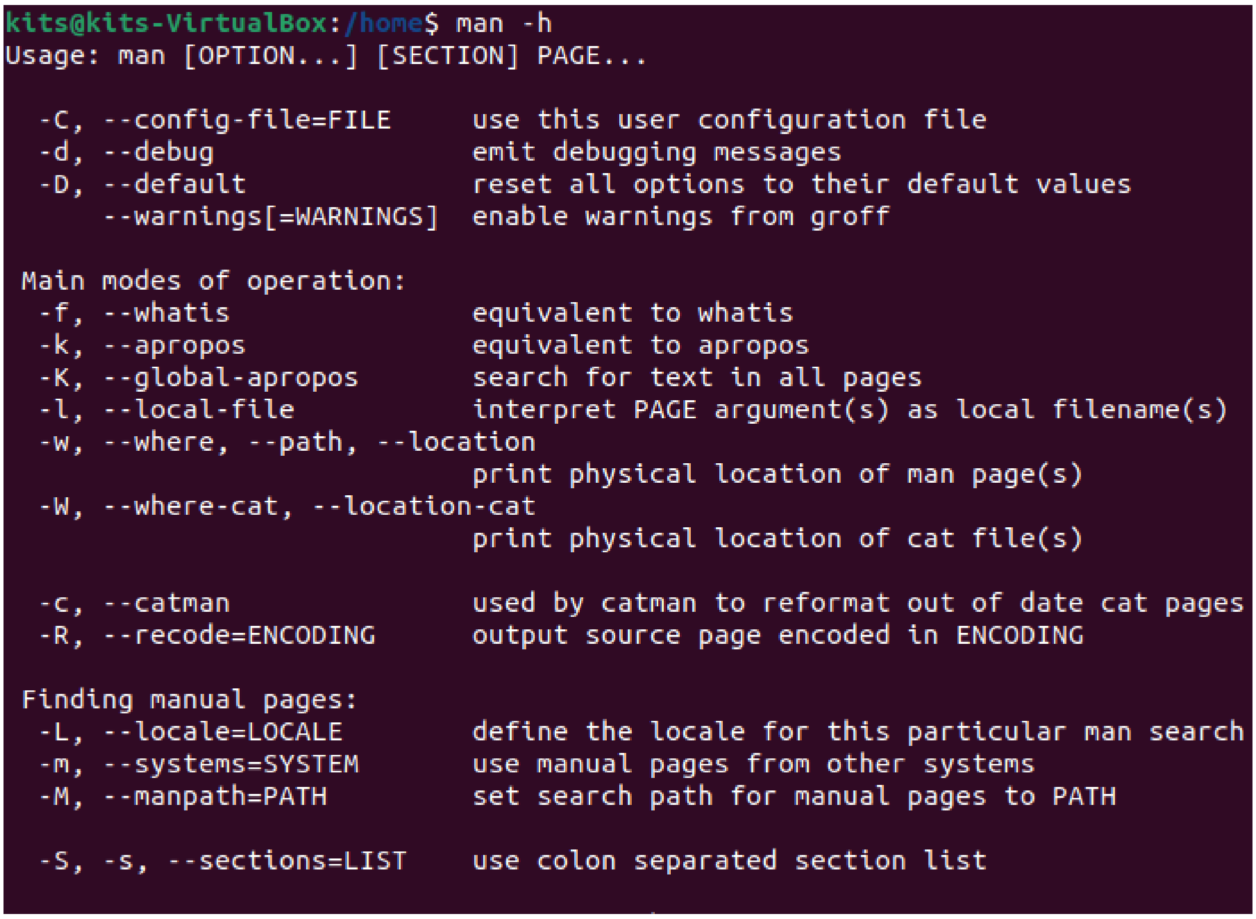
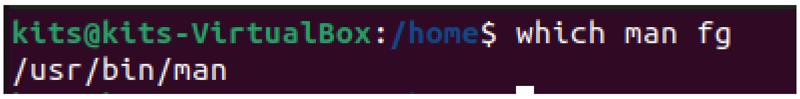
1.



