 <= N <= 1024) characters. This may be used to avoid network overrun when XOFF flow control is lacking.

n (ZMODEM) Send each file if destination file does not exist. Overwrite destination file if source file is newer or longer than the destination file.

N (ZMODEM) Send each file if destination file does not exist. Overwrite destination file if source file has different length or date.

o (ZMODEM) Disable automatic selection of 32 bit CRC.

p (ZMODEM) Protect existing destination files by skipping transfer if the destination file exists.

q Quiet suppresses verbosity.

r Resume interrupted file transfer. If the source file is longer than the destination file, the transfer commences at the offset in the source file that equals the length of the destination file.

t tim Change timeout to tim tenths of seconds.

u Unlink the file after successful transmission.

v Verbose causes a list of file names to be appended to /tmp/szlog. More v's generate more output.

X Send a single file with XMODEM or XMODEM-1k protocol.

y Instruct a ZMODEM receiving program to overwrite any existing file with the same name.

EXEMPLARY
To upload a file via XMODEM protocol while logged onto a bulletin board through cu, first issue the bulletin board command to accept an upload then enter:

```sh
~?sz -X -lf filename
```

This is the equivalent of the built-in command:

```sh
~%upld filename
```

To send a file via XMODEM protocol while logged onto a remote system through cu, enter:

```sh
rz -b filename
~?sz -X -lf filename
```

This is the equivalent of the built-in command:

```sh
~%putx filename
```

To send any number of files using ZMODEM protocol while logged onto a remote system through cu, enter:

```sh
rz -b
~?sz -lfy files ...
```

This is the equivalent of the built-in command:

```sh
~%putz files ...
```
FILES
   /tmp/szlog Stores debugging output from "sz -vv"

SEE ALSO
   rz(1), cu(1C).

Compile time options required for various operating systems are described in the source file.

RESTRICTIONS
   XMODEM transfers add up to 127 garbage bytes per file (1023 bytes with XMODEM-k). Most YMODEM programs use the file length transmitted at the beginning of the transfer to prune the file to the correct length; this may cause problems with source files that grow during the course of the transfer. This problem does not pertain to ZMODEM transfers, which preserve the exact file length unconditionally.

   Most ZMODEM options are merely passed to the receiving program; some do not implement all these options.

   Circular buffering and a ZMODEM sliding window should be used when input is from pipes instead of acknowledging frames each 1024 bytes. If no files can be opened, sz sends a ZMODEM command to echo a suitable complaint; perhaps it should check for the presence of at least one accessible file before getting hot and bothered. The test mode leaves a zero length file on the receiving system.

   Some high speed modems have a firmware bug that drops characters when the direction of high speed transmission is reversed. The environment variable ZNULLS may be used to specify the number of nulls to send before a ZDATA frame. Values of 101 for a 4.77 mHz PC and 124 for an AT are typical.

   Improperly specified options and failing file transfers may leave the terminal in an unpredictable state.

   Some versions of UNIX cu(1) do not operate properly with this program.
NAME
  tabs - set tabs on a terminal

SYNOPSIS
  tabs [ tabspec ] [ +mn ] [ -Ttype ]

DESCRIPTION
  Tabs sets the tab stops on the terminal according to the tab specification tabspec, after clearing any previous settings. The user terminal must have remotely-settable hardware tabs.

  GE TermiNet terminals behave in a different way than most other terminals for some tab settings: the first number in a list of tab settings becomes the left margin on a TermiNet terminal. Thus, any list of tab numbers whose first element is other than 1 sets the left margin on a TermiNet, but not on other terminals. A tab list beginning with 1 has the same effect on all terminals.

  Setting the left margin is possible on some other terminals (see below).

  Tabs usually must know the type of terminal in order to set tabs; tabs always must know the terminal type in order to set margins. This type may be specified using the -T option (see below), but if no -T option is specified, tabs searches for the $TERM value in the environment (see environ(5)). If no type can be found, tabs tries a sequence that works for many terminals.

  Tabs sets the tabs and margins using the standard output.

TAB SPECIFICATIONS
  Tabs accepts four types of tab specification for tabspec: standard, repetitive, arbitrary, and file. If no tabspec is given, the default value is -8, (the UNIX system standard tabs) and the lowest column number is 1.

  Note that for tabs, column 1 always refers to the leftmost column on a terminal, even one whose column markers begin at 0, e.g., the DASI 300, DASI 300s, and DASI 450.

  The following tabspecs invoke standard tabs suited for particular languages:

    -a  1,10,16,36,72
        Assembler, IBM S/370, first format

    -a2 1,10,16,40,72
        Assembler, IBM S/370, second format

    -c  1,8,12,16,20,55
        COBOL, normal format

    -c2 1,6,10,14,49
        COBOL compact format (columns 1-6 omitted). Using this code, the first typed character corresponds to card column 7, one space gets you to column 8, and a tab reaches column 12. Files using this tab setup should include a format specification as follows:

        <:t-c2 m6 s66 d:>
TABS(1)

-c3 1,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62,67
COBOL compact format (columns 1-6 omitted), with more tabs than -c2. -c3 is the recommended format for COBOL. Files using this tab setup should include a format specification as follows:

   <:t-c3 m6 s66 d:>

-f 1,7,11,15,19,23
FORTRAN
-p 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
PL/I
-s 1,10,55
SNOBOL
-u 1,12,20,44
UNIVAC 1100 Assembler

In addition to these standard formats, three other types exist:

-n A repetitive specification requests tabs at columns 1+n, 1+2*n, etc. Note that such a setting leaves a left margin of n columns on TermiNet terminals only. Of particular importance is the value -8: this represents the UNIX system standard tab setting, and is the most likely tab setting to be found at a terminal. This setting is required for use with the nroff(1) -h option for high-speed output. Another special case is the value -0, implying no tabs at all.

n1,n2,...
The arbitrary format permits the user to specify any chosen set of numbers, separated by commas, in ascending order. Up to 40 numbers are allowed. If any number (except the first one) is preceded by a plus sign, the number is assumed to be an increment to be added to the previous value. Thus, the tab lists 1,10,20,30 and 1,10,+,10,+,10 are considered identical.

--file If the name of a file is given, tabs reads the first line of the file. If tabs finds a format specification on the line, tabs sets the tab stops according to the specification; otherwise tabs sets the tabs as -8.

This specification may be used with the pr(1) command to assure that a tabbed file is printed with correct tab settings:

tabs -- file; pr file

OPTIONS
Any of the following may be used in addition to the tab specification; if a given option occurs more than once, the last value specified takes effect:

-Ttype
Denotes type as the terminal type, where type is a name listed in term(5).
TABS(1)

+m n Moves all tabs over n columns by making column n+1 the left margin. If +m is given without a value of n, the value assumed is 10. For a TermNet, the first value in the tab list should be 1, or the margin moves even further to the right. The normal (leftmost) margin on most terminals is obtained by +m0. The margin for most terminals is reset only when the +m option is given explicitly.

DIAGNOSTICS
illegal tabs when arbitrary tabs are ordered incorrectly.
illegal increment when tabs finds a zero or missing increment in an arbitrary specification.
unknown tab code when a standard code cannot be found.
cannot open when --file option is used, and tabs cannot open the file.
file indirection when --file option is used and the specification in that file points to another file.

SEE ALSO
nroff(1), pr(1), environ(5), term(5).

RESTRICTIONS
The methods for clearing tabs and setting the left margin are inconsistent among different terminals. Usually the left margin cannot be changed without also setting tabs.

Tabs clears only 20 tabs (on terminals requiring a long sequence), but sets 64.

The +m option is not supported for any Unisys terminals as well as some of the terminals listed in term(5).

The set tabs option is supported only for the vtxxx terminals.

The set tabs option is valid for any terminal which permits hard tab settings and has hts correctly defined in its terminfo table.

The tabs command has no effect on any terminal unless tab filtering is disabled with the stty tab0 command. (When adding a terminal using the system administrator menus, the terminal has tabs filtered by default.)
[This page left blank.]
NAME
tail - deliver the last part of a file

SYNOPSIS
tail [ +: - [ number ] [ lbc [ f ] ] [ file ]

DESCRIPTION
Tail copies the named file to the standard output beginning at a
designated place. If no file is named, the standard input is used.

Copying begins at distance +number from the beginning, or
-number from the end of the input (if number is null, the value +10
is assumed). Number is counted in units of lines, blocks, or char-
acters, according to the appended option l, b, or c. When no units
are specified, counting is by lines.

With the -f (follow) option, if the input file is not a pipe, the pro-
gram does not terminate after the line of the input file has been
copied, but enters an endless loop, wherein it sleeps for a second
and then attempts to read and copy further records from the input
file. Thus tail may be used to monitor the growth of a file that is
being written by some other process.

EXAMPLES
tail -f carter

prints the last ten lines of the file carter, followed by any lines
that are appended to carter between the time tail is initiated and
the time tail is killed. As another example, the command:

tail -15cf thompson

prints the last 15 characters of the file thompson, followed by any
lines that are appended to thompson between the time tail is ini-
tiated and the time tail is killed.

SEE ALSO
dd(1).

RESTRICTIONS
Tails relative to the end of the file are stored in a buffer, and thus
are limited in length.

WARNING
Tail may produce irregular output when input includes character
special files. Tail only tails the last 4096 bytes of a file regardless
of its line count.
TAIL(1)

[This page left blank.]
NAME
tape_size - print the logical tape size to standard out

SYNOPSIS
tape_size tape_device

DESCRIPTION
5000/30 and 5000/50 only.
Tape_size prints the logical size, in bytes, of the tape_device to
the standard output. The logical size is the size stored in the
driver for the tape_device.
The default value for the cartridge tape device driver is 40960000
bytes, approximately the length of a DC450A cartridge tape.

EXAMPLE
To print the logical tape length of the device driver for /dev/rtp,
insert a cartridge tape in the tape drive and enter the command:

    tape_size /dev/rtp

The output for the default logical tape length is:

    tape capacity = 4096000

SEE ALSO
    set_tape(1)
[This page left blank.]
NAME
tar - tape file archiver

SYNOPSIS
tar [ options ] files

DESCRIPTION
5000 Series only

Tar saves and restores files as though the files were on magnetic or
streaming tape. Its actions are controlled by options. Note that
tar normally functions silently.

Options designates a string of characters containing at most one
function letter and possibly one or more function modifiers.

Other arguments to the command are files (or directory names)
specifying which files are to be saved or restored. In all cases,
appearance of a directory name refers to the files and (recur-
sively) subdirectories of that directory.

FUNCTION LETTERS
r Write the named files to the end of the tape. The c function
implies this function. This option is valid only for disk
archives.

x Extract the named files from the tape. If a named file specifies
a directory whose contents have been written onto the tape, tar
recursively extracts this directory. If the named file on tape
does not exist on the system, tar creates the file with the same
mode as the one on tape except that the set-user-ID bit and the
set-group-ID bit are not set unless you are the superuser. If
the files exist, their modes are not changed except for the bits
described above. Tar restores the owner, modification time,
and mode (if possible). If no files are specified, tar extracts
the entire content of the tape. Note that if several files with
the same name are on the tape, the last one overwrites all ear-
lier ones.

List the names of the specified files each time that they occur on
the tape. If no files are specified, tar lists all the names on the
tape.

u Add the named files to the tape if they are not already there or
have been modified since last written on that tape. This option
is only valid for disk archives.

c Create a new tape; writing begins at the beginning of the tape,
instead of after the last file. This function implies the r func-
tion.

FUNCTION MODIFIERS
The following characters may be used in addition to the letter that
selects the desired function:

#s Select the drive on which the tape is mounted and the density.
# is the tape drive number (0-7) and s is the density: l - low
(800 bpi), m - medium (1600 bpi), or h - high (6250 bpi). The
default is 0m. This option is only valid for 9-track magnetic
tapes on 5000/30/35/50/55 systems.
v Type the name of each file `tar` processes preceded by the function letter. When used with the t function, v gives more information about the tape entries than just the name.

w Print the action to be taken, followed by the name of the file, and then wait for the confirmation from the user. If a word beginning with y is entered, `tar` performs the action. Any other input cancels the action.

f archive
Use `archive` as the name of the archive instead of `/dev/rstp/0yy`. If `archive` is -, `tar` writes to the standard output or reads from the standard input, whichever is appropriate. Thus, `tar` can be used as the head or tail of a pipeline. `Tar` can also be used to move hierarchies with the command:

```
cd fromdir; tar cf -. (cd todir; tar xf -.)
```

b Use the next argument as the blocking factor for tape records. The default is 1; the maximum is 20. This option should only be used with raw archives (see -f above). The block size is determined automatically when reading tapes (function letters x and t).

l Print an error message if `tar` cannot resolve all of the links to the files being saved. If l is not specified, no error messages are printed.

m Do not restore the modification times. The modification time of the file is the time of extraction.

o Cause extracted files to take on the user and group identifier of the user running `tar` rather than those on tape.

**EXAMPLE**
To preserve ownership, modification date, and permissions over a uucp(1) communication, create an archive file and communicate it:

```
cd /usr/src/xxx
    tar cf /tmp/xxx.tar
    uucp /tmp/xxx.tar remote! username
```

To restore the archived files on remote:

```
cd /usr/src/xxx
    tar xvf /usr/spool/uucppublic/username/xxx.tar
```

**FILES**

```
/dev/rstp/*
/tmp/tar*
```

**DIAGNOSTICS**

Messages for bad function letters and modifiers.
Messages for tape read/write errors.
Error messages if insufficient memory is available to hold the link tables.

**RESTRICTIONS**

There is no way to ask for the n-th occurrence of a file.
`Tar` does not handle errors gracefully.
The u option can be slow.
The b option should not be used with archives on tape that are
going to be updated. The `b` option should not be used with archives on disk because updating an archive on disk can destroy it.
The limit on file-name length is 100 characters.
Note that `tar c0m` is not the same as `tar cm0`. 
[This page left blank.]
NAME

tbl - format tables for nroff or troff

SYNOPSIS

tbl [ -TX ] [ files ]

DESCRIPTION

Not on 7000/40.

Tbl is a preprocessor that formats tables for nroff or troff(1). Tbl copies the input files to the standard output, except for lines between .TS and .TE command lines, which are assumed to describe tables and are re-formatted by tbl. (The .TS and .TE command lines are not altered by tbl).

.TS is followed by global options. The global options, if any, are terminated with a semi-colon (;).

Next come lines describing the format of each line of the table. Each such format line describes one line of the actual table, except the last format line, which describes all remaining lines of the table. The last format line must end with a period. Each column of each line of the table is described by a single format letter, optionally followed by format specifiers that determine the font and point size of the corresponding item, that indicate where vertical bars are to appear between columns, that determine column width, inter-column spacing, etc.

The format lines are followed by lines containing the actual data for the table, followed by .TE. Within such data lines, data items are normally separated by tab characters.

If a data line consists of only _ or =, a single or double line, respectively, is drawn across the table at that point; if a single item in a data line consists of only _ or =, then that item is replaced by a single or double line.

Full details of all these and other features of tbl are given in the reference manual cited below.

The -TX option forces tbl to use only full vertical line motions, making the output more suitable for devices that cannot generate partial vertical line motions (e.g., line printers).

If no file names are given as arguments (or if - is specified as the last argument), tbl reads the standard input, thus tbl may be used as a filter. When it is used with eqn(1) or neqn, tbl should come first to minimize the volume of data passed through pipes.

GLOBAL OPTIONS

center centers the table (default is left-adjust).
expand makes the table as wide as the current line length.
box encloses the table in a box.
doublebox encloses the table in a double box.
allbox encloses each item of the table in a box.
tab(x) uses the character x instead of a tab to separate items in a line of input data.
## FORMAT LETTERS
- `c` center item within the column;
- `r` right-adjust item within the column;
- `l` left-adjust item within the column;
- `n` numerically adjust item in the column: units positions of numbers are aligned vertically;
- `s` span previous item on the left into this column;
- `a` center longest line in this column and then left-adjust all other lines in this column with respect to that centered line;
- `_` span down previous entry in this column;
- `=` replace this entry with a horizontal line;
- `|` replace this entry with a double horizontal line.

## FORMAT SPECIFIERS
- **B** Bold font
- **I** Italic font
- `|` Vertical line between columns

### EXAMPLE
The input (if `@` represents a tab which should be typed as a genuine tab):
```
.TR
center tab(@);
ccs
s
l
.HOUSEHOLD POPULATION

-Bedminster@789@3.26
-Bernards Twp.@3087@3.74
-Bernardsville@2018@3.30
-Bound Brook@3425@3.04
-Bridgewater@7897@3.81
-Far Hills@240@3.19
.TE
```
yields:

<table>
<thead>
<tr>
<th>Town</th>
<th>Households</th>
<th>Number</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bedminster</td>
<td>789</td>
<td>3.26</td>
<td></td>
</tr>
<tr>
<td>Bernards Twp.</td>
<td>3087</td>
<td>3.74</td>
<td></td>
</tr>
<tr>
<td>Bernardsville</td>
<td>2018</td>
<td>3.30</td>
<td></td>
</tr>
<tr>
<td>Bound Brook</td>
<td>3425</td>
<td>3.04</td>
<td></td>
</tr>
<tr>
<td>Bridgewater</td>
<td>7897</td>
<td>3.81</td>
<td></td>
</tr>
<tr>
<td>Far Hills</td>
<td>240</td>
<td>3.19</td>
<td></td>
</tr>
</tbody>
</table>
SEE ALSO
   cw(1), eqn(1), mm(1), mmt(1), nroff(1), mm(5), mv(5).

RESTRICTIONS
   See RESTRICTIONS under nroff(1).