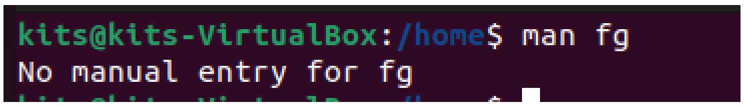
```
kits@kits-VirtualBox:/home$ man -h
Usage: man [OPTION...] [SECTION] PAGE...

  -C, --config-file=FILE     use this user configuration file
  -d, --debug                emit debugging messages
  -D, --default              reset all options to their default values
      --warnings[=WARNINGS]  enable warnings from groff

 Main modes of operation:
  -f, --whatis               equivalent to whatis
  -k, --apropos              equivalent to apropos
  -K, --global-apropos       search for text in all pages
  -l, --local-file           interpret PAGE argument(s) as local filename(s)
  -w, --where, --path, --location
                             print physical location of man page(s)
  -W, --where-cat, --location-cat
                             print physical location of cat file(s)

  -c, --catman               used by catman to reformat out of date cat pages
  -R, --recode=ENCODING      output source page encoded in ENCODING

 Finding manual pages:
  -L, --locale=LOCALE        define the locale for this particular man search
  -m, --systems=SYSTEM       use manual pages from other systems
  -M, --manpath=PATH         set search path for manual pages to PATH

  -S, -s, --sections=LIST    use colon separated section list
```

2.



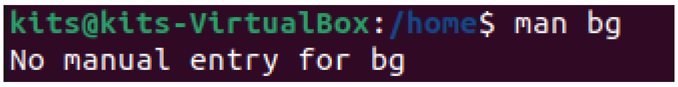
```
kits@kits-VirtualBox:/home$ which man fg
/usr/bin/man
```

3.



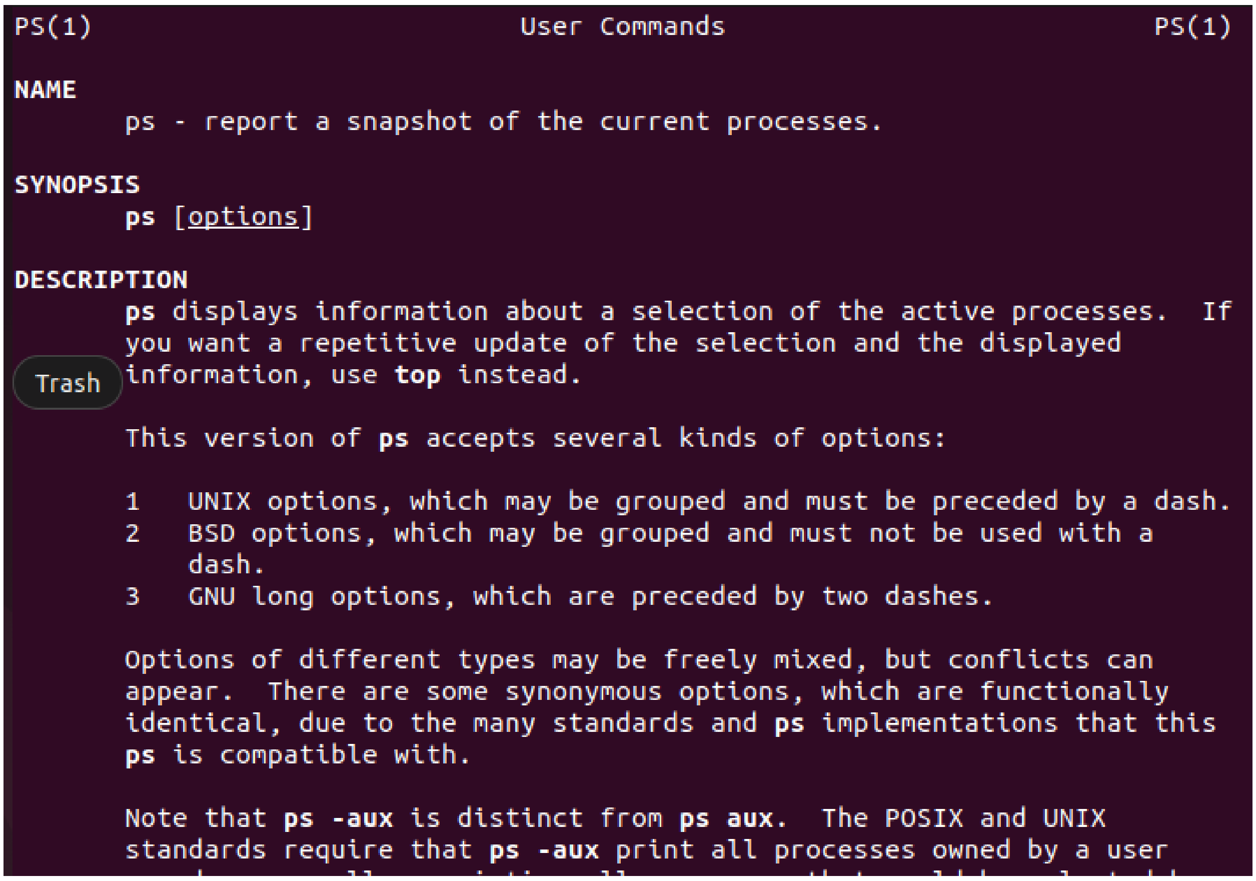
```
kits@kits-VirtualBox:/home$ man fg
No manual entry for fg
```

4.



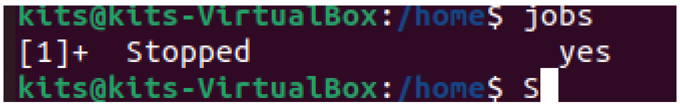
```
kits@kits-VirtualBox:/home$ man bg
No manual entry for bg
```

5.



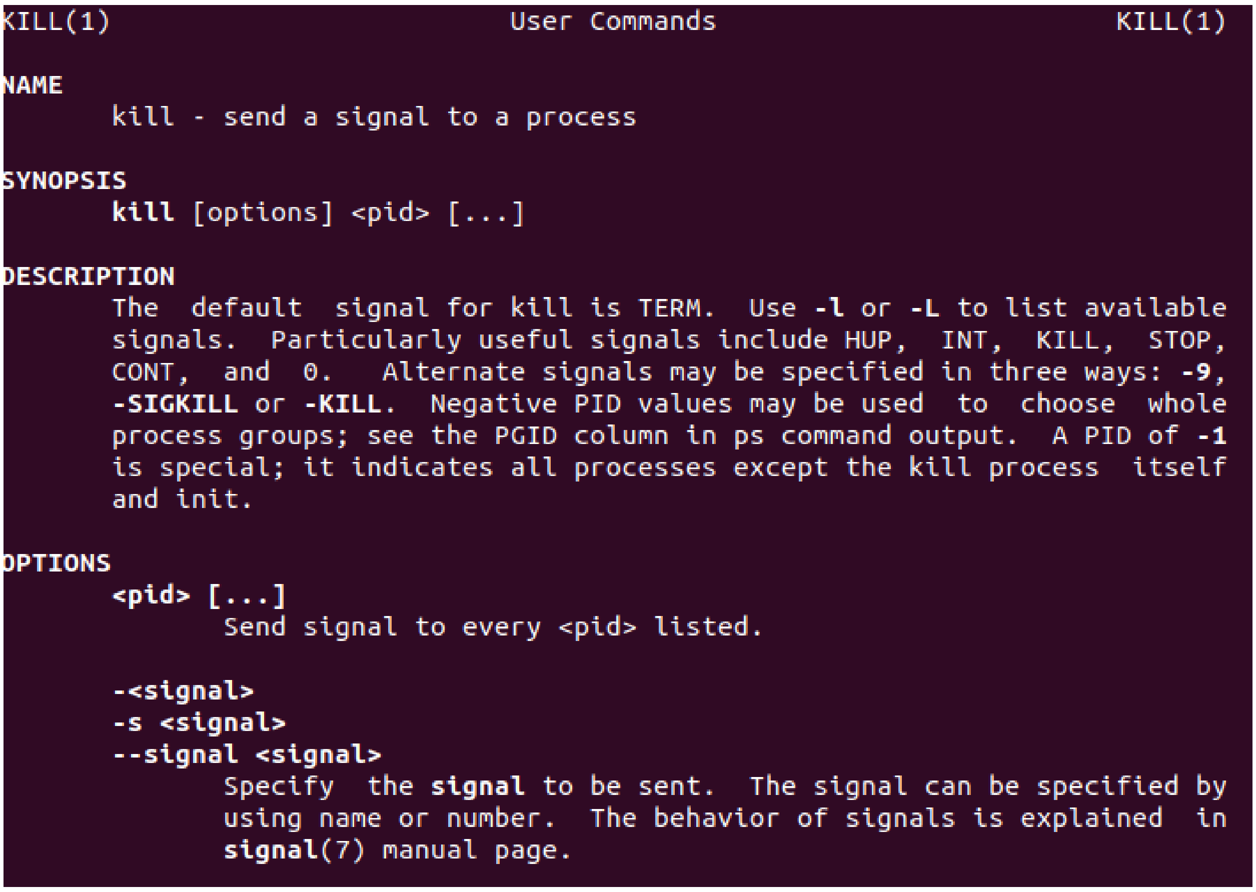
```
PS(1)                          User Commands                          PS(1)

NAME
       ps - report a snapshot of the current processes.

SYNOPSIS
       ps [options]

DESCRIPTION
       ps displays information about a selection of the active processes.  If
       you want a repetitive update of the selection and the displayed
       information, use top instead.

       This version of ps accepts several kinds of options:

       1   UNIX options, which may be grouped and must be preceded by a dash.
       2   BSD options, which may be grouped and must not be used with a
           dash.
       3   GNU long options, which are preceded by two dashes.

       Options of different types may be freely mixed, but conflicts can
       appear.  There are some synonymous options, which are functionally
       identical, due to the many standards and ps implementations that this
       ps is compatible with.

       Note that ps -aux is distinct from ps aux.  The POSIX and UNIX
       standards require that ps -aux print all processes owned by a user
```

6.