   Note: Some printers or terminals may not be capable of performing all of the global options (e.g., box) or formatting features (e.g., line positioning or font styles) of tbl(1).
[This page left blank.]
NAME
tc - phototypesetter simulator

SYNOPSIS
tc [ -t ] [ -sn ] [ -pl ] [ file ]

DESCRIPTION
Not on 7000/40.
Tc interprets its input (standard input default) as device codes for
a Wang Laboratories, Inc. C/A/T phototypesetter. The standard
output of tc is intended for a Tektronix 4014 terminal with ASCII
and APL character sets. The sixteen typesetter sizes are mapped
into the four sizes supported on the 4014 terminal; the entire
TROFF character set is drawn using the character generator in the
4014, with overstruck combinations where necessary. Typical
usage is:

    troff -t files | tc

At the end of each page, tc waits for a new-line (empty line) from
the keyboard before continuing on to the next page. In this wait
state, three commands can be entered:

    e  suppresses the screen erase before the next page
    sn  skips the next n pages to be skipped
    !cmd  sends cmd to the shell.

OPTIONS
-t  Do not wait between pages (for directing output into a file).
-sn  Skip the first n pages.
-pl  Set page length to l; l may include the scale factors p (points),
i (inches), c (centimeters), and P (picas); default is picas.

SEE ALSO
4014(1), sh(1), tplot(1G), troff(1).

RESTRICTIONS
Font distinctions are lost.
[This page left blank.]
NAME
tee - copy input to standard output and to files

SYNOPSIS
tee [ -i ] [ -a ] [ file ] ...

DESCRIPTION
Tee copies the standard input to the standard output and makes copies in the files overwriting their previous contents.

OPTIONS
- i Ignore interrupts.
- a Append the output to the files rather than overwriting them.

EXAMPLE
To print the nroff(1) format of filex and save the formatted file in filex.nro:

    nroff filex; tee filex.nro; lpr
NAME
tension - tension a cartridge tape

SYNOPSIS
tension [ cartridge_device_name [ -Size sizefile ] ]
       (5000/20/30/40/50)

tension [ cartridge_device_name ] (5000/60/80/90)

DESCRIPTION
This utility uniformly tensions a cartridge tape. In the process,
the tape is fully rewound, advanced to the end of tape, then again
rewound. This reduces the potential for read errors on new
tapes, tapes that have been in storage for extended periods of
time, and on tapes that have been subjected to physical or thermal
shock. If no cartridge device name is specified as an argument,
/dev/rmt1 is used on the 5000/60/80/90 and /dev/rtp is used on the
5000/20/30/40/50. Tension will fail if the argument given is not a
cartridge device, or if a tape is not inserted in the cartridge tape
drive.

EXAMPLES
tension
tension /dw/rtp/capacity (5000/20/30/40/50 only)

SEE ALSO
erase(1) ioctl(2)
NAME
test - condition evaluation command

SYNOPSIS
test expr
    [ expr ]

DESCRIPTION
Test evaluates the expression expr and, if its value is true, returns a zero (true) exit status; otherwise, test returns a non-zero (false) exit status; test also returns a non-zero exit status if there are no arguments.

Notice that all the operators and primitives are separate arguments to test.

PRIMITIVES
The following primitives are used to construct expr:
- r file true if file exists and is readable.
- w file true if file exists and is writable.
- x file true if file exists and is executable.
- f file true if file exists and is a regular file.
- d file true if file exists and is a directory.
- c file true if file exists and is a character special file.
- b file true if file exists and is a block special file.
- p file true if file exists and is a named pipe (fifo).
- u file true if file exists and its set-user-ID bit is set.
- g file true if file exists and its set-group-ID bit is set.
- k file true if file exists and its sticky bit is set.
- s file true if file exists and has a size greater than zero.
- t [ fildes ]
  true if the open file whose file descriptor number is fildes (1 by default) is associated with a terminal device.
- z s1 true if the length of string s1 is zero.
- n s1 true if the length of the string s1 is non-zero.
- s1 = s2 true if strings s1 and s2 are identical.
- s1 != s2 true if strings s1 and s2 are not identical.
- s1 true if s1 is not the null string.
- n1 -eq n2 true if the integers n1 and n2 are algebraically equal.

OPERATORS
The above primaries may be combined with the following operators:
!
    unary negation operator.
-a     binary and operator.
-o     binary or operator (-a has higher precedence than -o).
(expr)  parentheses for grouping. Parentheses are meaningful
to the shell and, therefore, must be escaped.

SEE ALSO
find(1), sh(1).

WARNING
In the second form of the command (i.e., the one that uses [ ],
rather than the word test), the square brackets must be delimited
by blanks.
NAME
time - time a command

SYNOPSIS
time command

DESCRIPTION
Time executes command, then prints the time elapsed during command, the time spent in the system, and the time spent in execution of command.

Times are reported in seconds and are printed on standard error.
Time prints the times on standard error.

SEE ALSO
timex(1), times(2).
[This page left blank.]
NAME
timex - time a command; report process data and system activity

SYNOPSIS
timex [options] command

DESCRIPTION
Timex executes the given command, then reports in seconds elapsed time, user time and system time spent in execution. Optionally, timex processes accounting data for the command and lists or summarizes all its children, and reports total system activity during the execution interval.

The output of timex is written on standard error.

OPTIONS
-p List process accounting records for command and all its children. Suboptions f, h, k, m, r, and t modify the data items reported, as defined in acctcom(1). The number of blocks read or written and the number of characters transferred are always reported. Note: System accounting must be active for this option.

-o Report the total number of blocks read or written and total characters transferred by command and all its children. Note: System accounting must be active for this option.

-s Report total system activity (not just that due to command) that occurred during the execution interval of command. All the data items listed in sar(1) are reported.

SEE ALSO
acctcom(1), sar(1).

WARNING
Process records associated with command are selected from the accounting file /usr/adm/pacct by inference, since process genealogy is not available. Timex includes background processes having the same user-id, terminal-id, and execution time window.

EXAMPLES
A simple example:
timex -ops sleep 60

A terminal session of arbitrary complexity can be measured by timing a sub-shell:
timex -opskmt sh
  session commands
  EOT
NAME
toc, dtoc, ttoc, vtoc - graphical table of contents routines

SYNOPSIS
dtoc [directory]
ttoc mm-file
vtoc [-c d h n i m s v n] [TTOC file]

DESCRIPTION
Not on 7000 Series Systems.
All of the commands listed below reside in /usr/bin/graf (see
graphics(1G)).
dtoc
Dtoc makes a textual table of contents, TTOC, of all subdirectories
beginning at directory (directory defaults to .) with one entry
per directory. The entry fields from left to right are level
number, directory name, and the number of ordinary readable files
in the directory. Dtoc is useful in making a visual display of all or
parts of a file system.
The following makes a visual display of all the readable directories
under /:

dtoc / | vtoc | td
ttoc
Ttoc translates the table of contents generated by the .TC macro of
mm(1) to TTOC format. Ttoc assumes that mm file uses the .H fam­
ily of macros for section headers. If no mm-file is given, the stan­
dard input is assumed.

vtoc
Vtoc produces a GPS describing a hierarchy chart from a TTOC.
The output drawing consists of boxes containing text connected in
a tree structure. If no file is given, the standard input is
assumed. Each TTOC entry describes one box and has the form:

id [line-weight,line-style] "text " [mark]

Id is an alternating sequence of numbers and dots. The id speci­
fies the position of the entry in the hierarchy. The id 0. is the
root of the tree.

Line-weight is one of the following:

n, normal-weight; or
m, medium-weight; or
b, bold-weight.

Line-style is one of the following:

so, solid-line;
do, dotted-line;
dd, dot-dash line;
da, dashed-line; or
ld, long-dashed

*Text* is a character string surrounded by quotes. The characters between the quotes become the contents of the box. To include a quote within a box escape it with a backslash.

*Mark* is a character string (surrounded by quotes if it contains spaces). Any included quotes or dots must be escaped. *Vtoc* puts the string above the top right corner of the box.

Entry example: 1.1 b, da "ABC" DEF

Entries may span more than one line by escaping the new-line (*\new-line*).

Comments are surrounded by the */*, */ pair. They may appear anywhere in a TTOC.

Options:

- **c** Use text as entered, (default is all upper case).
- **d** Connect the boxes with diagonal lines.
- **hn** Set horizontal interbox space to \( n \% \) of box width.
- **i** Suppress the box *id*.
- **m** Suppress the box *mark*.
- **s** Do not compact boxes horizontally.
- **vn** Set vertical interbox space to \( n \% \) of box height.

**SEE ALSO**

- graphics(1G), gps(4).
NAME

touch - update access and modification times of a file

SYNOPSIS

touch [ -ame ] [ mmddhhmm[yy] ] files

DESCRIPTION

Touch updates the access and modification times of each file. If no
time is specified (see date(1)), the current time is used. If a file
does not exist, touch creates the file.

OPTIONS

The default options are -am.
- a Update only the access time.
- m Update only the modification time.
- c Do not create the file if it does not exist.

EXIT STATUS

The number of files for which the times could not be successfully
modified (including files that did not exist and were not created).

SEE ALSO

date(1), utime(2).

TOUCH(1)
NAME
tpcvt - filter for old streaming tape format

SYNOPSIS
tpcvt [ -VB filename ]

DESCRIPTION

Tpcvt filters the data from a streaming tape and makes sure the
data was not written to the tape by a previous streaming tape
driver. Tpcvt reads from standard input and writes to standard
output. The VB option causes tpcvt to read from the specified file.

The input data may be data from an old or new tape; tpcvt deter­
mines the source of the data and produces the correct output.

Tpcvt should be used via a pipe for receiving data from a tape and
sending data to a destination program such as cpio(1). The VB
option should be used when input consists of multi-volume tapes
which were created with the T option of cpio. When VB is speci­
fied, tpcvt prompts for new volumes. After all volumes are pro­
cessed, respond with an end-of-file (control-d) to the prompt for
the next volume.

SEE ALSO

tp(4), cpio(1).
TPCVT(1)

[This page left blank.]
NAME
tplot - graphics filters

SYNOPSIS
tplot [ -T terminal [ -e raster ] ]

DESCRIPTION
Not on 7000/40.

Tplot reads plotting instructions (see plot(4)) from the standard input and in general produces, on the standard output, plotting instructions suitable for a particular terminal. If no terminal is specified, the environment parameter TERM (see environ(5)) is used. Known terminals are:

300   DASI 300.
300S  DASI 300s.
450   DASI 450.
4014  TEKTRONIX 4014.
ver   Versatec D1200A. This version of tplot places a scan-converted image in /usr/tmp/raster$$ and sends the result directly to the plotter device, rather than to the standard output. The -e option causes a previously scan-converted file raster to be sent to the plotter.

FILES
/usr/lib/t300
/usr/lib/t300s
/usr/lib/t450
/usr/lib/t4014
/usr/lib/vplot
/usr/tmp/raster$$

SEE ALSO
    plot(3X), plot(4), term(5).
TPLOT(1G)

[This page left blank.]
NAME
tps - show processes use of GPTF (General Purpose Transaction Facility, see Administration Guide for further information)

SYNOPSIS
tps [-T] [-ef] [-n (namelist)]

DESCRIPTION
This command applies to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.

The General Purpose Transaction Facility is an optional subsystem of the operating system. It provides facilities to support real-time applications. When this subsystem is available, processes may be given special treatment and have access to a special set of kernel calls. Tps shows the use processes make of the GPTF facilities by displaying the GPTF flag together with other process-related information.

Tps works like ps and has all the facilities and recognizes all the options as does ps. Only the T option distinguishes tps from ps.

OPTIONS
-T The T option results in a display of the following columns per process:

F - S - GPTF - UID - PID - PPID - C -
UPRI - KPRI - ADDR - SZ - TTY - TIME - COMD

where GPTF is the GPTF transaction flag (octal and additive) associated with each process:

00 - uses no GPTF facility
01 - the process is resident
02 - the process has transaction capability
04 - the process has real-time priority
10 - the process has fixed priority
20 - not used
40 - the process may use GPTF kernel functions and bypass parameter checking

UPRI
is the user priority of the process. The user priority is the priority maintained by the system for a standard process (possibly influenced by a setting of the nice value) or it is the priority set by the user to either a fixed priority (60..127) or a real-time priority (40..59).

KPRI
is the kernel priority of the process (the kernel priority is a priority assigned temporarily to a process when it is sleeping, its value depends upon the reason for why it is sleeping).
The other columns have the same meaning as for a ps listing.

-e Print information about all processes.
-f Generate a full listing.
-n (namelist)
    The argument is taken as the name of an alternate system file in
    place of /unix.

EXAMPLES
    Show the status of all processes on a system called /unixgptf:

        tps -T -e -n/unixgptf

SEE ALSO
    ps(1), tpset(1M).
NAME
tput - query terminfo database

SYNOPSIS
tput [-T type ] capname

DESCRIPTION
Tput uses the terminfo(4) database to make terminal-dependent capabilities and information available to the shell. Tput outputs a string if the attribute (capability name) is of type string, or an integer if the attribute is of type integer. If the attribute is of type boolean, tput simply sets the exit code (0 for TRUE, 1 for FALSE), and does no output.

-Ttype Indicates the type of terminal. Normally this option is unnecessary, as the default is taken from the environment variable $TERM.
capname Indicates the attribute from the terminfo database. See terminfo(4).

EXAMPLES
tput clear Echo clear-screen sequence for the current terminal.
tput cols Print the number of columns for the current terminal.
tput -T450 cols Print the number of columns for the 450 terminal.
bold='tput smso'
Set shell variable bold to stand-out mode sequence for current terminal. This might be followed by a prompt:
echo "$\{bold\}Please type in your name: \c"
tput hc Set exit code to indicate if current terminal is a hardcopy terminal.

FILES
/etc/term/*/ Terminal descriptor files
/usr/include/term.h Definition files
/usr/include/curses.h

DIAGNOSTICS
Tput prints error messages and returns the following error codes on error:
-1 Usage error.
-2 Bad terminal type.
-3 Bad capname.
In addition, if a capname is requested for a terminal that has no value for that capname (e.g., tput -T450 lines), -1 is printed.

SEE ALSO
stty(1), terminfo(4).
[This page left blank.]
NAME

tr - translate characters

SYNOPSIS

tr [ -cdfs ] [ string1 [ string2 ] ]

DESCRIPTION

Tr copies the standard input to the standard output with substitution or deletion of selected characters. Tr maps input characters found in string1 into the corresponding characters of string2.

The following abbreviation conventions may be used to introduce ranges of characters or repeated characters into the strings:

[a-z] Stands for the string of characters whose ASCII codes run from character a to character z, inclusive.

[a*n ] Stands for n repetitions of a. If the first digit of n is 0, n is considered octal; otherwise, n is taken to be decimal. A zero or missing n is assumed to be a large number; this facility is useful for padding string2.

The escape character \ may be used as in the shell to remove special meaning from any character in a string. In addition, \ followed by 1, 2, or 3 octal digits stands for the character whose ASCII code is given by those digits.

OPTIONS

Any combination of the options -cdfs may be used:

-c Complements the set of characters in string1 with respect to the universe of characters whose ASCII codes are 001 through 377 octal.

-d Deletes all input characters in string1.

-s Squeezes all strings of repeated output characters that are in string2 to single characters.

EXAMPLE

The following example creates a list of all the words in file1 one per line in file2, where a word is taken to be a maximal string of alphabetics. The strings are quoted to protect the special characters from interpretation by the shell; 012 is the ASCII code for newline.

tr -cs "[A-Z][a-z]" "[\012*]" <file1 >file2

SEE ALSO

ed(1), sh(1), ascii(5).

RESTRICTIONS

ASCII NUL may not be used in string1 or string2; tr always deletes NUL from input.
[This page left blank.]
NAME
true, false - provide truth values

SYNOPSIS
true
false

DESCRIPTION
True does nothing, successfully. False does nothing, unsuccessfully. They are typically used in input to sh(1) such as:

    while true
    do
        command
    done

SEE ALSO
sh(1).

DIAGNOSTICS
True has exit status zero, false nonzero.
[This page left blank.]
NAME
ts - topological sort

SYNOPSIS
tsort [ file ]

DESCRIPTION
Tsort produces on the standard output a totally ordered list of
items consistent with a partial ordering of items mentioned in the
input file. If no file is specified, the standard input is assumed.

The input consists of pairs of items (nonempty strings) separated
by blanks. Pairs of different items indicate ordering. Pairs of
identical items indicate presence, but not ordering.

SEE ALSO
lorder(1).

DIAGNOSTICS
Odd data
    there is an odd number of fields in the input file.

RESTRICTIONS
Uses a quadratic algorithm for the typical use of ordering a library
archive file.
[This page left blank.]
NAME
  tty - get the name of the terminal

SYNOPSIS
  tty [-1] [-s]

DESCRIPTION
  Tty prints the path name of the terminal.

OPTIONS
  -1  Print the synchronous line number to which the terminal is con­
      nected if it is on an active synchronous line.

  -s  Suppress printing of the path name; generate the exit code
      only.

EXIT STATUS
  2  Invalid options were specified.
  0  Standard input is a terminal.
  1  Otherwise.

DIAGNOSTICS
  not on an active synchronous line
    The standard input is not a synchronous terminal and -1 is
    specified.

  not a tty
    The standard input is not a terminal and -s is not specified.
NAME
  ul - underline output for a terminal

SYNOPSIS
  ul [ -i ] [ -t terminal ] [ file ... ]

DESCRIPTION
  5000/20, 5000/30, 5000/40, and 5000/50 Systems only.
  Ul reads the named files (or standard input if none are given) and
  translates occurrences of underscores to the sequence which indi­
  cates underlining for the terminal in use, as specified by the
  environment variable TERM.
  Ul reads the file /etc/termcap to determine the appropriate
  sequences for underlining. If the terminal is incapable of underlin­
  ing, but is capable of a standout mode then that is used instead. If
  the terminal can overstrike, or handles underlining automatically,
  ul degenerates to cat(1). If the terminal cannot underline, under­
  lining is ignored.