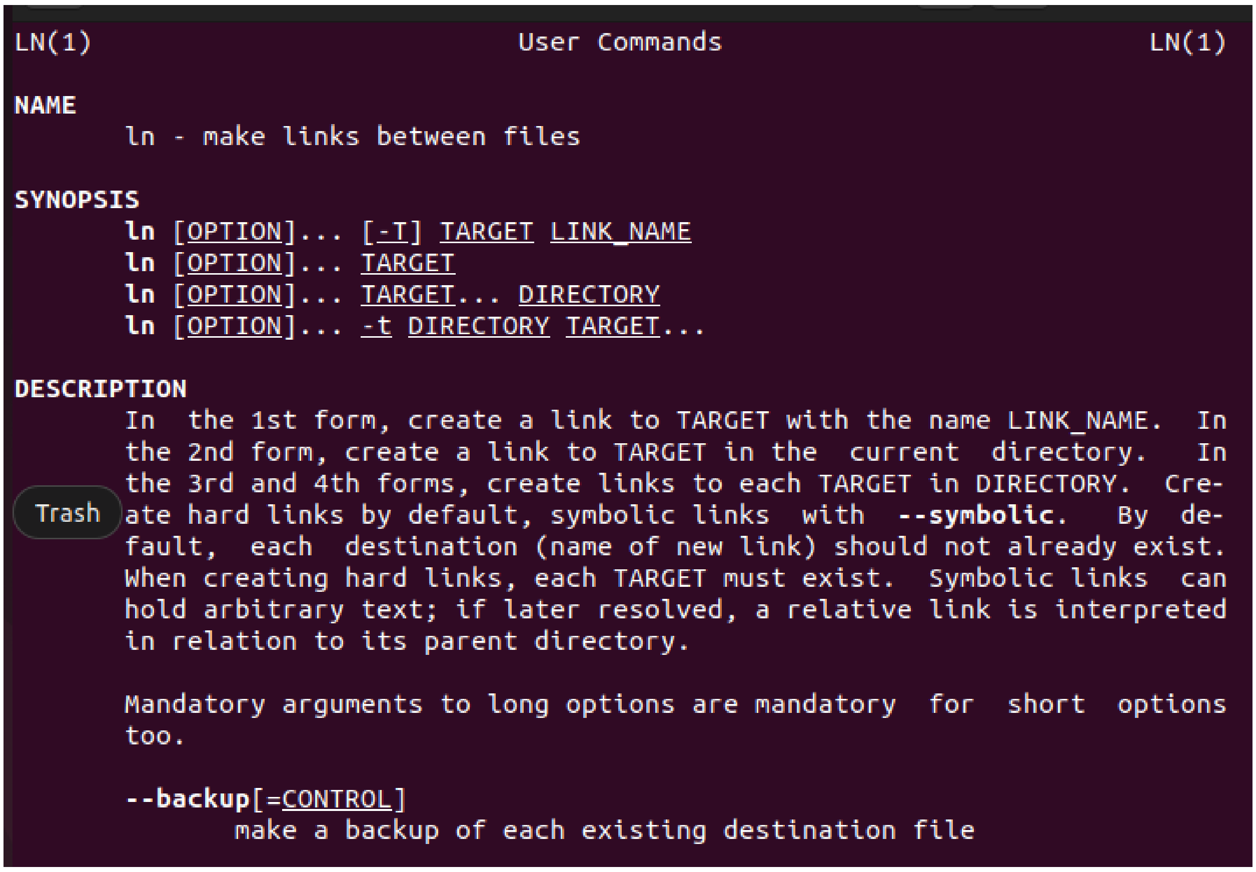
```
kits@kits-VirtualBox:/home$ jobs
[1]+  Stopped                 yes
kits@kits-VirtualBox:/home$ S
```

7.