OPTIONS
  -i  Indicate underlining by a separate line containing appropriate
       dashes; this is useful when you want to look at the underlining
       which is present in an nroff(1) output stream on a CRT­
       terminal.
  -t  Use the terminal kind terminal instead of the kind specified in
       the environment.

SEE ALSO
  man(1), nroff(1).

RESTRICTION
  Nroff usually outputs a series of backspaces and underlines inter­
  mixed with the text to indicate underlining. No attempt is made to
  optimize the backward motion.
[This page left blank.]
NAME
ulim - increase maximum file size limit

SYNOPSIS
/local/bin/ulim

DESCRIPTION
(5000/60/80/90 only)

Ulim is a login shell for changing a user's default maximum file size limit (ulimit). It must be invoked as a login shell from /etc/passwd to actually increase the ulimit. Otherwise, a default shell will be exec'd. To give a user an increased ulimit, two files must be changed. First, the user's password entry in /etc/passwd must be changed so the login shell is /local/bin/ulim. Second, an entry must be created for the user in the file /etc/ulimrc. The format for lines in this file is as follows:

```
login_name<tab>login_shell<tab>ulimit
```

where login_name is the user's login name (from /etc/passwd), login_shell is the shell the user previously used, and ulimit is the new maximum file size limit in 1024 byte blocks. Each field must be seperated by a tab. If a user's login shell is /local/bin/ulim, but that user is not listed in /etc/ulimrc, or if /local/bin/ulim is executed as a user process, ulim will exec /bin/sh (unless the user executing ulim is listed in /etc/ulimrc, in which case the shell listed in /etc/ulimrc will be forked) with the default ulimit.

EXAMPLE
The original /etc/passwd entry would look like this:

```
abc:OHqhAw60brU:97:7:User Name:/usr/abc:/bin/sh
```

The new entry should look like this:

```
abc:OHqhAw60brU:97:7:User Name:/usr/abc:/local/bin/ulim
```

The new entry in /etc/ulimrc should look like this:

```
abc/bin/sh 4096
```

This will give user 'abc' a ulimit of 4096 (four megabytes).

FILES
/etc/ulimrc add entry for user
/etc/passwd change user's login shell to /local/bin/ulim
NAME
umask - set file-creation mode mask

SYNOPSIS
umask [ 000 ]

DESCRIPTION
Umask sets the user file-creation mode mask to 000. The three octal digits refer to read/write/execute permissions for owner, group, and others, respectively (see chmod(2) and umask(2)). The value of each specified digit is subtracted from the corresponding digit specified by the system for the creation of a file (see creat(2)). For example, umask 022 removes group and others write permission (directories typically created with mode 777 become mode 755; files created with mode 666 become mode 644).

If 000 is omitted, the current value of the mask is printed.

The shell recognizes and executes umask.

SEE ALSO
chmod(1), sh(1), chmod(2), creat(2), umask(2).
UMASK(1)

[This page left blank.]
NAME
uname - print name of current UNIX system

SYNOPSIS
uname [ -amnrsv ]

DESCRIPTION
Uname prints the current system name of the UNIX system on the standard output file. Uname is mainly useful to determine what system one is using.

OPTIONS
-a Print all information. This is the same as entering all options.
-m Print the machine hardware name.
-n Print the nodename (the nodename may be a name that the system is known by to a communication network). Note: On the 5000/20/40/50, the node_name used by uucp(1) resides in /usr/lib/uucp/SYSTEMNAME.
-r Print the operating system release.
-s Print the system name (default).
-v Print the operating system version.

SEE ALSO
uname(2), setuname(1m), uucp(1c)
[This page left blank.]
NAME
unget - undo a previous get of an SCCS file

SYNOPSIS
unget [-rSID] [-s] [-n] files

DESCRIPTION
Unget undoes the effect of a get -e made prior to creating the intended new delta. If a directory is named, unget behaves as though each file in the directory were specified as a named file, except that non-SCCS files and unreadable files are silently ignored. If a name of - is given, the standard input is read with each line being taken as the name of an SCCS file to be processed.

OPTIONS
-rSID
Uniquely identify which delta is no longer intended. (This would have been specified by get as the new delta). The use of this option is necessary only if two or more outstanding gets for editing on the same SCCS file were done by the same person (login name). A diagnostic results if the specified SID is ambiguous, or if it is necessary and was omitted on the command line.

-s Suppress the printout, on the standard output, of the SID of the intended delta.

-n Retain the get file which is normally removed from the current directory.

SEE ALSO
delta(1), get(1), help(1), sact(1).

DIAGNOSTICS
Use help(1) for explanations.
[This page left blank.]
NAME
uniq - report repeated lines in a file

SYNOPSIS
uniq [ -udc [ +n ] [ -n ] ] [ input [ output ] ]

DESCRIPTION
Uniq reads the input file comparing adjacent lines, and removes the second and succeeding copies of repeated lines. Uniq writes the remaining lines on the output file. Input and output should always be different. Note that repeated lines must be adjacent in order to be found; see sort(1).

OPTIONS
-u Output only lines not repeated in the original file.
-d Output one copy of every repeated line (no other lines).
-c Generate normal output, preceding each line with the number of times it occurred (supersedes -u and -d).
-n Ignore the first n fields together with any blanks before each for the comparison. A field is defined as a string of non-space, non-tab characters separated by tabs and spaces from its adjacent fields.
+n Ignore the first n characters for the comparison. Uniq skips fields before characters.

SEE ALSO
comm(1), sort(1).
[This page left blank.]
NAME
   units - interactive conversion program

SYNOPSIS
   units

DESCRIPTION
   Units converts quantities expressed in various standard scales to their equivalents in other scales. It works interactively in this fashion:
   
   You have: inch
   You want: cm
   * 2.540000e+00
   3.937008e-01

   Units specifies a quantity as a multiplicative combination of units optionally preceded by a numeric multiplier. Powers are indicated by suffixed positive integers, division by the usual sign:
   
   You have: 15 lbs force/in2
   You want: atm
   * 1.020689e+00
   9.797299e-01

   Units only does multiplicative scale changes; thus it can convert Kelvin to Rankine, but not Celsius to Fahrenheit. Most familiar units, abbreviations, and metric prefixes are recognized, along with a few constants of nature including:

   pi    ratio of circumference to diameter,
   c     speed of light,
   e     charge on an electron,
   g     acceleration of gravity,
   force same as g,
   mole  Avogadro's number,
   water pressure head per unit height of water,
   au    astronomical unit.

   Units recognizes lb, rather than pound as a unit of mass.

   Compound names are run together, (e.g. lightyear).

   British units that differ from their U.S. counterparts should be prefixed thus: brgallon.

   For a complete list of units, type:
       cat /usr/lib/unittab

FILES
   /usr/lib/unittab
[This page left blank.]
NAME
uptime - show how long system has been up

SYNOPSIS
uptime

DESCRIPTION
7000 Series Systems only.

Uptime prints the current time, the length of time the system has been up, and the average number of jobs in the run queue over the last 1, 5 and 15 minutes. It is, essentially, the first line of a w(1) command.

FILES
/unix system name list

SEE ALSO
w(1)
UPTIME(1)

[This page left blank.]
NAME
   usage - retrieve a command description and usage examples

SYNOPSIS
   [ help ] usage [ -d ] [ -e ] [ -o ] [ command_name ]

DESCRIPTION
   5000/60, 5000/80, and 5000/90 only.

   The UNIX system Help Facility command usage retrieves information about UNIX system commands. With no argument, usage displays a menu screen prompting the user for the name of a command, or allows the user to retrieve a list of commands supported by usage. The user may also exit to the shell by typing q (for "quit").

   After a command is selected, the user is asked to choose among a description of the command, examples of typical usage of the command, or descriptions of the command's options. Then, based on the user's request, the appropriate information will be printed.

   A command name may also be entered at shell level as an argument to usage. To receive information on the command's description, examples, or options, the user may use the -d, -e, or -o options respectively. (The default option is -d.)

   From any screen in the Help Facility, a user may execute a command via the shell (sh(1)) by typing a ! and the command to be executed. The screen will be redrawn if the command that was executed was entered at a first level prompt. If entered at any other prompt level, only the prompt will be redrawn.

   By default, the Help Facility scrolls the data that is presented to the user. If you prefer to have the screen clear before printing the data (non-scrolling), the shell variable SCROLL must be set to no and exported so it will become part of your environment. This is done by adding the following line to your .profile file (see profile(4)): "export SCROLL ; SCROLL=no". If you later decide that scrolling is desired, SCROLL must be set to yes.

   Information on each of the Help Facility commands (starter, locate, usage, glossary, and help) is located on their respective manual pages.

SEE ALSO
   glossary(1), help(1), locate(1), sh(1), starter(1).

WARNINGS
   If the shell variable TERM (see sh(1)) is not set in the user's .profile file, then TERM will default to the terminal value type 450 (a hard-copy terminal). For a list of valid terminal types, refer to term(5).
NAME
uucp, uulog, uuname - UNIX system to UNIX system copy

SYNOPSIS
uucp [ options ] source-files destination-file
uulog [ options ]
uuname [ options ]
The following items apply to the, 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00 only.
uulog [ options ] -s system
uulog [ options ] system
uulog [ options ] -f system
uuname [ -c ] [ -l ]

DESCRIPTION
Uucp
Uucp copies files named by the source-file arguments to the file named by the destination-file argument. A file name may be a path name on your machine, or may have the form:

    system-name!path-name

where system-name is taken from a list of system names which uucp knows about. The system-name may also be a list of names such as

    system-name!system-name!...!system-name!path-name

in which case an attempt is made to send the file via the specified route, and only to a destination in PUBDIR (see below). Care should be taken to insure that intermediate nodes in the route are set up to forward information.

The shell metacharacters ? , * , and [ ... ] appearing in path-name are expanded on the appropriate system. In order to send files that begin with a dot (e.g., the files must be qualified with a dot. For example: and are correct; whereas *prof* and ?prof* are not correct.

Path names may be one of:

(1) a full path name;
(2) a path name preceded by ~ user where user is a login name on the specified system and is replaced by that user's login directory;
(3) a path name preceded by ~/destination where destination is appended to /usr/spool/uuuppublic. Note: This destination is treated as a file name unless more than one file is being transferred by this request or the destination is already a directory. To ensure that it is a directory, follow the destination with a /. For example, ~/dan/ as the destination makes the directory /usr/spool/uuucppublic/dan if it does not exist and puts the requested file(s) in that directory.
(4) a path name preceded by `/user where user is a login name on
the specified system and is replaced by that user's directory
under PUBDIR; or
(5) a file name or path name; uucp prefixes either with the current
directory.

If the result is an erroneous path name for the remote system the
copy fails. If the destination-file is a directory, the last part of the
source-file name is used.

Uucp preserves execute permissions across the transmission and
gives 0666 read and write permissions (see chmod (2)).

Uucp associates a job number with each request. This job number
can be used by uustat (1C) to obtain status information or ter-
minate the job.

The environment variable JOBNO and the -j option of uucp are used
to control the listing of the uucp job number on standard output.
If the environment variable JOBNO is undefined or set to OFF, the
job number is not listed (default). If uucp is then invoked with the
-j option, the job number is listed. If the environment variable
JOBNO is set to ON and is exported, a job number is written to
standard output each time uucp is invoked. In this case, the -j
option supresses output of the job number. Uucp does not gen-
erate a job number for a strictly local transaction.

The following options are interpreted by uucp:

```
-c  uses the source file when copying out rather than
copying the file to the spool directory (default).
-C  copies the source file to the spool directory.
-l  forces the link of local files to the spool directory
    for transfer. If the link is not possible, a copy is
    performed.
-d  makes all necessary directories for the file copy
    (default).
-esys sends the uucp command to system sys to be exe-
cuted there. This is successful only if the remote
machine allows the uucp command to be executed
by /usr/lib/uucp/uuxqt. (This option is avail-
able on the 5000/35 or the 5000/55.)
-f  does not make intermediate directories for the file
copy.
-ggrade The grade is a single letter/number, lower ASCII
    sequence characters cause the job to be transmitt-
ed earlier during a particular conversation.
-j  controls writing of the uucp job number to stan-
dard output by changing the value of the
    environment variable JOBNO.
```
UUCP(1C)

-mf file reports the status of the transfer in file. If file is omitted, send mail to the requester when the copy is completed.

-s file reports the status of the transfer to file. Note that the file must be a full path name.

-n user notifies user on the remote system that a file was sent.

-r queues job, but does not start the file transfer process. By default, a file transfer process is started each time uucp is evoked.

-xdebug_level produces debugging output on standard output. The debug_level is a number between 0 and 9 with the higher numbers giving more detailed information.

Uulog
Uulog queries a summary log of uucp and uux (1C) (appears as uuxqt on the 5000/30, 5000/35, 5000/50, and the 5000/55 Release 2.00) transactions in the file /usr/spool/uucp/LOGFILE.

On the 5000/35 and the 5000/55 systems, uulog queries a log file of uucp or uuxqt transactions in a file called /usr/spool/uucp/.LOG/uucico/system or /usr/spool/uucp/.LOG/uuxqt/system.

OPTIONS
The options cause uulog to print logging information:

-ssys prints information about work involving system sys. If sys is not specified, then logging information for all systems is not printed.

-f system does a tail of -f of the file transfer log for system. Other options are used in conjunction with the previous information.

-x looks in the uuxqt log file for the given system.

-number indicates that a tail command of number lines should be executed.

-uuser prints information about work done for the specified user. If the user is not specified, then logging information for all users is printed.

Uuname
Uuname lists the uucp names of known systems. A description is printed for each system that has a line of information in /usr/lib/uucp/admin. The format of admin is:
sysname tab description tab.

OPTIONS
The following options are interpreted by uuname.

-c lists the names of systems known to cu.
(The two lists are the same, unless the machine is using different Systems files for cu)
UUCP(1C)

and uucp. See the Sysfiles files.

- I
  returns the local system name.
- V
  prints additional information about each system
not available on the 5000/30, 5000/35, 5000/50, and the 500

FILES
    /usr/spool/uucp         spool directory
    /usr/spool/uucppublic   public directory for receiving and sending
                           (PUBDIR)
    /usr/lib/uucp/*         other data and program files

SEE ALSO
    mail(1), uustat(1), uux(1C), chmod(2), uuxqt(1M).
    "UUCP Administration" in the Administrator Guide.

WARNING
    The domain of remotely accessible files can (and for obvious secu-
    rity reasons, usually should) be severely restricted. You will
    very likely not be able to fetch files by path name; ask a responsi-
    ble person on the remote system to send them to you. For the same
    reasons, you will probably not be able to send files to arbitrary
    path names. As distributed, the remotely accessible files are those
    whose names begin /usr/spool/uucppublic (equivalent to "nuucp
    or just ").

    (5000/30, 5000/35, 5000/50, and 5000/55 release 2.00 only.)
Retrieving multiple files specified by special shell characters ?, *,
and [...] activates the -m option. The -m option is ignored if the -s
option is specified.

The forwarding of files through other systems may not be compati-
ble with the previous version of uucp. If forwarding is used, all
systems in the route must have the same version of uucp.

For mail to be used correctly, uname -n must be the same as
uuname-1. The setuname(1) command permits you to change the
node name in the kernel (memory) and on disk.

RESTRICTIONS
    All files received by uucp are owned by uucp .
The -m option works only sending files or receiving a single file.
Receiving multiple files specified by special shell characters ?, *,
and [...] does not activate the -m option.

For the 5000/35 and 5000/55 Release 2.0 only, retrieving multiple
files specified by special shell characters ?, *, and [...] activates
the -m option. The -m option is ignored if the -s option is specified.
The -m option does not work if all transactions are local or if uucp is
executed remotely using the -e option.
The -n option functions only when the source and destination are
not on the same machine.

Only the first six characters of a system-name are significant. Any
excess characters are ignored.
Protected files that are in protected directories that are owned by the requester can be sent by uucp. However, if the requester is root, and the directory is not searchable by "other" or the file is not readable by "other", the request fails.

A source file can only be retrieved if there is no more than an ! in the path. That is, a user on system_A wishes to retrieve a file from system_B, the command:

```
uucp system_B!/filename!/filename
```

retrieves the file filename from the remote system_B and places it in the user's local PUBDIR (/usr/spool/uucppublic).

If the user is connected to system_B, through an intermediate remote system_C, however, the command:

```
uucp system_C!/system_B!/filename!/filename
```

does not work.
UUCP(1C)

[This page left blank.]
NAME
   uustat - uucp status inquiry and job control

SYNOPSIS
   uustat [ options ]

DESCRIPTION
   Uustat displays the status of, or cancels, previously specified
   uucp (1) commands, or provides general status on uucp connec-
   tions to other systems.

OPTIONS
   When no options are given, uustat outputs the status of all uucp
   requests issued by the current user.

   The following options are mutually exclusive; that is, only one of
   the following may be specified on the command line:
     -a     Output all jobs in queue.
     -chour Remove the status entries which are older than hour hours.
     This administrative option can only be initiated by the user
     uucp or the superuser.
     -jjobn Requests the status of the uucp request jobn (jobnumber).
     If all is used for jobn, uustat reports the status of all uucp
     requests. An argument must be supplied. If jobn is omit-
     ted, uustat prints the usage message and fails. (This
     option does not apply to the 5000/30, 5000/35, 5000/50, and
     5000/55 Release 2.00.00.)
     -kjobn Kill the uucp request whose job number is jobn. The speci-
     fied uucp request must belong to the user issuing the uustat
     command unless the user is the superuser.
     -mmch Report the status of accessibility of machine mch. If mch is
     specified as all, uustat provides the status of all machines
     known to the local uucp.
     -p     Execute a "ps -flp" for all the process-ids that are in the
     lock files.
     -Mmch Same as the -m option except that the last status was
     obtained and the time that the last successful transfer to
     that system occurred. (This option is not available on the
     5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00.)
     -rjobn Rejuvenate jobn. Set the modification time of jobn to the
     current time. This prevents uuclean(1M) from deleting the
     job until the modification time of the job reaches the limit
     imposed by uuclean. (For the 5000/30, 5000/35, 5000/50,
     5000/55 Release 2.00, uucleanup replaces uuclean.)

   The following options are not mutually exclusive:
     -ohour reports the status of all uucp requests wich are older than
     hour hours. (This option is not available on the 5000/30,
     5000/35, 5000/50, or the 5000/55 Release 2.00.00.)
The command:

```
uustat -ucarter -stower -y72
```

prints the status of all uucp requests that were issued by user carter to communicate with system tower within the last 72 hours.

**STATUS CODES**

The meanings of the job request status are:

- **job-number** user remote-system command-time status-time status
where the status may be either an octal number or a verbose description. The octal code corresponds to the following description:

<table>
<thead>
<tr>
<th>Octal</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>000001</td>
<td>the copy failed, but the reason cannot be determined</td>
</tr>
<tr>
<td>000002</td>
<td>permission to access local file is denied</td>
</tr>
<tr>
<td>000004</td>
<td>permission to access remote file is denied</td>
</tr>
<tr>
<td>000010</td>
<td>bad uucp command is generated</td>
</tr>
<tr>
<td>000020</td>
<td>remote system cannot create temporary file</td>
</tr>
<tr>
<td>000040</td>
<td>cannot copy to remote directory</td>
</tr>
<tr>
<td>000100</td>
<td>cannot copy to local directory</td>
</tr>
<tr>
<td>000200</td>
<td>local system cannot create temporary file</td>
</tr>
<tr>
<td>000400</td>
<td>cannot execute uucp</td>
</tr>
<tr>
<td>001000</td>
<td>copy (partially) succeeded</td>
</tr>
<tr>
<td>002000</td>
<td>copy finished, job deleted</td>
</tr>
<tr>
<td>004000</td>
<td>job is queued</td>
</tr>
<tr>
<td>010000</td>
<td>job killed (complete)</td>
</tr>
<tr>
<td>020000</td>
<td>job killed (incomplete)</td>
</tr>
</tbody>
</table>

The meanings of the machine accessibility status are:

```
system-name time status
```

where time is the latest status time and status is a self-explanatory description of the machine status.

FILES

```
/usr/spool/uucp spool directory
/usr/lib/uucp/L_stat
```

system status file

```
/usr/lib/uucp/R_stat
```

request status file

SEE ALSO

uucp(1C), uuclean(1M).
[This page left blank.]
NAME
uuto, uupick - public UNIX-to-UNIX system file copy

SYNOPSIS
uuto [ options ] source-files destination
uupick [ -s system ]

DESCRIPTION
Uuto
Uuto sends source-files to destination. Uuto uses the uucp(1C) facility to send files, while it allows the local system to control the file access.

A source-file name is a path name on your machine. In order to send files that begin with a dot (e.g., .profile) the files must be qualified with a dot. For example: .profile, .prof*, and .profil? are correct; whereas *prof* and ?profile are not correct.

Destination has the form:

    system!user

where system is taken from a list of system names that uucp knows about (see uuname(1)). User is the login name of someone on the specified system.

The following options are available:

-p copies the source file into the spool directory before transmission.

-m sends mail to the sender when the copy is complete. This option is applicable when there is only one remote system is involved.

-l links the source file into the spool directory /usr/spool/uucp/system before transmission. If the link is not possible, a copy is performed.

Uuto sends the files (or sub-trees if directories are specified) to PUBDIR on system, where PUBDIR is a public directory defined to uucp. Specifically uuto sends the files to:

    PUBDIR/receive/user/mysystem/files

Uuto notifies the destined recipient by mail(1) of the arrival of files.

Uupick
Uupick accepts or rejects the files transmitted to the user. Specifically, uupick searches PUBDIR for files destined for the user. For each entry (file or directory) found, uupick prints the following message on the standard output:

    from system: [file file-name] [dir dirname] ?

Uupick then reads a line from the standard input to determine the disposition of the file:
Go on to next entry.

Delete the entry.

Move the entry to named directory \texttt{dir} (current directory is default).

Same as \texttt{m} except move \textit{all} the files sent from \texttt{system}.

Print the content of the file.

Stop.

Same as \texttt{q}.

Escape to the shell to do \texttt{command}.

Print a command summary.

\texttt{Uupick} invoked with the \texttt{-s system} option only searches the \texttt{PUBDIR} for files sent from \texttt{system}.

\begin{verbatim}
/usr/spool/uucppublic public directory (PUBDIR)
\end{verbatim}

SEE ALSO

\texttt{mail(1), uuclean(IM), uucp(1C), uustat(1C), uux(1C), uname(1), setuname(IM)}.

WARNING

This warning pertains to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.

For \texttt{mail} to be used correctly, \texttt{uname -n} must be the same as \texttt{uuname -l}. The \texttt{setuname(IM)} command permits you to change the node name in the kernel (memory) and on disk.

RESTRICTIONS

This restriction pertains to the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00.00 only.

\texttt{Uuto} does not send null directories.
NAME
uux - UNIX-to-UNIX system command execution

SYNOPSIS
uux [ options ] command-string

DESCRIPTION
Uux gathers zero or more files from various systems, executes a command on a specified system, and then sends standard output to a file on a specified system. Note that, for security reasons, many installations limit the list of commands executable on behalf of an incoming request from uux. Many sites permit little more than the receipt of mail (see mail(1)) via uux.

When uux gathers the files, the files are given permission 600 and are owned by uucp. The actual program running is nuucp so access permission to the files may be denied. For example, print(1) checks permissions and fails. Redirect standard input for uux to print the files.

The command-string is made up of one or more arguments that look like a shell command line, except that the command and file names may be prefixed by system-name!. A null system-name is interpreted as the local system.

File names may be
• a full path name;
• a path name preceded by ~xxx where xxx is a login name on the specified system and is replaced by that user login directory; or
• a file name or path name; uux prefixes either with the current directory.

Any special shell characters such as <>; should be quoted either by quoting the entire command-string or quoting the special characters as individual arguments.

Uux attempts to get all files to the execution system. For files which are output files, the file name must be escaped using parentheses.

Uux notifies you if the requested command on the remote system was disallowed. The response comes by remote mail from the remote machine. Executable commands are listed in /usr/lib/uucp/L.cmds on the remote system. The format of the L.cmds file is:

    cmd, machine1, machine2, ...

If no machines are specified, then any machine can execute cmd. If machines are specified, only the listed machines can execute cmd. If the desired command is not listed in L.sys, then no machine can execute that command.

Redirection of standard input and output is usually restricted to files in PUBDIR. Directories into which redirection is allowed must be specified in /usr/lib/uucp/USERFILE by the system administrator. The directory must have 777 permissions.
UUX associates a job number with each request. This job number can be used by uustat(1) to obtain status information or terminate the job. The environment variable JOBNO and the -j option are used to control the listing of the uux job number on standard output. If the environment variable JOBNO is undefined or set to OFF, the job number is not listed (default). If uux is then invoked with the -j option, the job number is listed. If the environment variable JOBNO is set to ON and is exported, a job number is written to standard output each time uux is invoked. In this case, the -j option suppresses output of the job number.

OPTIONS
The following options are interpreted by uux:

- is the standard input to uux is made the standard input to the command-string.

-aname uses name as the user identification replacing the initiator user ID. (Notification is returned to the user.)

-b returns whatever standard input was provided to the uux command if the exit status is non-zero.

-c does not copy local file to the spool directory for transfer to the remote machine (default).

-C forces the copy of local files to the spool directory for transfer.

-l forces the link of local files to the spool directory for transfer. If a link is not possible, a copy is done.

-ggrade grade is a single letter or number. Lower ASCII sequence characters cause the job to be transmitted earlier during a particular conversation.

-j controls writing of the uux job number to standard output by changing the value of the environment variable JOBNO.

-n sends no notification to user.

-mfile reports the status of the transfer in file. If file is omitted, send mail to the requester when the copy is completed. (This option is not available on the 5000/30, 5000/35, 5000/50, or the 5000/55 Release 2.00.)

-r queues the job, but does not start the file transfer process. By default, a file transfer process is started each time uux is evoked.

-p is the same as -:. The standard input to uux is made the standard input to the command-string.

-sfile reports the status of the transfer in file.
-xdebug_level produces debugging output on the standard output. The debug_level is a number between 0 and 9 with higher numbers giving more detailed information.

-z sends success notification to the user.

EXAMPLES
The command
   uux "!diff 5000-30/usr/carter/f1 5000-50/a4/carter/f1 > f1.diff"
gets the f1 files from the 5000-30 5000-50 machines, executes a diff(1) command and put the results in f1.diff in the local directory.

The command
   uux 5000-30/uucp 5000-50/usr/file \(5000-55/usr/file\)
 sends a uucp command to system 5000-30 get /usr/file from system 5000-50 and send it to system 5000-55.

FILES
/usr/spool/uucp spool directory
/usr/spool/uucppublic public directory (PUBDIR)
/usr/lib/uucp/* other data and programs
/usr/lib/uucp/Permissions remote execution permissions

SEE ALSO
mail(1), uuclean(1M), uucp(1C), uucleanup(1M).

RESTRICTIONS
Only the first command of a shell pipeline may have a system-name!. All other commands are executed on the system of the first command.

The use of the shell metacharacter * probably does not do what you want it to do. The shell tokens << and >> are not implemented.

Only the first six characters of the system-name are significant. Any excess characters are ignored.

To redirect output to a file, the name of the directory containing the file must be on the default system line in /usr/lib/uucp/USERFILE. This allows any system to redirect to this directory.

For the 5000/30, 5000/35, 5000/50, and 5000/55 Release 2.00 only, the execution of commands on remote systems takes place in an execution directory known to the uucp system. All files required for the execution are put into this directory unless they already reside on that machine. Therefore, the simple file name (without path or machine reference) must be unique within the uux request. The following command does NOT work:

   uux "a!diff b!/usr/dan/xyz c!/usr/dan/xyz> !xyz.diff"

but the command:
uux "a!diff a!/usr/dan/xyz c!/usr/dan/xyz > !xyz.diff"

works is diff(1) is a permitted command.

For uux to perform on the local system, there must be a MACHINE = "local system name" in the local system /usr/lib/uucp/Permissions file with the allowable commands defined.

For mail to be used correctly, uname -n must be the same as uuname-1. The setuname(1M) command permits you to change the node name in the kernel (memory) and on disk.

Protected files and files that are in portected directories that are owned by the requester can be sent in commands using uux. If the requester is root, however, and the directory is not searchable by "other", the request fails.

The system name may be up to eight characters and it must not con­tain a slash (/) character. For uux to operate properly in all cases, every system should have the system name returned from the uuname -l command the same as the uname -n command on their system.
NAME
val - validate SCCS file

SYNOPSIS
val -
    val [-s] [-rSID] [-mname] [-ytype] files

DESCRIPTION
Val determines if the specified file is an SCCS file with the characteristics specified by the optional argument list. Arguments to val may appear in any order. The arguments consist of options, which begin with a -, and named files.

When the file argument - is specified, val reads the standard input until it detects an end-of-file condition. Val processes each input line independently as if the input were a command line argument list. Val generates diagnostic messages on the standard output for each command line and file processed and also returns a single 8-bit code upon exit as described below.

OPTIONS
Each option affects each named file on the command line independently.

-s Silence the diagnostic message normally generated on the standard output for any error that val detects while processing each named file on a given command line.

-rSID Denote the argument value SID (SCCS IDentification String) as an SCCS delta number. Val first determines whether the SID is ambiguous (e.g., r1 is ambiguous because it physically does not exist but implies 1.1, 1.2, etc. which may exist) or invalid (e.g., r1.0 or r1.1.0 are invalid because neither case can exist as a valid delta number). If the SID is valid and not ambiguous, val determines whether the corresponding file actually exists.

-mname Compare the name with the SCCS %M% keyword in file.

-ytype Compare type with the SCCS %Y% keyword in file.

EXIT STATUS
The 8-bit code returned by val can be interpreted as a bit string where (moving from left to right) set bits are interpreted as follows:

bit 0 = missing file argument;
bit 1 = unknown or duplicate keyletter argument;
bit 2 = corrupted SCCS file;
bit 3 = cannot open file or file not SCCS;
bit 4 = invalid or ambiguous SID;
bit 5 = nonexistent SID;
bit 6 = %Y%, -y mismatch;
bit 7 = %M%, -m mismatch;
Note that val can process two or more files on a given command line and in turn can process multiple command lines (when reading the standard input). In these cases val returns an aggregate code: a logical OR of the codes generated for each command line and file processed.

SEE ALSO
admin(1), delta(1), get(1), prs(1).

DIAGNOSTICS
Use help(1) for explanations.

RESTRICTIONS
Val can process up to 50 files on a single command line. Any number above 50 produces a core dump.
NAME
val - validate SCCS file

SYNOPSIS
val
val [-s] [-rSID] [-mname] [-ytype] files

DESCRIPTION
Val determines if the specified file is an SCCS file with the characteristics specified by the optional argument list. Arguments to val may appear in any order. The arguments consist of options, which begin with a -, and named files.

When the file argument - is specified, val reads the standard input until it detects an end-of-file condition. Val processes each input line independently as if the input were a command line argument list.

Val generates diagnostic messages on the standard output for each command line and file processed and also returns a single 8-bit code upon exit as described below.

OPTIONS
Each option affects each named file on the command line independently.

-s Silence the diagnostic message normally generated on the standard output for any error that val detects while processing each named file on a given command line.

-rSID Denote the argument value SID (SCCS ID entification String) as an SCCS delta number. Val first determines whether the SID is ambiguous (e.g., r1 is ambiguous because it physically does not exist but implies 1.1, 1.2, etc. which may exist) or invalid (e.g., r1.0 or r1.1.0 are invalid because neither case can exist as a valid delta number). If the SID is valid and not ambiguous, val determines whether the corresponding file actually exists.

-mname Compare the name with the SCCS %M% keyword in file.

-ytype Compare type with the SCCS %Y% keyword in file.

EXIT STATUS
The 8-bit code returned by val can be interpreted as a bit string where (moving from left to right) set bits are interpreted as follows:

bit 0 = missing file argument;
bit 1 = unknown or duplicate keyletter argument;
bit 2 = corrupted SCCS file;
bit 3 = cannot open file or file not SCCS;
bit 4 = invalid or ambiguous SID;
bit 5 = non-existent SID;
bit 6 = %Y%, -y mismatch;
bit 7 = %M%, -m mismatch;
Note that `val` can process two or more files on a given command line and in turn can process multiple command lines (when reading the standard input). In these cases `val` returns an aggregate code: a logical OR of the codes generated for each command line and file processed.

**SEE ALSO**

`admin(1), delta(1), get(1), prs(1)`.

**DIAGNOSTICS**

Use `help(1)` for explanations.

**RESTRICTIONS**

`Val` can process up to 50 files on a single command line. Any number above 50 produces a core dump.
NAME
vc - version control

SYNOPSIS
vc [-a] [-t] [-cchar] [-s] [keyword=value ... keyword=value]

DESCRIPTION
The vc command copies lines from the standard input to the stan-
dard output under control of its arguments and any control state-
ments encountered in the standard input. In the process of per-
forming the copy operation, user declared keywords may be
replaced by their string value when they appear in plain text
and/or control statements.

The copying of lines from the standard input to the standard out-
put is conditional, based on tests (in control statements) of key-
word values specified in control statements or as vc command argu-
ments.

A control statement is a single line beginning with a control charac-
ter, except as modified by the -t option (see below). The default
control character is colon (:), except as modified by the -c option
(see below). Input lines beginning with a backslash followed by a
control character are not control lines and are copied to the stan-
dard output with the backslash removed. Lines beginning with a
backslash followed by a non-control character are copied in their
entirety.

A keyword is composed of up to 9 alphanumerics, the first of which
must be alphabetic. A value is any ASCII string that can be
created with ed(1); a numeric value is an unsigned string of
digits. Keyword values may not contain blanks or tabs.

Vc replaces keywords with values whenever a keyword surrounded
by control characters is encountered on a version control state-
ment. The -a option (see below) forces replacement of keywords in
all lines of text. An uninterpreted control character may be
included in a value by preceding it with a backslash. If a literal
backslash is desired, then it too must be preceded by backslash.

OPTIONS
-a Replace keywords surrounded by control characters
with their assigned value in all text lines and not just
in vc statements.

-t Ignore all characters from the beginning of a line up
to and including the first tab character for the pur-
pose of detecting a control statement. If a control
statement is found, discard all characters up to and
including the tab.

-cchar Use char as a control character in place of ::

-s Silence warning messages (not error messages) that
are normally printed on the diagnostic output.
VERSION CONTROL STATEMENTS
  :dcl keyword[, ..., keyword]
      Declares keywords. All keywords must be declared.
  :asg keyword=value
      Assigns values to keywords. An asg statement overrides the
      assignment for the corresponding keyword on the vc command
      line and all previous asg's for that keyword. Keywords
      declared, but not assigned values have null values.
  :if condition
     ...
     ...
  :end
      Skips lines of the standard input. If condition is true, vc
      copies all lines between the if statement and the matching end
      statement to the standard output. If condition is false, vc dis­
      cards all intervening lines, including control statements. Note
      that vc recognizes intervening if statements and matching end
      statements solely for the purpose of maintaining the proper
      if-end matching.