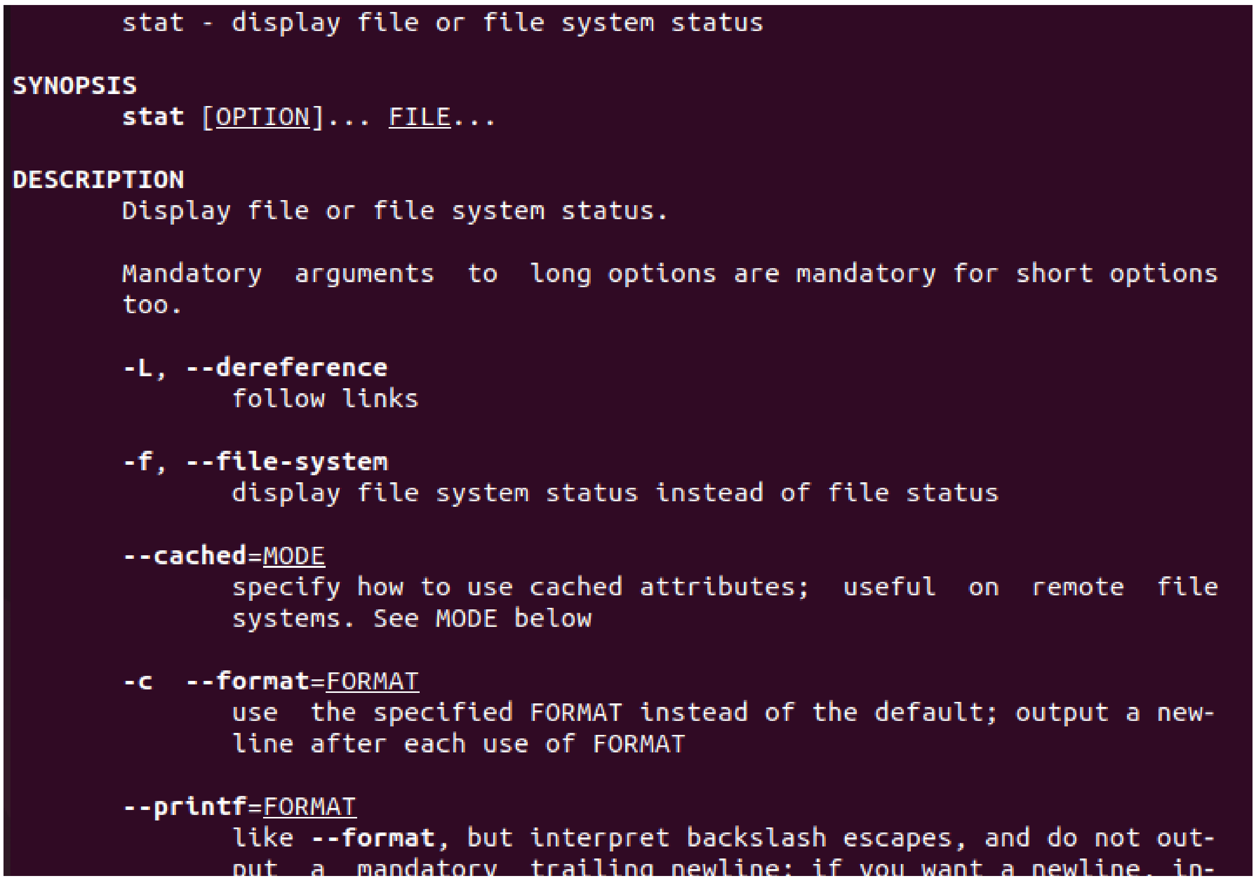
```
KILL(1)                       User Commands                      KILL(1)

NAME
       kill - send a signal to a process

SYNOPSIS
       kill [options] <pid> [...]

DESCRIPTION
       The  default  signal for kill is TERM.  Use -l or -L to list available
       signals.  Particularly useful signals include HUP,  INT,  KILL,  STOP,
       CONT,  and  0.   Alternate signals may be specified in three ways: -9,
       -SIGKILL or -KILL.  Negative PID values may be used  to  choose  whole
       process groups; see the PGID column in ps command output.  A PID of -1
       is special; it indicates all processes except the kill process  itself
       and init.

OPTIONS
       <pid> [...]
              Send signal to every <pid> listed.

       -<signal>
       -s <signal>
       --signal <signal>
              Specify  the signal to be sent.  The signal can be specified by
              using name or number.  The behavior of signals is explained  in
              signal(7) manual page.
```

8.