The syntax of a condition is:

<cond> ::= [ "not" ] <or>
<or> ::= <and> | <and> "\&" <or>
<and> ::= <exp> | <exp> "\&" <and>
<exp> ::= "(" <or> ")" | <value> <op> <value>
<op> ::= "=" : "!=" : "<" : ">"
[value] ::= <arbitrary ASCII string> | <numeric string>

The available operators and their meanings are:

  =    equal
  !=   not equal
  &    and
  I    or
  >    greater than
  <    less than
  ( )  used for logical groupings
  not  may only occur immediately after the if, and
       when present, inverts the value of the
       entire condition

The > and < operate only on unsigned integer values (for
example, 012 > 12 is false). All other operators take strings
as arguments (for example : 012 != 12 is true). The pre­
cedence of the operators (from highest to lowest) is:

  = != > <  equal precedence
  & I
Parentheses may be used to alter the order of precedence. Values must be separated from operators or parentheses by at least one blank or tab.

::text
Replaces keywords on lines that are copied to the standard output. Vc removes the two leading control characters, and replaces keywords surrounded by control characters in text by their value before copying the line to the output file. This action is independent of the -a option.

:on
:off
Turn on or off keyword replacement on all lines.

:ctl char
Change the control character to char.

:msg message
Prints the given message on the diagnostic output.

:err message
Prints the given message followed by:

ERROR: err statement on line ... (915)
on the diagnostic output. Vc halts execution, and returns an exit code of 1.

SEE ALSO ed(1), help(1).

DIAGNOSTICS
Use help(1) for explanations.

EXIT STATUS
0 - normal
1 - error
NAME
version - display release identifications of installed software

SYNOPSIS
version

DESCRIPTION
7000 Series only.

The `version` command displays the operating system's name, release identification, machine identification and patch number, along with the installed options' names and release identifications. The release identification has five fields describing, in order, the major and minor release identification, patch number, product identification, machine identification, and preliminary release identification. The fifth field is optional.

EXAMPLE
The release identification "1R2 2.11 5.R2 7000/40" describes release id 2.1b, patch number 01, product identification 5.R2 (System V, Release 2), and machine identification 7000 (7000/40). There is no preliminary release identification.

FILES
/install/versions/*vers

SEE ALSO
uname(1).
[This page left blank]
NAME

vi, view, vedit - screen-oriented (visual) display editor based on ex

SYNOPSIS

vi [-t tag] [-r file] [-l] [-wn] [-x] [-R] [+command] name ...
view [-t tag] [-r file] [-l] [-wn] [-x] [-R] [+command] name ...
vedit [-t tag] [-r file] [-l] [-wn] [-x] [-R] [+command] name ...

DESCRIPTION

Vi (visual) is a display-oriented text editor based on the underlying line editor ex(1). It is possible to use the command mode of ex from within vi and vice-versa. When using vi, changes you make to the file are reflected in what you see on your terminal screen. The position of the cursor on the screen indicates the position within the file.

The view invocation is the same as vi except that the readonly flag is set, preventing accidental overwriting of the file.

The vedit invocation is intended for beginners. The report flag is set to 1, and the showmode and novice flags are set. These defaults make it easier to get started learning the editor.

OPTIONS

The following invocation options are interpreted by vi:

-t tag
Edit the file containing the tag and position the editor at its definition.

-r file
Recover file after an editor or system crash. If file is not specified, a list of all saved files is printed.

-l Indent appropriately for lisp code, the ( ) {} [] and \ commands in vi and open are modified to have meaning for lisp.

-wn
Set the default window size to n. This is useful when using the editor over a slow speed line.

-x Set encryption mode; a key is prompted for allowing creation or editing of an encrypted file (see crypt(1)). This option can be used only with the domestic release of the operating system.

-R Set read only mode; the readonly flag is set preventing accidental overwriting of the file.

+command
The specified ex command is interpreted before editing begins.

name
Files to be edited.

VI MODES

Command
Normal and initial mode. Other modes return to command mode upon completion. ESC (escape) is used to cancel a partial command.

Input
Entered by a i A I o O c C s S R. Arbitrary text may then be entered. Input mode is normally terminated with ESC
character or abnormally with interrupt.

Last Line
Reading input for : / ? or ! ; terminate with CR to execute, interrupt to cancel.

COMMAND SUMMARY
Sample commands
- arrow keys move the cursor
- h j k l same as arrow keys
- itextESC insert text abc
- cwnewESC change word to new
- easESC pluralize word
- x delete a character
- dw delete a word
- dd delete a line
- 3dd ... 3 lines
- u undo previous change
- ZZ exit vi, saving changes
- :q!CR quit, discarding changes
- /textCR search for text
- "U "D scroll up or down
- :ex cmdCR any ex or ed command

Counts before vi commands
Numbers may be typed as a prefix to some commands. They are interpreted in one of these ways.

- line/column number  z G i
- scroll amount  "D "U
- repeat effect  most of the rest

Interrupting, canceling
ESC end insert or incomplete cmd
- ? (delete or rubout) interrupts
- "L reprint screen if "? scrambles it
- "R reprint screen if "L is key

File manipulation
- :wCR write back changes
- :qCR quit
- :q!CR quit, discard changes
- :e nameCR edit file name
- :e!CR reedit, discard changes
- :e + nameCR edit, starting at end
- :e +nCR edit starting at line n
- :e #CR edit alternate file
  synonym for :e #
- :w nameCR write file name
- :w! nameCR overwrite file name
- :shCR run shell, then return
- :!cmdCR run cmd, then return
- :nCR edit next file in arglist
- :n argsCR specify new arglist
- "G show current file and line
:ta tag\textasciicircum{CR} to tag file entry tag
ˆ] \textasciicircum{CR} :ta, following word is tag
In general, any \texttt{ex} or \texttt{ed} command (such as \texttt{substitute} or \texttt{global}) may be typed, preceded by a colon and followed by a CR.

Positioning within file

\texttt{^F} forward screen
\texttt{^B} backward screen
\texttt{^D} scroll down half screen
\texttt{^U} scroll up half screen
\texttt{G} go to specified line (end default)
\texttt{/pat} next line matching \texttt{pat}
\texttt{?pat} prev line matching \texttt{pat}
\texttt{n} repeat last / or ?
\texttt{N} reverse last / or ?
\texttt{/pat/+n} nth line after \texttt{pat}
\texttt{?pat?-n} nth line before \texttt{pat}
\texttt{]} next section/function
\texttt{[[} previous section/function
\texttt{(} beginning of sentence
\texttt{)} end of sentence
\texttt{\{} beginning of paragraph
\texttt{\}} end of paragraph
\texttt{%} find matching \texttt{() \{ or \}}

Adjusting the screen

\texttt{^L} clear and redraw
\texttt{^R} retype, eliminate @ lines
\texttt{z\textasciicircum{CR}} redraw, current at window top
\texttt{z-CR} \ldots at bottom
\texttt{z .CR} \ldots at center
\texttt{/pat/z-CR} \texttt{pat} line at bottom
\texttt{zn .CR} use \texttt{n} line window
\texttt{^E} scroll window down \texttt{1} line
\texttt{^Y} scroll window up \texttt{1} line

Marking and returning

\texttt{\ldots} move cursor to previous context
\texttt{\ldots} at first non-white in line
\texttt{mx} mark current position with letter \texttt{x}
\texttt{\ldots} move cursor to mark \texttt{x}
\texttt{\ldots} at first non-white in line

Line positioning

\texttt{H} top line on screen
\texttt{L} last line on screen
\texttt{M} middle line on screen
\texttt{+} next line, at first non-white
\texttt{-} previous line, at first non-white
\texttt{CR} return, same as \texttt{+}
or \texttt{j} next line, same column
or \texttt{k} previous line, same column
Character positioning
- first non white
0 beginning of line
$ end of line
h or forward
l or backwards
^H same as
space same as
fx find \(x\) forward
Fx f backward
tx upto \(x\) forward
Tx back up to \(x\)
; repeat last f F t or T
, inverse of ;
| to specified column
% find matching ( { ) or }

Words, sentences, paragraphs
w word forward
b back word
e end of word
) to next sentence
} to next paragraph
( back sentence
{ back paragraph
W blank delimited word
B back W
E to end of W

Commands for LISP Mode
) Forward s-expression
} ... but do not stop at atoms
( Back s-expression
} ... but do not stop at atoms

Corrections during insert
^H erase last character
^W erase last word
erase your erase, same as ^H
kill your kill, erase input this line
\ quotes ^H, your erase and kill
ESC ends insertion, back to command
? interrupt, terminates insert
^D backtab over autoindent
^D kill autoindent, save for next
0^D ... but at margin next also
^V quote non-printing character

Insert and replace
a append after cursor
i insert before cursor
A append at end of line
I insert before first non-blank
o open line below
O  open above
rx  replace single char with x
RtextESC  replace characters

Operators
Operators are followed by a cursor motion, and affect all text that would have been moved over. For example, since w moves over a word, dw deletes the word that would be moved over. Double the operator, e.g. dd to affect whole lines.

d  delete
c  change
y  yank lines to buffer
<  left shift
>  right shift
!  filter through command
=  indent for LISP

Miscellaneous Operations
C  change rest of line (c$)
D  delete rest of line (d$)
s  substitute chars (cl)
S  substitute lines (cc)
J  join lines
x  delete characters (dl)
X  ... before cursor (dh)
Y  yank lines (yy)

Yank and Put
Put inserts the text most recently deleted or yanked. However, if a buffer is named, the text in that buffer is put instead.
P  put back text after cursor
P  put before cursor
"xp  put from buffer x
"xy  yank to buffer x
"xd  delete into buffer x

Undo, Redo, Retrieve
u  undo last change
U  restore current line
.  repeat last change
"d p  retrieve d'th last delete

SEE ALSO
ex (1), crypt(1).
5000 and 7000 Series User Guide.

RESTRICTIONS
Software tabs using \textasciitilde T work only immediately after the autoindent.

Left and right shifts on intelligent terminals do not make use of insert and delete character operations in the terminal.

Some terminal arrow key definitions are identical to the control characters which permit positioning within the file (i.e. control-F, control-B, control-D, and control-U). Therefore, these positioning operations do not function as described, but instead perform an
arrow key operation.
NAME
vmstat - report virtual memory statistics

SYNOPSIS
vmstat [ -0lrs ] [ interval [ count ] ]

DESCRIPTION
7000 Series Systems only.
Vmstat delves into the system and normally reports certain statistics kept about process, virtual memory, disk, trap and cpu activity.
The 7000 systems can support up to 32 disk drives. Vmstat can only display 16 drives at one time. The -0 argument will display the first 16 mounted disk drives, and the -1 argument will display the second 16 mounted disk drives. These options take effect only if the system has more than 16 mounted drives. The -0 argument is the default. Any time during the display, you can type in a 0, 1 or r to change the display.
The -r argument, switches between displaying the first and second 16 mounted disk drives, i.e., it switches back and forth from the 0 and 1 options. This option takes effect only if the system has more than 16 mounted drives. The duration for each set of drives will be [interval]/2.
If given an -f argument, vmstat reports on the number of forks and vforks since system startup and the number of pages of virtual memory involved in each kind of fork. If given a -s argument, it instead prints the contents of the sum structure, giving the total number of several kinds of paging related events which have occurred since boot.
If none of these options are given, vmstat will report in the first line a summary of the virtual memory activity since the system has been booted. If interval is specified, then successive lines are summaries over the last interval seconds. "vmstat 5" will print what the system is doing every five seconds; this is a good choice of printing interval since this is how often some of the statistics are sampled in the system; others vary every second, running the output for a while will make it apparent which are recomputed every second. If a count is given, the statistics are repeated count times. The format fields are:
Procs: information about numbers of processes in various states.
r in run queue
b blocked for resources (i/o, paging, etc.)
w runnable or short sleeper (< 20 secs) but swapped
Memory: information about the usage of virtual and real memory. Virtual pages are considered active if they belong to processes which are running or have run in the last 20 seconds. A "page" here is 1024 bytes.
avm active virtual pages
fre  size of the free list

Page: information about page faults and paging activity. These are averaged each five seconds, and given in units per second.

re  pages reclaim less the free lists in the swap devices and the file system
at  pages of free lists in the swap devices and the file system
pi  pages paged in
po  pages paged out
fr  pages freed per second
de  anticipated short term memory shortfall
sr  pages scanned by clock algorithm, per-second

x/f/s: Disk operations per second (this field is system dependent). Typically paging will be split across several of the available drives. The number under each of these is the unit number.

Faults: trap/interrupt rate averages per second over last 5 seconds.

in  (non clock) device interrupts per second
sy  system calls per second
cs  cpu context switch rate (switches/sec)

Cpu: breakdown of percentage usage of CPU time

us  user time for normal and low priority processes
sy  system time
id  cpu idle

FILES
/dev/kmem, /unix

BUGS
There should be a screen oriented program which combines vmstat and ps(1) in real time as well as reporting on other system activity.
NAME
vpr - Versatec printer spooler

SYNOPSIS
vpr [ options ] [ files ]

DESCRIPTION
5000/20 and 5000/40 only.
Vpr causes the named files to be queued for printing on a Versatec printer. If no names appear, the standard input is assumed; thus vpr may be used as a filter.

OPTIONS
The following options may be given (each as a separate argument and in any order) before any file name arguments:

-c Make a copy of the file to be sent before returning to the user.
-r Remove the file after sending it.
-m When printing is complete, report that fact by mail(1).
-n Do not report the completion of printing by mail(1). This is the default option.

-ffile Use file as a dummy file name to report back in the mail. This is useful for distinguishing multiple runs, especially when vpr is being used as a filter.

-p [ -e raster ]
Use the plot filter vplot to output files produced by graph(1G). The -e option causes a previously scan converted file raster to be sent to the Versatec.

EXAMPLES
Two common uses are:
pr [ options ] file | vpr
and
graph [ options ] file | vpr -p

FILES
/etc/passwd user identification and accounting data
/usr/spool/vpd/* spool area
/usr/lib/vpd line printer daemon
/usr/lib/vpd.pr print filter
/usr/lib/vplot plot filter

SEE ALSO
graph(1G), mail(1), tplot(1G).
NAME
vs - report statistics of major subsystems

SYNOPSIS
vs [-h] [-u file] [ interval ]

DESCRIPTION
Vs reads selected kernel counters (such as vmmeter, vmtotal, etc.) to display an un-smoothed (instantaneous) interpretation of their contents on the screen. The screen is divided into three horizontal sub-windows. Each window is capable of displaying the statistics of a kernel subsystem. Currently, vs is capable of showing the following subsystems:

- Master Processor statistics ('M')
- Slave Processor statistics (7000-52 only) ('S')
- Memory subsystem statistics ('m')
- Paging subsystem statistics ('p')

One can place any of the above subsystems in any of the three sub-windows by typing the window number followed by the subsystem indicator (above paranthesized characters). If no window number is specified, then the third window is assumed. Typing a Control-L refreshes the screen, and typing 'f' freezes the screen updating for prolonged viewing.

The -h flag prints the usage and -u flag can be used to indicate a UNIX file other than the default /unix. If interval is specified on the command line, vs would update the screen every interval seconds.

The following is a brief description of each of the above subsystem's fields:

Master Processor

This general subsystem is divided into 4 parts: memory, paging, faults, and cpu usage. These parts contain:

- rm: Number of resident memory pages (text+data+stack).
- fre: Number of free memory pages.
- pf: Number of page faults per second.
- pi: Number of pages paged in per second.
- po: Number of pages paged out per second.
- re: Number of reclaimed pages per second.
- sr: Number of pages scanned by pageout daemon per second.
- fr: Number of pages freed per second.
- i: Number of interrupts per second.
- sy: Number of system calls per second.
- tr: Number of traps per second.
- cs: Number of context switches per second.
us: Percentage of CPU time spent in user mode.
ni: Percentage of CPU spent in positively niced processes (p_nice > NZERO).
sy: Percentage of CPU time spent in system mode.
id: Percentage of CPU time spent in idle loop.

Slave Processor (7000-52 only)

This subsystem is only relevant to the 7000-52 configuration. The fields are:

sy: Number of processes per second leaving Slave due to system calls.
qu: Number of processes per second leaving Slave due to completing quantum service time.
tr: Number of processes per second leaving Slave due to traps.
si: Number of processes per second leaving Slave due to pending signals.
tcks: Average number of ticks executed on Slave per process.
stl: Number of processes stolen by Master per second.
sy: Number of what is believed to be read-only system calls per second.
cs: Number of context switches on Slave per second.
ns: Number of times Slave has been suspended in the last interval.
ds: Duration of suspension in micro-seconds within the past interval.

up: Percentage of Slave CPU in user mode.
ni: Percentage of Slave in niced mode.
sy: Percentage of Slave in system mode.
id: Percentage of Slave being idle.

Memory Subsystem

This subsystem is divided into two parts, virtual statistics and resident statistics. These fields are:

vmtxt: Number of virtual text pages.
vmdat: Number of virtual data pages.
avtxt: Number of active virtual text pages.
avdat: Number of active virtual data pages.
rmtxt: Number of resident text pages.
rmdat:
Number of resident data pages.

artxt:
    Number of active resident text pages.

ardat:
    Number of active resident data pages.

Paging subsystem

This subsystem is consisted of two parts, page-in and page-out. These fields are:

pf: Number of page faults per second on Master.
sl: Number of "false" page faults per second on Slave.
re: Number of pages reclaimed.
pi: Number of page-ins per second.
it: Number of page faults on intransit pages.
fl: Number of pages reclaimed from free list.
sw: Number of pages reclaimed from free list rather than on swap device.
in: Number of pages reclaimed from free list rather than in file system.
png: Number of pages paged-in.
ex: Number of executable filled-on-demand pages paged in.
zf: Number of zero fill-on-demand pages created.
rx: Number of page faults on executable fill-on-demand pages.
zn: Number of page faults on zero-fill-on-demand pages.
rv: Number of hand revolutions of pageout daemon.
sc: Number of pages scanned by pageout daemon per second.
fr: Number of pages freed by pageout daemon per second.
po: Number of page-outs per second.
pg: Number of pages paged-out per second.

FILES
/unix, /dev/kmem

SEE ALSO
    vmstat(1), uptime(1), iostat(1), sar(1)
[This page left blank.]
NAME
  w - who is on and what they are doing

SYNOPSIS
  w [ -h ] [ -s ] [ user ]

DESCRIPTION
  7000 Series Systems only.

  W prints a summary of the current activity on the system, including what each user is doing. The heading line shows the current time of day, how long the system has been up, the number of users logged into the system, and the load averages. The load average numbers give the number of jobs in the run queue averaged over 1, 5 and 15 minutes.

  The fields output are: the users login name, the name of the tty the user is on, the time of day the user logged on, the number of minutes since the user last typed anything, the CPU time used by all processes and their children on that terminal, the CPU time used by the currently active processes, the name and arguments of the current process.

  The -h flag suppresses the heading. The -s flag asks for a short form of output. In the short form, the tty is abbreviated, the login time and cpu times are left off, as are the arguments to commands. -l gives the long output, which is the default.

  If a user name is included, the output will be restricted to that user.

FILES
  /etc/utmp
  /dev/kmem
  /dev/drum

SEE ALSO
  who(1), ps(1)

BUGS
  The notion of the "current process" is muddy. The current algorithm is "the highest numbered process on the terminal that is not ignoring interrupts, or, if there is none, the highest numbered process on the terminal". This fails, for example, in critical sections of programs like the shell and editor, or when faulty programs running in the background fork and fail to ignore interrupts. (In cases where no process can be found, w prints "-".)

  The CPU time is only an estimate, in particular, if someone leaves a background process running after logging out, the person currently on that terminal is "charged" with the time.

  Background processes are not shown, even though they account for much of the load on the system.

  Sometimes processes, typically those in the background, are printed with null or garbaged arguments. In these cases, the name of the command is printed in parentheses.
W does not know about the new conventions for detection of background jobs. It will sometimes find a background job instead of the right one.
NAME
   wait - await completion of process

SYNOPSIS
   wait

DESCRIPTION
   Wait until all processes started with & have completed, and report on
   abnormal terminations.

   Because the wait(2) system call must be executed in the parent
   process, the shell itself executes wait, without creating a new pro-
   cess.

SEE ALSO
   sh(1), wait(2).

RESTRICTIONS
   Not all the processes of a 3- or more-stage pipeline are children of
   the shell, and thus cannot be waited for.
[This page left blank.]
NAME
   wc - word count

SYNOPSIS
   wc [ -lwc ] [ files ]

DESCRIPTION
   Wc counts lines, words, and characters in the named files, or in
   the standard input if no files are specified. It also keeps a total
   count for all named files.

   A word is a string of characters delimited by spaces, tabs, or
   new-lines.

   When files are specified on the command line, they are printed
   along with the counts.

OPTIONS
   The following options can be used in any combination:
   -l   Report number of lines only.
   -w   Report number of words only.
   -c   Report number of characters only.
NAME
what - identify SCCS files

SYNOPSIS
what [ -s ] files

DESCRIPTION
What searches the given files for all occurrences of the pattern that get(1) substitutes for %Z% (this is @(#) ) and prints out what follows until the first ",", >, new-line, \ or null character. For example, if the C program in file f.c contains

```c
char ident[] = "@(#)identification information ";
```
and f.c is compiled to yield f.o and a.out, then the command

```bash
what f.c f.o a.out
```
prints

```bash
f.c: identification information
f.o: identification information
a.out: identification information
```

What is intended to be used in conjunction with the -lSCCS command get(1), which automatically inserts identifying information, but what can also be used where the information is inserted manually.

OPTION
-s Quit after finding the first occurrence of pattern in each file.

SEE ALSO
get(1), help(1).

DIAGNOSTICS
Use help(1) for explanations.

ENT CODES
0 Match
1 No match

RESTRICTIONS
An unintended occurrence of the pattern @(#) may be found just by chance, but this is harmless in most cases.
NAME
whereis - locate source, binary, and or manual for program

SYNOPSIS
whereis [ -sbm ] [ -u ] [ -SBM dir ... -f ] file ...

DESCRIPTION
Whereis locates source, binary, and manual sections for specified files.
Whereis first strips the supplied names of leading pathname components and any (single) trailing extension of the form .ext, e.g. .c. Prefixes of s. resulting from use of source code control are also stripped.
Whereis then attempts to locate the desired program in a list of standard places.

OPTIONS
One or two of the restrictive options, -b, -s, and -m, may be specified.

-b Search only for binary sections.
-s Search only for source sections.
-m Search only for manual sections.
-B, -S, and -M
Change or otherwise limit the places where whereis searches; dir must be a full pathname.
-f Terminate each directory list and signal the start of file names.
-u Search for unusual entries. A file is said to be unusual if it does not have one entry of each requested type. Thus whereis -m -u * asks for those files in the current directory which have no manual section.

EXAMPLE
The following example finds all the files in /usr/bin which are not documented in /usr/catman/u_man/man1 with source in /usr/src/cmd:

    cd /usr/ucb
    whereis -u -M /usr/catman/u_man/man1 -S /usr/src/cmd -f *

FILES
/usr/src/*
/usr/catman/*
/lib, /etc, /usr/{lib, bin, ucb}

RESTRICTIONS
Because whereis uses chdir(2) to run faster, pathnames given with the -M, -S, and -B must be full pathnames; that is, they must begin with a /.
NAME
who - who is on the system

SYNOPSIS
who [-abdHlpqrstTu] [ file ]
who am i
who am I

DESCRIPTION
Who lists the user name, terminal line, login time, elapsed time since activity occurred on the line, and the process-ID of the command interpreter (shell) for each current UNIX system user. Who examines the /etc/utmp file to obtain its information. If file is given, that file is examined. Usually, file is /etc/wtmp, which contains a history of all the logins since the file was last created.

Who am i or who am I identifies the invoking user.

Except for the -s option (which is assumed if no options are specified), the general format for output entries is:

name [state] line time activity pid [comment] [exit]

OPTIONS
With options, who lists logins, logoffs, reboots, and changes to the system clock, as well as other processes spawned by the init process. These options are:

-a Process /etc/utmp or the named file using all of the options.
-A Process /etc/utmp or the named file, displaying records having ut_type = ACCOUNTING See utmp(4).
-b Indicate the time and date of the last reboot.
-d Display all processes that have expired and have not been respawned by init. The exit field appears for dead processes and contains the termination and exit values as returned by wait(2) of the dead process. This option can be useful in determining why a process terminated.
-H Print column headings above the regular output.
-l List only those lines on which the system is waiting for someone to login. The name field is LOGIN in such cases. Other fields are the same as for user entries except that the state field does not exist.
-p List any other process which is currently active and has been previously spawned by init. The name field is the name of the program executed by init as found in /etc/inittab. The state, line, and activity fields have no meaning. The comment field shows the id field of the line from /etc/inittab that spawned this process. See inittab(4).
-q Display only the names and number of users currently logged on. When this option is used, all other options are ignored.
-r Indicate the current run-level of the init process.
-s List only the name, line, and time fields (default).

-t Indicate the last change to the system clock (by the date(1) command) by root. See su(1).

-T Print the state of the terminal line as well as all fields requested by the -u option. The state describes whether someone else can write to that terminal. A + indicates the terminal is writable by anyone; a - indicates it is not. Root can write to all lines having a + or a - in the state field. If a bad line is encountered, a ? is printed.

-u List the following information about those users who are currently logged in:

   name
   User login name.

   line
   Line name as found in the directory /dev.

   time
   Time that the user logged in.

   activity
   Number of hours and minutes since activity last occurred on that particular line. A dot (.) indicates that the terminal has seen activity in the last minute. If more than twenty-four hours have elapsed or the line has not been used since boot time, the entry is marked old. This information is useful when trying to determine if a user is working at the terminal.

   pid
   Process-ID of the user shell.

   comment
   Comment field associated with this line as found in /etc/inittab (see inittab(4)). This comment may contain the location of the terminal, the telephone number of the dataset, type of terminal if hard-wired, etc.

FILES
/etc/utmp
/etc/wtmp
/etc/inittab

SEE ALSO
date(1), init(1M), login(1), msg(1), su(1), wait(2), inittab(4), utmp(4).
NAME
    whoami - print effective current user id

SYNOPSIS
    whoami

DESCRIPTION
    Whoami prints who you are. It works even if you are su’d, while
    'who am i' does not since it uses /etc/utmp.

FILES
    /etc/passwd

SEE ALSO
    who (1)
NAME
write - write to another user

SYNOPSIS
write user [ line ]

DESCRIPTION
Write copies lines from your terminal to the terminal of another user. When first called, write sends the message

Message from yourname (tty??) [ date ]...

to the person you want to talk to. When the connection is completed, two bells are sent to your own terminal. Enter your message and press the newline or return key. Pressing the newline or return key at the end of entering your message sends the message. Each time you enter a message and press the newline key, write sends the message. The recipient of the message should use write to respond to your message.

The following protocol is suggested for using write. When you first write to another user, wait for them to write back before starting to send. Each person should end a message with a distinctive signal, for example (o) for over, so that the other person knows when to reply. The signal (oo) for over and out is suggested when conversation is to be terminated.

Communication continues until you enter an end of file (control-d) from the terminal or until write receives an interrupt or the recipient issues a mesg(1) command to deny messages. At that point, write writes EOT on the other terminal and exits.

If you want to write to a user who is logged in more than once, the line argument may be used to indicate which line or terminal is to receive your message, for example, tty00. Otherwise, the first instance of the user found in /etc/utmp is assumed and the following message is displayed:

User is logged on more than one place.
You are connected to terminal.
Other locations are: terminal

Permission to write may be denied or granted by use of the mesg(1) command. Writing to others is normally allowed by default. Certain commands, in particular nroff(1) and pr(1) disallow messages in order to prevent interference with their output. However, if the user has superuser permissions, write forces messages onto a write inhibited terminal.

If the character ! is found at the beginning of a line, write calls the shell to execute the rest of the line as a command.

FILES
/etc/utmp to find user
/bin/sh to execute!

SEE ALSO
mail(1), mesg(1), nroff(1), pr(1), sh(1), who(1).
### DIAGNOSTICS

**User not logged on**

The person you are trying to write to is not logged in. Use `who(1)` to determine who is logged in or `mailx(1)` to mail a message to the user who is not logged in.

**Permission denied**

The person you are trying to write to denies that permission with `mesg(1)`. Try using `mailx(1)`.

**Warning: cannot respond, set mesg y**

Your terminal is set to `mesg n` and the recipient cannot respond to you. Enter `!mesg y` to accept messages.

**Can no longer write to user**

The recipient denied permission (mesg n) after you started writing.
NAME
xargs - construct argument list(s) and execute command

SYNOPSIS
xargs [ options ] [ command [ initial-arguments ] ]

DESCRIPTION
Xargs combines the fixed initial-arguments with arguments read from standard input to execute the specified command one or more times. The number of arguments read for each command invocation and the manner in which they are combined are determined by the options specified.

Xargs searches for command, which may be a shell file, using the $PATH of the user. If command is omitted, xargs uses /bin/echo.

Arguments read in from standard input are defined to be contiguous strings of characters delimited by one or more blanks, tabs, or new-lines; xargs discards empty lines. Blanks and tabs may be embedded as part of an argument if escaped or quoted: Characters enclosed in quotes (single or double) are taken literally, and the delimiting quotes are removed. A backslash escapes the next character if the backslash does not appear in a quoted string.

Xargs constructs each argument list starting with the initial-arguments, followed by some number of arguments read from standard input (Exception: see -l option). Options -l, -i, and -n determine how arguments are selected for each command invocation. When none of these options are specified, the initial-arguments are followed by arguments read continuously from standard input until an internal buffer is full, then xargs executes command with the accumulated args. This process is repeated until there are no more args.

Xargs terminates if it receives a return code of -1 from, or if it cannot execute, command. When command is a shell program, it should explicitly exit (see sh(1)) with an appropriate value to avoid accidentally returning with -1.

OPTIONS
When there are option conflicts (e.g., -i vs. -n), the last option has precedence.

-lnumber Execute command for each non-empty number lines of arguments from standard input.

The last invocation of command will have fewer lines of arguments if fewer than number remain.

A line is considered to end with the first new-line unless the last character of the line is a blank or a tab; a trailing blank or tab signals continuation through the next non-empty line.

If number is omitted, 1 is assumed.
Option -i forces option -x.

-ireplstr

(Insert mode) Execute command for each line from standard input, taking the entire line as a single arg, and inserting it in initial-arguments for each occurrence of replstr. A maximum of 5 arguments in initial-arguments may each contain one or more instances of replstr.

Xargs strips blanks and tabs from the beginning of each line.

Constructed arguments may become at most 255 characters long.

Option -i forces option -x.

{} is assumed for replstr if not specified.

-nnumber

Execute command using as many standard input arguments as possible, up to number arguments maximum. Fewer arguments are used if their total size is greater than size characters, and if there are fewer than number arguments remaining on the last invocation.

If option -x is also specified, each number argument must fit in the size limitation, else xargs terminates execution.

-t

(Trace mode) Echo the command and each constructed argument list to file descriptor 2 just prior to their execution.

-p

(Prompt mode) Prompt the user whether to execute command before each invocation.

Trace mode (-t) prints the command instance to be executed, followed by a ?•• prompt. A reply of y (followed by anything) executes the command; any other response, including pressing the carriage return, skips that particular invocation of command.

-x

Terminate xargs if any argument list would be greater than size characters;

-x is forced by the options -i and -1. When neither of the options -i, -1, or -n are coded, the total length of all arguments must be within the size limit.

-ssize

Set the maximum total size of each argument list to size characters; size must be a positive integer less than or equal to 470. If -s is not coded, 470 is assumed.

Note that the character count for size includes one
extra character for each argument and the count of characters in the command name.

\texttt{-eeofstr}

Designate \texttt{eofstr} as the logical end-of-file string.

Underscore (\_\_) is assumed for the logical EOF string if \texttt{-e} is not specified.

\texttt{-e} with no \texttt{eofstr} specified turns off the logical EOF string capability (underscore is taken literally).

\texttt{Xargs} reads standard input until either end-of-file or the logical EOF string is encountered.

**EXAMPLES**

This example moves all files from directory \$1 to directory \$2, and echos each move command just before doing it:

\begin{verbatim}
ls $1 \pipe xargs -i -t mv $1/{ } $2/{ }
\end{verbatim}

This example combines the output of the parenthesized commands onto one line, which is then appended to the file \texttt{log}:

\begin{verbatim}
(logname; date; echo $0 $*) \pipe xargs >>log
\end{verbatim}

This example asks the user which files in the current directory are to be archived and archives them into \texttt{arch} (1.) one at a time, or (2.) many at a time.

1. \texttt{ls \pipe xargs -p -I \pipe xargs ar r arch}
2. \texttt{ls \pipe xargs -p -I \pipe xargs ar r arch}

This example executes \texttt{diff(1)} with successive pairs of arguments originally typed as shell arguments:

\begin{verbatim}
echo $* \pipe xargs -n2 diff
\end{verbatim}

**SEE ALSO**

\texttt{sh(1)}.

**DIAGNOSTICS**

Self explanatory.
NAME
xstr - extract strings from C programs to implement shared strings

SYNOPSIS
xstr [-c] [-] [file]

DESCRIPTION
Xstr maintains a file strings into which strings in component parts of a large program are hashed. These strings are replaced with references to this common area. This serves to implement shared constant strings, most useful if they are also read-only. Xstr reads from its standard input when the argument '-' is given.

The command

xstr -c name

extracts the strings from the C source in name, replacing string references by expressions of the form (&xstr[number]) for some number. An appropriate declaration of xstr is prepended to the file. The resulting C text is placed in the file x.c, to then be compiled. The strings from this file are placed in the strings data base if they are not there already. Repeated strings and strings which are suffixes of existing strings do not change the data base.

After all components of a large program have been compiled, a file xs.c declaring the common xstr space can be created by a command of the form

xstr

This xs.c file should then be compiled and loaded with the rest of the program. If possible, the array can be made read-only (shared), saving space and swap overhead.

Xstr can also be used on a single file. A command

xstr name

creates files x.c and xs.c as before, without using or affecting any strings file in the same directory.

It may be useful to run xstr after the C preprocessor if any macro definitions yield strings or if there is conditional code which contains strings which may not, in fact, be needed. An appropriate command sequence for running xstr after the C preprocessor is:

cc -E name.c | xstr -c -
cc -c x.c
mv x.o name.o

Xstr does not touch the file strings unless new items are added, thus make can avoid remaking xs.o unless truly necessary.

FILES
strings Data base of strings
x.c Massaged C source
xs.c C source for definition of array xstr
/tmp/xs* Temp file when xstr name does not touch strings
SEE ALSO
mkstr(1)

WARNING
If a string is a suffix of another string in the data base, but the shorter string is seen first by xstr both strings are placed in the data base, when just placing the longer one there suffices.
NAME
yacc - yet another compiler-compiler

SYNOPSIS
yacc [-vdlit] grammar

DESCRIPTION
Yacc converts a context-free grammar into a set of tables for a simple automaton which executes an LR(1) parsing algorithm. The grammar may be ambiguous; yacc uses specified precedence rules to break ambiguities.