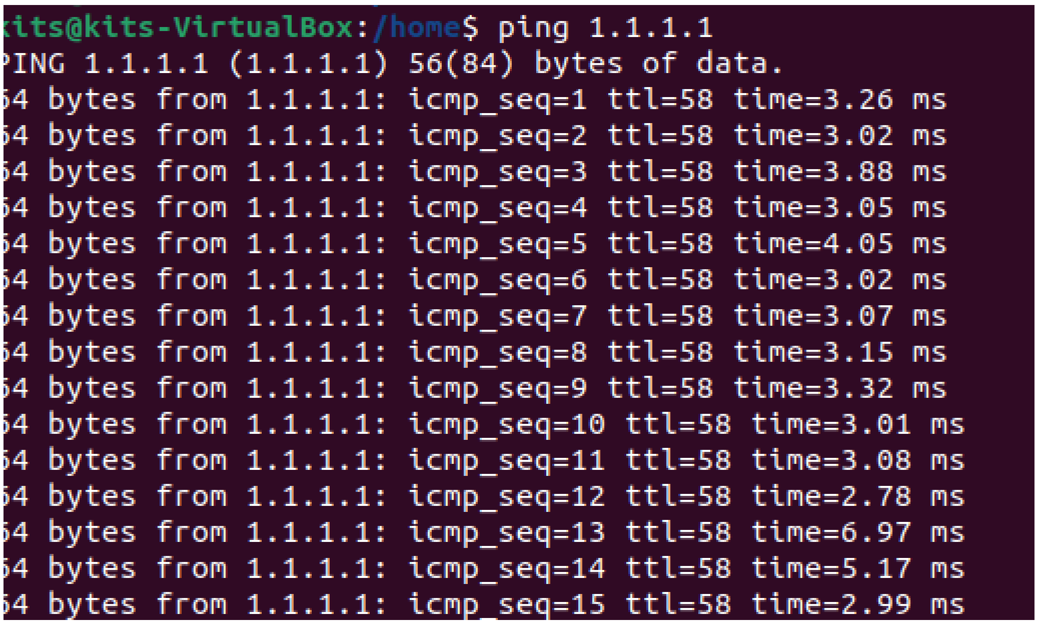


```
LN(1)                         User Commands                        LN(1)

NAME
       ln - make links between files

SYNOPSIS
       ln [OPTION]... [-T] TARGET LINK_NAME
       ln [OPTION]... TARGET
       ln [OPTION]... TARGET... DIRECTORY
       ln [OPTION]... -t DIRECTORY TARGET...

DESCRIPTION
       In  the 1st form, create a link to TARGET with the name LINK_NAME.  In
       the 2nd form, create a link to TARGET in the  current  directory.   In
       the 3rd and 4th forms, create links to each TARGET in DIRECTORY.  Cre-
       ate hard links by default, symbolic links  with  --symbolic.   By  de-
       fault,  each  destination (name of new link) should not already exist.
       When creating hard links, each TARGET must exist.  Symbolic links  can
       hold arbitrary text; if later resolved, a relative link is interpreted
       in relation to its parent directory.

       Mandatory arguments to long options are mandatory  for  short  options
       too.

       --backup[=CONTROL]
              make a backup of each existing destination file
```

9.

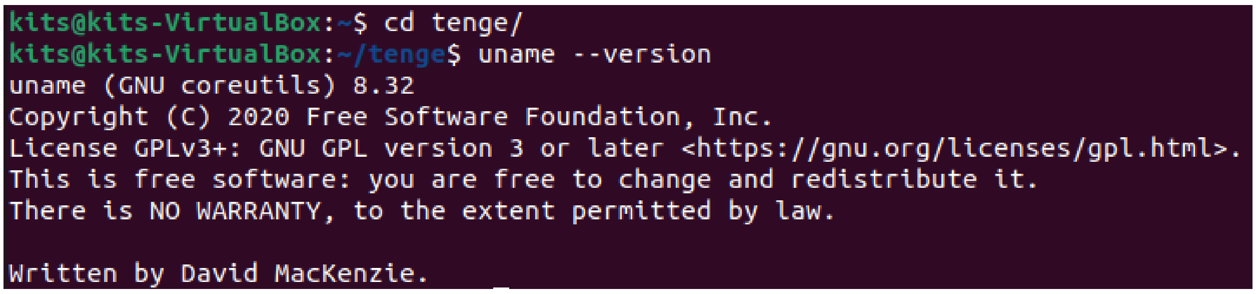


```
       stat - display file or file system status

SYNOPSIS
       stat [OPTION]... FILE...

DESCRIPTION
       Display file or file system status.

       Mandatory  arguments  to  long options are mandatory for short options
       too.

       -L, --dereference
              follow links

       -f, --file-system
              display file system status instead of file status

       --cached=MODE
              specify how to use cached attributes;  useful  on  remote  file
              systems. See MODE below

       -c  --format=FORMAT
              use  the specified FORMAT instead of the default; output a new‐
              line after each use of FORMAT

       --printf=FORMAT
              like --format, but interpret backslash escapes, and do not out‐
              put  a  mandatory  trailing newline; if you want a newline, in‐
```