The output file, y.tab.c, must be compiled by the C compiler to produce a program yyparse. This program must be loaded with the lexical analyzer program, yylex, as well as main and yyerror, an error handling routine. These routines must be supplied by the user; lex(1) is useful for creating lexical analyzers usable by yacc.

Yacc always generates runtime debugging code in y.tab.c under conditional compilation control. This code is normally not included when y.tab.c is compiled. See the -t option below. The runtime debugging code is under the control of YYDEBUG, a pre-processor symbol. If YYDEBUG has a non-zero value, then the debugging code is included. If its value is zero, then the code is not included. The size and execution time of a program produced without the runtime debugging code is smaller and slightly faster.

OPTIONS
-v Prepare the file y.output, which describes the parsing tables and reports conflicts generated by ambiguities in the grammar.
-d Generate the file y.tab.h with the #define statements that associate the yacc-assigned token codes with the user-declared token names. This option allows source files other than y.tab.c to access the token codes.
-l Produce code in y.tab.c which does not contain any #line constructs. This option should only be used after the grammar and the associated actions are fully debugged.
-t Include runtime debugging code when y.tab.c is compiled.

FILES
y.output
y.tab.c
y.tab.h defines for token names
yacc.tmp temporary file
yacc.debug temporary file
yacc.acts temporary file
/usr/lib/yaccpar parser prototype for C programs

SEE ALSO
lex(1), malloc(3x).
Yet Another Compiler Compiler (yacc) in the Support Tools Guide.
DIAGNOSTICS
The number of reduce-reduce and shift-reduce conflicts is reported on the standard error output; a more detailed report is found in the y.output file. Similarly, yacc also reports any rules that could not be reached from the start symbol in y.output.

RESTRICTIONS
Because file names are fixed, at most one yacc process can be active in a given directory at a time.
NAME
   intro - introduction to games

DESCRIPTION
   This section describes the recreational and educational programs
   found in the directory /usr/games.

   The availability of these programs may vary from system to system.
[This page left blank.]
NAME
arithmetic - provide drill in arithmetic problems

SYNOPSIS
/usr/games/arithmetic [ +-x/ ] [ range ]

DESCRIPTION
Arithmetic types out simple arithmetic problems, and waits for an answer to be entered.

If the answer is correct, it replies

Right!

and prints a new problem.

If the answer is wrong, it replies

What?

and waits for another answer.

Every twenty problems, arithmetic publishes statistics on correctness and the time required to answer.

To quit the program, enter an interrupt (delete).

The program does not give correct answers, since the learner should, in principle, be able to calculate them.

For almost all users, the relevant statistic should be time per problem, not percent correct.

ARGUMENTS
One or more of the following characters specifies the type of problem to be generated. If more than one is given, the different types of problems are mixed in random order; default is +-.

+ generate addition problems
- generate subtraction problems
x generate multiplication problems
/ generate division problems

Range is a decimal number; all operands and answers are less than or equal to the value of range. Default range is 10.

At the start, all numbers less than or equal to range are equally likely to appear. If the respondent makes a mistake, the numbers in the problem which was missed become more likely to reappear.
[This page left blank.]
NAME
back - the game of backgammon

SYNOPSIS
/usr/games/back

DESCRIPTION
Back is a program which provides a partner for the game of backgammon. It is designed to play at three different levels of skill, one of which you must select.

In addition to selecting the level of your opponent, you may also indicate that you would like to roll your own dice during your turns.

You are also given the opportunity to move first. The practice of each player rolling one die for the first move is not incorporated.

The points are numbered 1-24: 1 is the extreme inner table of white, 24 is the inner table of brown, 0 is the bar for removed white pieces, and 25 is the bar for brown.

For details on how moves are expressed, type y when back asks

Instructions?

at the beginning of the game. When back first asks

Move?

type a question mark to see a list of move options other than entering your numerical move.

When the game is finished, back asks you if you want the log. If you respond with y, back attempts to append to or create a file back.log in the current directory.

FILES
/usr/games/lib/backrules
/rules file
/tmp/back* 
log temp file
back.log 
log file

RESTRICTIONS
Back complains loudly if you attempt to make too many moves in a turn, but becomes very silent if you make too few. Doubling is not implemented.
[This page left blank.]
NAME
bj - the game of black jack

SYNOPSIS
/usr/games/bj

DESCRIPTION
Bj is a serious attempt at simulating the dealer in the game of black jack (or twenty-one). The following rules apply:

The bet is $2 every hand.

A player natural (black jack) pays $3. A dealer natural loses $2. Both dealer and player naturals is a push (no money exchange).

If the dealer has an ace up, the player is allowed to make an insurance bet against the chance of a dealer natural. If this bet is not taken, play resumes as normal. If the bet is taken, it is a side bet where the player wins $2 if the dealer has a natural and loses $1 if the dealer does not.

If the player is dealt two cards of the same value, he is allowed to double. He is allowed to play two hands, each with one of these cards. (The bet is doubled also; $2 on each hand.)

If a dealt hand has a total of ten or eleven, the player may double down. He may double the bet ($2 to $4) and receive exactly one more card on that hand.

Under normal play, the player may hit (draw a card) as long as his total is not over twenty-one. If the player busts (goes over twenty-one), the dealer wins the bet.

When the player stands (decides not to hit), the dealer hits until he attains a total of seventeen or more. If the dealer busts, the player wins the bet.

If both player and dealer stand, the one with the largest total wins. A tie is a push.

The machine deals and keeps score. The following questions are asked at appropriate times. Each question is answered by y followed by a return for yes, or just return for no.

? (do you want a hit?)
Insurance?
Double down?

Every time the deck is shuffled, the dealer so states and the action (total bet) and standing (total won or lost) is printed.

To exit, hit the interrupt key (DEL); bj prints the action and standing.
[This page left blank.]
NAME
chess - the game of chess

SYNOPSIS
/usr/games/chess

DESCRIPTION
Not on 5000/20/40/50.

Chess is a computer program that plays class D chess. Moves may be given either in standard (descriptive) notation or in algebraic notation. The symbol + must be placed at the end of a line when the move on that line places the opponent's king in check. The values o-o and o-o-o specify castling, king side or queen side, respectively.

The user is prompted for a move or command by a *. To play black, type first at the onset of the game. To print a copy of the board in play, type a carriage return only. Each move is echoed in the appropriate notation, followed by the program's reply. Near the middle and end games, the program can take considerable time in computing its moves.

A ? or help may be typed to get a help message that briefly describes the possible commands.

DIAGNOSTICS
The most cryptic diagnostic is "eh?" which means that the input was syntactically incorrect.

RESTRICTIONS
Pawns may be promoted only to queens.
NAME
  craps - the game of craps

SYNOPSIS
  /usr/games/craps

DESCRIPTION
  Craps is a form of the game of craps that is played in Las Vegas. The program simulates the roller, while the user (the player) places bets. The player may choose, at any time, to bet with the roller or with the House. A bet of a negative amount is taken as a bet with the House, any other bet is a bet with the roller.

The player starts off with a bankroll of $2,000.

The program prompts with:

  bet?

The bet can be all or part of the bankroll of the player. Any bet over the total bankroll is rejected and the program prompts with bet? until a proper bet is made.

Once the bet is accepted, the roller throws the dice. The following rules apply (the player wins or loses depending on whether the bet is placed with the roller or with the House; the odds are even). The first roll is the roll immediately following a bet:

1. On the first roll:
   7 or 11  wins for the roller;
   2, 3, or 12 wins for the House;
   any other number is the point, roll again (Rule 2 applies).

2. On subsequent rolls:
   point  roller wins;
   7   House wins;
   any other number roll again.

If a player loses the entire bankroll, the House offers to lend the player an additional $2,000. The program prompts:

  marker?

A yes (or y) consummates the loan. Any other reply terminates the game.

If a player owes the House money, the House reminds the player, before a bet is placed, how many markers are outstanding.

If, at any time, the bankroll of a player who has outstanding markers exceeds $2,000, the House asks:

  Repay marker?

A reply of yes (or y) indicates the willingness of the player to repay the loan. If only 1 marker is outstanding, it is immediately repaid. However, if several markers are outstanding, the House asks:
How many?

markers the player would like to repay. If an invalid number is entered (or just a carriage return), an appropriate message is printed and the program prompts with How many? until a valid number is entered.

If a player accumulates 10 markers (a total of $20,000 borrowed from the House), the program informs the player of the situation and exits.

Should the bankroll of a player who has outstanding markers exceed $50,000, the total amount of money borrowed is automatically repaid to the House.

Any player who accumulates $100,000 or more breaks the bank. The program then prompts:

   New game?

to give the House a chance to win back its money.

Any reply other than yes is considered to be a no (except in the case of bet? or How many?). To exit, send an interrupt (break), DEL, or control-d. The program indicates whether the player won, lost, or broke even.

NOTES

The random number generator for the die numbers uses the seconds from the time of day. Depending on system usage, these numbers, at times, may seem strange but occurrences of this type in a real dice situation are not uncommon.
NAME
    hangman - guess the word
SYNOPSIS
    /usr/games/hangman [ arg ]
DESCRIPTION
    Not on 5000/20/40/50.
    
    *Hangman* chooses a word at least seven letters long from a dictionary. The user is to guess letters one at a time.

    The optional argument *arg* names an alternate dictionary.
FILES
    /usr/lib/w2006
RESTRICTIONS
    Hyphenated compounds are run together.
[This page left blank.]
NAME
jotto - secret word game

SYNOPSIS
/usr/games/jotto [-p]

DESCRIPTION
Not on 5000/20/40/50.

Jotto is a word guessing game. You try to guess the computer's secret word before it guesses yours. Clues are obtained by entering probe words. For example, if the computer's secret word is brown and you probe with stare, it replies 1 indicating that there is one letter in common between your probe and the secret word. Double letters count only once unless they appear in both words. For example, if the hidden word is igloo and you probe with broke, the computer replies 1. If you probe with gloom, the computer responds 4. All secret words and probe words should be non-proper English five-letter words. If the computer guesses your word exactly, please respond with y. It then tells you what its secret word was.

The -p option instructs the computer to report its progress in guessing your word.

RESTRICTIONS
The dictionary contains some unusual words and lacks some common ones.
[This page left blank.]
NAME
   maze - generate a maze

SYNOPSIS
   /usr/games/maze

DESCRIPTION
   Maze asks a few questions and then prints a maze.

RESTRICTIONS
   Some mazes (especially small ones) have no solutions.
NAME
moo - guessing game

SYNOPSIS
/usr/games/moo

DESCRIPTION
Moo is a guessing game imported from England. The computer picks a number consisting of four distinct decimal digits. The player guesses four distinct digits, being scored on each guess.

A cow is a correct digit in an incorrect position.

A bull is a correct digit in a correct position.

The game continues until the player guesses the number (a score of four bulls).
[This page left blank.]
NAME
quiz - test your knowledge

SYNOPSIS
/usr/games/quiz [ -i file ] [ -t ] [ category1 category2 ]

DESCRIPTION
Not on 5000/20/40/50.

Quiz gives associative knowledge tests on various subjects. It asks items chosen from category1 and expects answers from category2, or vice versa. If no categories are specified, quiz gives instructions and lists the available categories.

Quiz tells a correct answer whenever you type a bare new-line. At the end of input, upon interrupt, or when questions run out, quiz reports a score and terminates.

The -t option specifies tutorial mode, where missed questions are repeated later, and material is gradually introduced as you learn.

The -i option causes the named file to be substituted for the default index file. The lines of these files have the syntax:

line = category new-line | category : line
category = alternate | category | alternate
alternate = empty | alternate primary
primary = character | [ category ] | option
option = { category }

The first category on each line of an index file names an information file. The remaining categories specify the order and contents of the data in each line of the information file. Information files have the same syntax. Backslash \ is used as with sh(1) to quote syntactically significant characters or to insert transparent new-lines into a line. When either a question or its answer is empty, quiz refrains from asking it.

FILES
/usr/games/lib/quiz/index
/usr/games/lib/quiz/*

RESTRICTIONS
The construct a\;ab does not work in an information file. Use a{b}.
[This page left blank.]
NAME
reversi - a game of dramatic reversals

SYNOPSIS
/usr/games/reversi [ [ -r ] file ]

DESCRIPTION
Not on 5000/20/40/50.

Reversi (also known as friends, Chinese friends, and Othello) is
played on an 8 by 8 board using two-sided tokens. Each player
takes his turn by placing a token with his side up in an empty
square. During the first four turns, players may only place
tokens in the four central squares of the board. Subsequently,
with each turn, a player must capture one or more of his
opponent's tokens. He does this by placing one of his tokens such
that it and another of his tokens embrace a solid line of his
opponent's tokens horizontally, vertically or diagonally. Captured
tokens are flipped over and thus can be re-captured. If a player
cannot outflank his opponent, he forfeits his turn. The play con­
tinues until the board is filled or until no more outflanking is possi­
ble.

In this game, your tokens are asterisks (*) and the machine's are
at-signs (@). You move by typing in the row and column at which
you want to place your token as two digits (1-8), optionally
separated by blanks or tabs. You can also type in:
c To continue the game after hitting break (this is only neces­
sary if you interrupt the machine while it is deliberating).
g n To start reversi playing against itself for the next n moves (or
until the break key is pressed).

n To stop printing the board after each move.
o To start it up again.
p To print the board regardless.
q To quit (without dishonor).
s To print the score.
! To escape to the shell. Control-d gets you back.

Reversi also recognizes several commands which are valid only at
the start of the game, before any moves have been made. They are:
f To let the machine go first.
h n To ask for a handicap of from one to four corner squares. If
you are really good, you can give the machine a handicap by
typing a negative number.
l n To set the amount of look-ahead used by the machine in
searching for moves. Zero means none at all. Four is the
default. Greater than six means you may fall asleep waiting
for the machine to move.
t n To tell reversi that you only need n seconds to consider each
move. If you fail to respond in the allotted time, you forfeit
your turn.

If reversi is given a file name as an argument, it checkpoints the
game, move by move, by dumping the board onto file. The -r
option causes \textit{reversi} to restart the game from file and continue logging.

\textbf{DIAGNOSTICS}

\textit{Illegal!} for an invalid move, and \textit{Huh?} for a move that even the machine cannot understand.
NAME
  rogue - Exploring The Dungeons of Doom

SYNOPSIS
  rogue [ -r ] [ save_file ] [ -s ]

DESCRIPTION
  Rogue is a computer fantasy game with a new twist. It is CRT-
  oriented and the object of the game is to survive the attacks of
  various monsters and get a lot of gold, rather than the puzzle solv-
  ing orientation of most computer fantasy games.

  To get started you really only need to know two commands. The
  command ? will give you a list of the available commands and the
  command / will identify the things you see on the screen.

  To win the game (as opposed to merely playing to beat other
  people's high scores) you must locate the Amulet of Yendor which
  is somewhere below the 20th level of the dungeon and get it out.
  Nobody has achieved this yet and if somebody does, they will prob-
  ably go down in history as a hero among heroes.

  When the game ends, either by your death, when you quit, or if
  you (by some miracle) manage to win, rogue will give you a list of
  the top ten scorers. The scoring is based entirely upon how much
  gold you get. There is a 10% penalty for getting yourself killed.

  If save_file is specified, rogue will be restored from the specified
  saved game file. If the -r option is used, the game save_file is
  presumed to be the default.

  The -s option will print out the list of scores.

  For more detailed directions, read the document A Guide to the
  Dungeons of Doom.

FILES
  /usr/games/lib/rogue_roll  Score file
  ~/rogue.save  Default save file

SEE ALSO
  Michael C. Toy, A Guide to the Dungeons of Doom

BUGS
  Probably infinite, however, that Floating Eyes sometimes transfix
  you permanently is not a bug. It's a feature. !Funky!Stuff! echo
  extracting -rogue.doc sed 's/\`X/\`' >rogue.doc <<'!Funky!Stuff!'

A Guide to the Dungeons of Doom

Michael C. Toy
Kenneth C. R. C. Arnold

Computer Systems Research Group
Department of Electrical Engineering and Computer Science
University of California
Berkeley, California 94720

Rogue is a visual CRT-based fantasy game which runs under the UNIX timesharing system. This paper describes how to play rogue, and gives a few hints for those who might otherwise get lost in the Dungeons of Doom.
You have just finished your years as a student at the local fighter's guild. After much practice and sweat you have finally completed your training and are ready to embark upon a perilous adventure. As a test of your skills, the local guildmasters have sent you into the Dungeons of Doom. Your task is to return with the Amulet of Yendor. Your reward for the completion of this task will be a full membership in the local guild. In addition, you are allowed to keep all the loot you bring back from the dungeons. In preparation for your journey, you are given an enchanted mace, a bow, and a quiver of arrows taken from a dragon's hoard in the far off Dark Mountains. You are also outfitted with elf-crafted armor and given enough food to reach the dungeons. You say goodbye to family and friends for what may be the last time and head up the road. You set out on your way to the dungeons and after several days of uneventful travel, you see the ancient ruins that mark the entrance to the Dungeons of Doom. It is late at night, so you make camp at the entrance and spend the night sleeping under the open skies. In the morning you gather your mace, put on your armor, eat what is almost your last food, and enter the dungeons. You have just begun a game of rogue. Your goal is to grab as much treasure as you can, find the Amulet of Yendor, and get out of the Dungeons of Doom alive. On the screen, a map of where you have been and what you have seen on the current dungeon level is kept. As you explore more of the level, it appears on the screen in front of you. Rogue differs from most computer fantasy games in that it is screen oriented. Commands are all one or two keystrokes as opposed to pseudo-English sentences, and the results of your commands are displayed graphically on the screen rather than being explained in words. Another major difference between rogue and other computer fantasy games is that once you have solved all the puzzles in a standard fantasy game, it has lost most of its excitement and it ceases to be fun. Rogue, on the other hand, generates a new dungeon every time you play it and even the author finds it an entertaining and exciting game. In order to understand what is going on in rogue you have to first get some grasp of what rogue is doing with the screen. The rogue screen is intended to replace the You can see ... descriptions of standard fantasy games. Figure 1 is a sample of what a rogue screen might look like.

```
Level: 1  Gold: 0  Hp: 12(12)  Str: 16(16)  Ac: 6  Exp: 1/0
```

Figure 1
At the bottom line of the screen are a few pieces of cryptic information describing your current status. Here is an explanation of what these things mean:

This number indicates how deep you have gone in the dungeon. It starts at one and goes up as you go deeper into the dungeon. The number of gold pieces you have managed to find and keep with you so far.

Your current and maximum hit points. Hit points indicate how much damage you can take before you die.

The more you get hit in a fight, the lower they get. You can regain hit points by resting. The number in parentheses is the maximum number your hit points can reach.

Your current strength and maximum ever strength. The higher the number, the stronger you are. The number in the parentheses is the maximum strength you have attained so far this game.

Your current armor class. This number indicates how effective your armor is in stopping blows from unfriendly creatures. The lower this number is, the more effective the armor.

These two numbers give your current experience level and experience points. As you do things, you gain experience points. At certain experience point totals, you gain an experience level. The more experienced you are, the better you are able to fight and to withstand magical attacks.

The top line of the screen is reserved for printing messages that describe things that are impossible to represent visually. If you see a "--More--" on the top line, this means that rogue wants to print another message on the screen, but it wants to make certain that you have read the one that is there first. To read the next message, just type a space.

The rest of the screen is the map of the level as you have explored it so far. Each symbol on the screen represents something. Here is a list of what the various symbols mean:

This symbol represents you, the adventurer. These symbols represent the walls of rooms. A door to/from a room.
The floor of a room.
The floor of a passage between rooms.
A pile or pot of gold.
A weapon of some sort.
A piece of armor.
A flask containing a magic potion.
A piece of paper, usually a magic scroll.
A ring with magic properties
A magical staff or wand
A trap, watch out for these.
A staircase to other levels
A piece of food.

The lowercase letters represent the various inhabitants of the Dungeons of Doom.
Watch out, they can be nasty and vicious.

Commands are given to rogue by typing one or two characters.
Most commands can be preceded by a count to repeat them (e.g. typing
10s
will do ten searches).
Commands for which counts make no sense
have the count ignored.
To cancel a count or a prefix,
type <ESCAPE>.

The list of commands is rather long,
but it can be read at any time during the game with the
? command.

Here it is for reference,
with a short explanation of each command.
The help command.
Asks for a character to give help on.
If you type a
*,
it will list all the commands,
otherwise it will explain what the character you typed does.
This is the What is that on the screen? command.
A
/ followed by any character that you see on the level,
will tell you what that character is.
For instance,
typing
/@
will tell you that the
@ symbol represents you, the player.
Move left.
You move one space to the left.
If you use upper case
h,
you will continue to move left until you run into something. This works for all movement commands

(e.g.

L
means run in direction

L)
Move down.
Move up.
Move right.
Move diagonally up and left.
Move diagonally up and right.
Move diagonally down and left.
Move diagonally down and right.
Throw an object.
This is a prefix command.
When followed with a direction
it throws an object in the specified direction.
(e.g. type

th

to throw
something to the left.)
Find prefix.
When followed by a direction
it means to continue moving in the specified direction until you pass something interesting or run into a wall.
You should experiment with this,
since it is a very useful command,
but very difficult to describe.
Zap prefix.
Point a staff or wand in a given direction and fire it.
Even non-directional staves must be pointed in some direction to be used.
Identify trap command.
If a trap is on your map and you can't remember what type it is,
you can get rogue to remind you by getting next to it and typing

followed by the direction that would move you on top of it.
Search for traps and secret doors.
Examine each space immediately adjacent to you for the existence of a trap or secret door.
There is a large chance that even if there is something there, you won't find it,
so you might have to search a while before you find something.
Climb down a staircase to the next level.
Not surprisingly, this can only be done if you are standing on staircase.
Climb up a staircase to the level above.
This can't be done without the Amulet of Yendor in your posession.
Rest.
This is the do nothing command.
This is good for waiting and healing.
Inventory.
List what you are carrying in your pack.
Selective inventory.
Tells you what a single item in your pack is.
Quaff one of the potions you are carrying.
Read one of the scrolls in your pack.
Eat food from your pack.
Wield a weapon.
Take a weapon out of your pack and carry it for use in combat,
replacing the one you are currently using (if any).
Wear armor.
You can only wear one suit of armor at a time.
This takes extra time.
Take armor off.
You can’t remove armor that is cursed.
This takes extra time.
Put on a ring.
You can wear only two rings at a time
(one on each hand).
If you aren't wearing any rings,
this command will ask you which hand you want to wear it on,
otherwise, it will place it on the unused hand.
The program assumes that you wield your sword in your right hand.
Remove a ring.
If you are only wearing one ring,
this command takes it off.
If you are wearing two,
it will ask you which one you wish to remove,
Drop an object.
Take something out of your pack and leave it lying on the floor.
Only one object can occupy each space.
You cannot drop a cursed object at all
if you are wielding or wearing it.
Call an object something.
If you have a type of object in your pack
which you wish to remember something about,
you can use the call command to give a name to that type of object.
This is usually used when you figure out what a
potion, scroll, ring, or staff is
after you pick it up.
(See the
askme
option below.)
Print out which things you’ve discovered something about.
This command will ask you what type of thing you are interested in.
If you type the character for a given type of object
(e.g.
!}
for potion) it will tell you which kinds of that type of object you’ve discovered (i.e., figured out what they are).

This command works for potions, scrolls, rings, staves, and wands.

Examine and set options.

This command is further explained in the section on options.

Redraws the screen.

Useful if spurious messages or transmission errors have messed up the display.

Repeat last message.

Useful when a message disappears before you can read it.

This only repeats the last message that was not a mistyped command so that you don’t loose anything by accidentally typing the wrong character instead of \"R.\"

Cancel a command, prefix, or count.

Escape to a shell for some commands.

Quit.

Leave the game.

Save the current game in a file.

It will ask you whether you wish to use the default save file. Rogue won’t let you start up a copy of a saved game, and it removes the save file as soon as you start up a restored game. This is to prevent people from saving a game just before a dangerous position and then restarting it if they die.

To restore a saved game, give the file name as an argument to rogue. As in

% rogue save_file

To restart from the default save file (see below), run

% rogue -r

Prints the program version number.

Rooms in the dungeons are either lit or dark.

If you walk into a lit room, the entire room will be drawn on the screen as soon as you enter.

If you walk into a dark room, it will only be displayed as you explore it.

Upon leaving a room, all objects inside the room which might move or be removed are erased from the screen.

In the darkness you can only see one space in all directions around you.

A corridor is always dark.

If you see a monster and you wish to fight it, just attempt to run into it.
Many times a monster you find will mind its own business unless you attack it.

It is often the case that discretion is the better part of valor.

When you find something in the dungeon, it is common to want to pick the object up.

This is accomplished in rogue by walking over the object.

If you are carrying too many things, the program will tell you and it won't pick up the object, otherwise it will add it to your pack and tell you what you just picked up.

Many of the commands that operate on objects must prompt you to find out which object you want to use.

If you change your mind and don't want to do that command after all, just type an <ESCAPE> and the command will be aborted.

Some objects, like armor and weapons, are easily differentiated.

Others, like scrolls and potions, are given labels which vary according to type.

During a game, any two of the same kind of object with the same label are the same type.

However, the labels will vary from game to game.

When you use one of these labeled objects, if its effect is obvious, rogue will remember what it is for you.

If it's effect isn't extremely obvious, you can use the call command (see above) or the askme option (see below) to scribble down something about it so you will recognize it later.

Some weapons, like arrows, come in bunches, but most come one at a time.

In order to use a weapon, you must wield it.

To fire an arrow out of a bow, you must first wield the bow, then throw the arrow.

You can only wield one weapon at a time, but you can't change weapons if the one you are currently wielding is cursed.

There are various sorts of armor lying around in the dungeon. Some of it is enchanted,
some is cursed,
and some is just normal.
Different armor types have different armor classes.
The lower the armor class,
the more protection the armor affords against the blows of monsters.
Here is a list of the various armor types and their normal armor class:

<table>
<thead>
<tr>
<th>Type</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>10</td>
</tr>
<tr>
<td>Leather armor</td>
<td>8</td>
</tr>
<tr>
<td>Studded leather / Ring mail</td>
<td>7</td>
</tr>
<tr>
<td>Scale mail</td>
<td>6</td>
</tr>
<tr>
<td>Chain mail</td>
<td>5</td>
</tr>
<tr>
<td>Banded mail / Splint mail</td>
<td>4</td>
</tr>
<tr>
<td>Plate mail</td>
<td>3</td>
</tr>
</tbody>
</table>

If a piece of armor is enchanted,
its armor class will be lower than normal.
If a suit of armor is cursed,
it will be higher,
and you will not be able to remove it.
However, not all armor with a class that is higher than normal
is cursed.

Scrolls come with titles in an unknown tongue.
After you read a scroll,
it disappears from your pack.
Potions are labeled by the color of the liquid inside the flask.
They disappear after being quaffed.
Staves and wands do the same kinds of things.
Staves are identified by a type of wood;
wands by a type of metal or bone.
They are generally things you want to do to something
over a long distance,
so you must point them at what you wish to affect
to use them.
Some staves are not affected by the direction
they are pointed, though.
Staves come with multiple magic charges,
the number being random,
and when they are used up,
the staff is just a piece of wood or metal.
Rings are very useful items,
since they are relatively permanent magic,
unlike the usually fleeting effects of potions, scrolls, and staves.
Of course,
the bad rings are also more powerful.
Most rings also cause you to use up food more rapidly,
the rate varying with the type of ring.
Rings are differentiated by their stone settings.
Due to variations in personal tastes
and conceptions of the way rogue should do things,
there are a set of options you can set
that cause rogue to behave in various different ways.

There are two ways to set the options.
The first is with the
  o
command of rogue;
the second is with the
ROGUEOPTS
environment variable.
On Version 6 systems,
there is no equivalent of the ROGUEOPTS feature.
When you type
  o
in rogue,
it clears the screen
and displays the current settings for all the options.
It then places the cursor by the value of the first option
and waits for you to type.
You can type a <RETURN>
which means to go to the next option,
a
which means to go to the previous option,
an <ESCAPE>
which means to return to the game,
or you can give the option a value.
For boolean options this merely involves typing
  t
for true or
  f
for false.
For string options,
type the new value followed by a <RETURN>.
The ROGUEOPTS variable is a string
containing a comma separated list of initial values
for the various options.
Boolean variables can be turned on by listing their name
or turned off by putting a
  no
in front of the name.
Thus to set up an environment variable so that
is on,
is off,
and the
is set to Blue Meanie,
use the command:

% setenv ROGUEOPTS "jump,noterse,name=Blue Meanie"

For those of you who use the Bourne shell, the commands would be
$ ROGUEOPTS="jump, noterse, name=Blue Meanie"
   export ROGUEOPTS

Here is a list of the options
and an explanation of what each one is for.
The default value for each is enclosed in square brackets.
For character string options,
input over fifty characters will be ignored.
Useful for those who are tired of the sometimes
lengthy messages of rogue.
This is a useful option for playing on Snow terminals,
so this option defaults to
if your
are on a slow (1200 baud or under) terminal.
   If this option is set,
running moves will not be displayed
until you reach the end of the move.
This saves considerable cpu and display time.
   This option defaults to
if you are using a slow terminal.
All typeahead is thrown away after each round of battle.
   This is useful for those who type far ahead
and then watch in dismay as a Kobold kills them.
Upon reading a scroll or quaffing a potion
which does not automatically identify itself upon use,
rogue will ask you what to name it
so you can recognize it if you encounter it again.
   Follow turnings in passageways.
      If you run in a passage
and you run into stone or a wall,
rogue will see if it can turn to the right or left.
      If it can only turn one way,
         it will turn that way.
      If it can turn either or neither,
         it will stop.
   This is followed strictly,
which can sometimes lead to slightly confusing occurrences
(which is why it defaults to being off).
   The
   f
prefix still works.
   Inventory type.
This can have one of three values:
or
   With
the top lines of the map are overwritten
with the list
when inventory is requested
or when
Which item do you wish to ...? questions
are answered with a
* However, if the list is longer than a screenful, the screen is cleared. With lists are displayed one item at a time on the top of the screen, and with the screen is cleared, the list is displayed, and then the dungeon level is re-displayed. Due to speed considerations, is the default for terminals without clear-to-end-of-line capabilities. This is the name of your character. It is used if you get on the top ten scorers' list. This should hold the name of a fruit that you enjoy eating. It is basically a whimsey that the program uses in a couple of places. The default file name for saving the game. If your phone is hung up by accident, rogue will automatically save the game in this file. The file name may contain the special character which expands to be your home directory. Rogue usually maintains a list of the top ten scoring people on your machine. Each account on the machine can post only one non-winning score on this list. If you score higher than someone else on this list, or better your previous score on the list, you will be inserted in the proper place under your current name. If you quit the game, you get out with all of your gold intact. If, however, you get killed in the Dungeons of Doom, your body is forwarded to your next-of-kin, along with 90% of your gold; ten percent of your gold is kept by the Dungeons' wizard as a fee. This should make you consider whether you want to take one last hit at that monster and possibly live, or quit and thus stop with whatever you have. If you quit, you do get all your gold, but if you swing and live, you might find more. If you just want to see what the current top ten list is, you can type

% rogue -s

Rogue was originally conceived of by Glenn Wichman and Michael Toy. Ken Arnold and Michael Toy then smoothed out the user interface, and added jillions of new features. We would like to thank Bob Arnold, Michelle Busch,
Andy Hatcher,  
Kipp Hickman,  
Mark Horton,  
Daniel Jensen,  
Bill Joy,  
Joe Kalash,  
Steve Maurer,  
Marty McNary,  
Jan Miller,  
and  
Scott Nelson  
for their ideas and assistance,  
and also the teeming multitudes  
who graciously ignored work, school, and social life to play rogue  
and send us bugs, complaints, suggestions, and just plain flames.  
And also Mom.  
! Funky! Stuff!  
echo All done.  
exit
NAME
sky - obtain ephemerides

SYNOPSIS
/usr/games/sky [ -1 ]

DESCRIPTION
Not on 5000/20/40/50.

Sky predicts the apparent locations of the Sun, the Moon, the planets out to Saturn, stars of magnitude at least 2.5, and certain other celestial objects. Sky reads the standard input to obtain a GMT time typed on one line with blanks separating year, month number, day, hour, and minute; if the year is missing the current year is used. If a blank line is typed, the current time is used. The program prints the azimuth, elevation, and magnitude of objects which are above the horizon at the ephemeris location of Murray Hill at the indicated time. The -1 option causes it to ask for another location.

Placing a "1" input after the minute entry causes the program to print out the Greenwich Sidereal Time at the indicated moment and to print for each body its topographic right ascension and declination as well as its azimuth and elevation. Also, instead of the magnitude, the semidiameter of the body, in seconds of arc, is reported.

A "2" after the minute entry makes the coordinate system geocentric.

The effects of atmospheric extinction on magnitudes are not included; the brightest magnitudes of variable stars are marked with $^\circ$.

For all bodies, the program takes into account precession and nutation of the equinox, annual (but not diurnal) aberration, diurnal parallax, and the proper motion of stars. In no case is refraction included.

The program takes into account perturbations of the Earth due to the Moon, Venus, Mars, and Jupiter. The expected accuracies are: for the Sun and other stellar bodies a few tenths of seconds of arc; for the Moon (on which particular care is lavished) likewise a few tenths of seconds. For the Sun, Moon and stars the accuracy is sufficient to predict the circumstances of eclipses and occultations to within a few seconds of time. The planets may be off by several minutes of arc.

There are lots of special options not described here, which do things like substituting named star catalogs, smoothing nutation and aberration to aid generation of mean places of stars, and making conventional adjustments to the Moon to improve eclipse predictions.

For the most accurate use of the program it is necessary to know that it actually runs in Ephemeris time.
SEE ALSO

American Ephemeris and Nautical Almanac, for the appropriate years; also, the Explanatory Supplement to the American Ephemeris and Nautical Almanac.
NAME
ttt, cubic - tic-tac-toe

SYNOPSIS
/usr/games/ttt
/usr/games/cubic

DESCRIPTION
Ttt is the X and O game popular in the first grade. This is a learning program that never makes the same mistake twice.

Although it learns, it learns slowly. It must lose nearly 80 games to completely know the game.

Cubic plays three-dimensional tic-tac-toe on a 4 by 4 by 4 board. Moves are specified as a sequence of three coordinate numbers in the range 1-4.

FILES
/usr/games/ttt.k learning file

RESTRICTIONS
/usr/games/cubic not available on 5000/20/40/50.
NAME
   wump - the game of hunt-the-wumpus
SYNOPSIS
   /usr/games/wump
DESCRIPTION
   Wump plays the game of Hunt the Wumpus.
   A Wumpus is a creature that lives in a cave with several rooms connected by tunnels. You wander among the rooms, trying to shoot the Wumpus with an arrow, meanwhile avoiding being eaten by the Wumpus and falling into Bottomless Pits. There are also Super Bats which are likely to pick you up and drop you in some random room.
   The program asks various questions which you answer one per line; it gives a more detailed description if you want.
[This page left blank.]