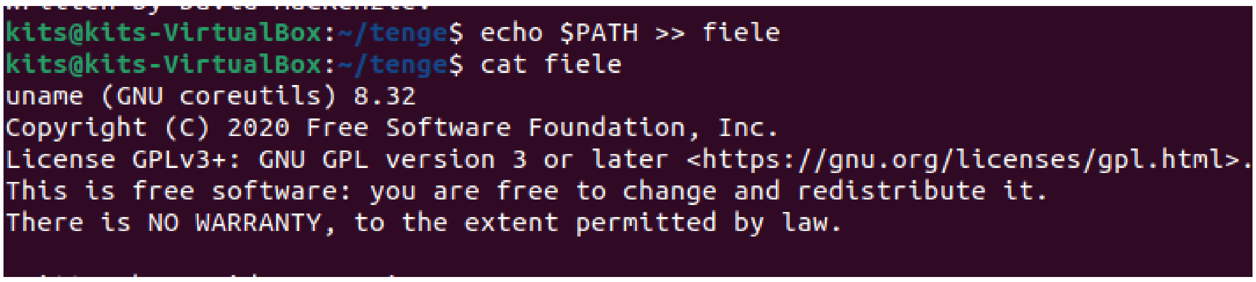


```
its@kits-VirtualBox:/home$ ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
54 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=3.26 ms
54 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=3.02 ms
54 bytes from 1.1.1.1: icmp_seq=3 ttl=58 time=3.88 ms
54 bytes from 1.1.1.1: icmp_seq=4 ttl=58 time=3.05 ms
54 bytes from 1.1.1.1: icmp_seq=5 ttl=58 time=4.05 ms
54 bytes from 1.1.1.1: icmp_seq=6 ttl=58 time=3.02 ms
54 bytes from 1.1.1.1: icmp_seq=7 ttl=58 time=3.07 ms
54 bytes from 1.1.1.1: icmp_seq=8 ttl=58 time=3.15 ms
54 bytes from 1.1.1.1: icmp_seq=9 ttl=58 time=3.32 ms
54 bytes from 1.1.1.1: icmp_seq=10 ttl=58 time=3.01 ms
54 bytes from 1.1.1.1: icmp_seq=11 ttl=58 time=3.08 ms
54 bytes from 1.1.1.1: icmp_seq=12 ttl=58 time=2.78 ms
54 bytes from 1.1.1.1: icmp_seq=13 ttl=58 time=6.97 ms
54 bytes from 1.1.1.1: icmp_seq=14 ttl=58 time=5.17 ms
54 bytes from 1.1.1.1: icmp_seq=15 ttl=58 time=2.99 ms
```

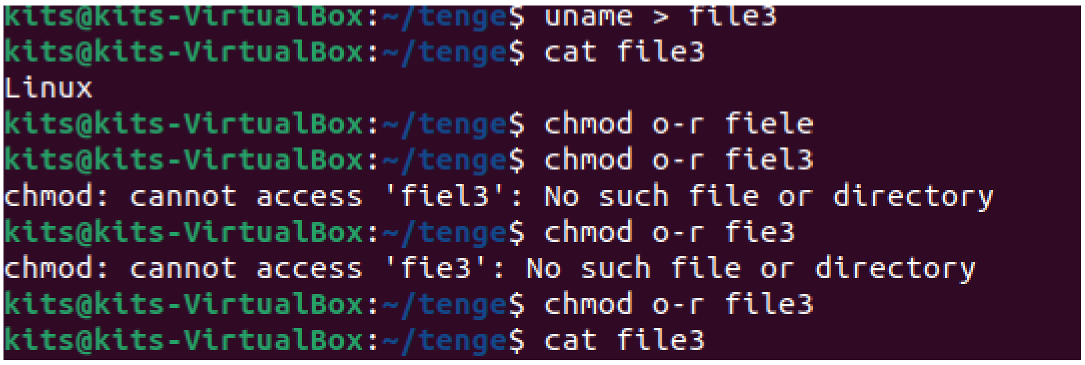
10.



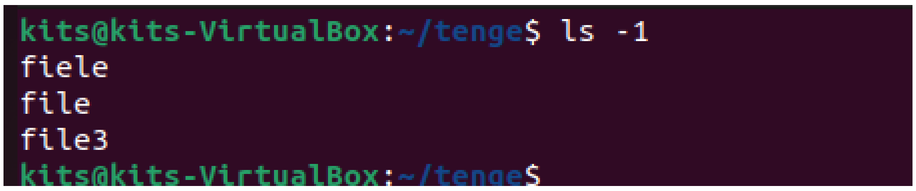
```
kits@kits-VirtualBox:~$ cd tenge/
kits@kits-VirtualBox:~/tenge$ uname --version
uname (GNU coreutils) 8.32
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by David MacKenzie.
```



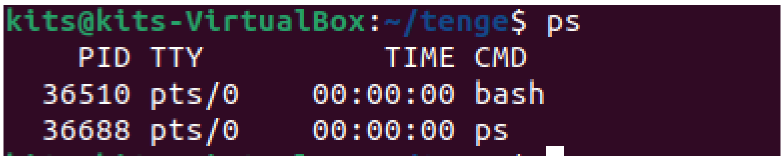
```
kits@kits-VirtualBox:~/tenge$ echo $PATH >> fiele
kits@kits-VirtualBox:~/tenge$ cat fiele
uname (GNU coreutils) 8.32
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```



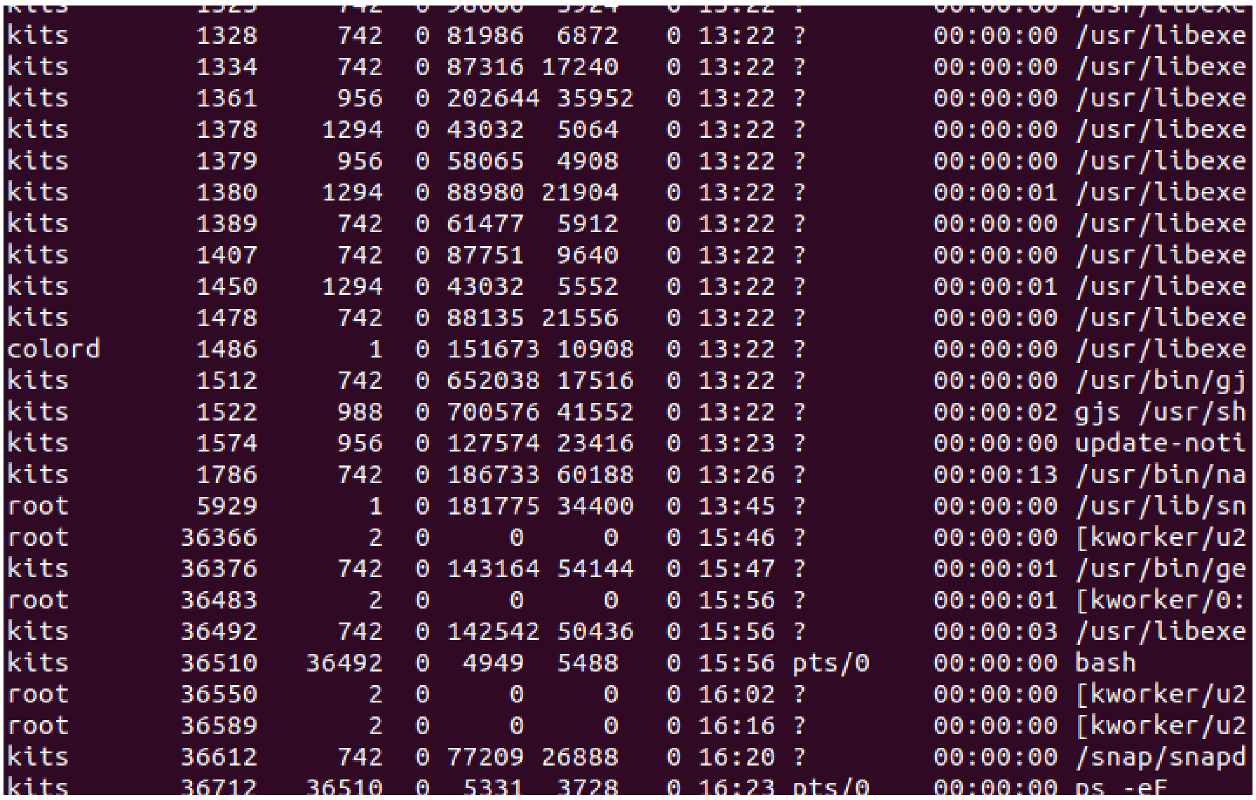
```
kits@kits-VirtualBox:~/tenge$ uname > file3
kits@kits-VirtualBox:~/tenge$ cat file3
Linux
kits@kits-VirtualBox:~/tenge$ chmod o-r fiele
kits@kits-VirtualBox:~/tenge$ chmod o-r fiel3
chmod: cannot access 'fiel3': No such file or directory
kits@kits-VirtualBox:~/tenge$ chmod o-r fie3
chmod: cannot access 'fie3': No such file or directory
kits@kits-VirtualBox:~/tenge$ chmod o-r file3
kits@kits-VirtualBox:~/tenge$ cat file3
```



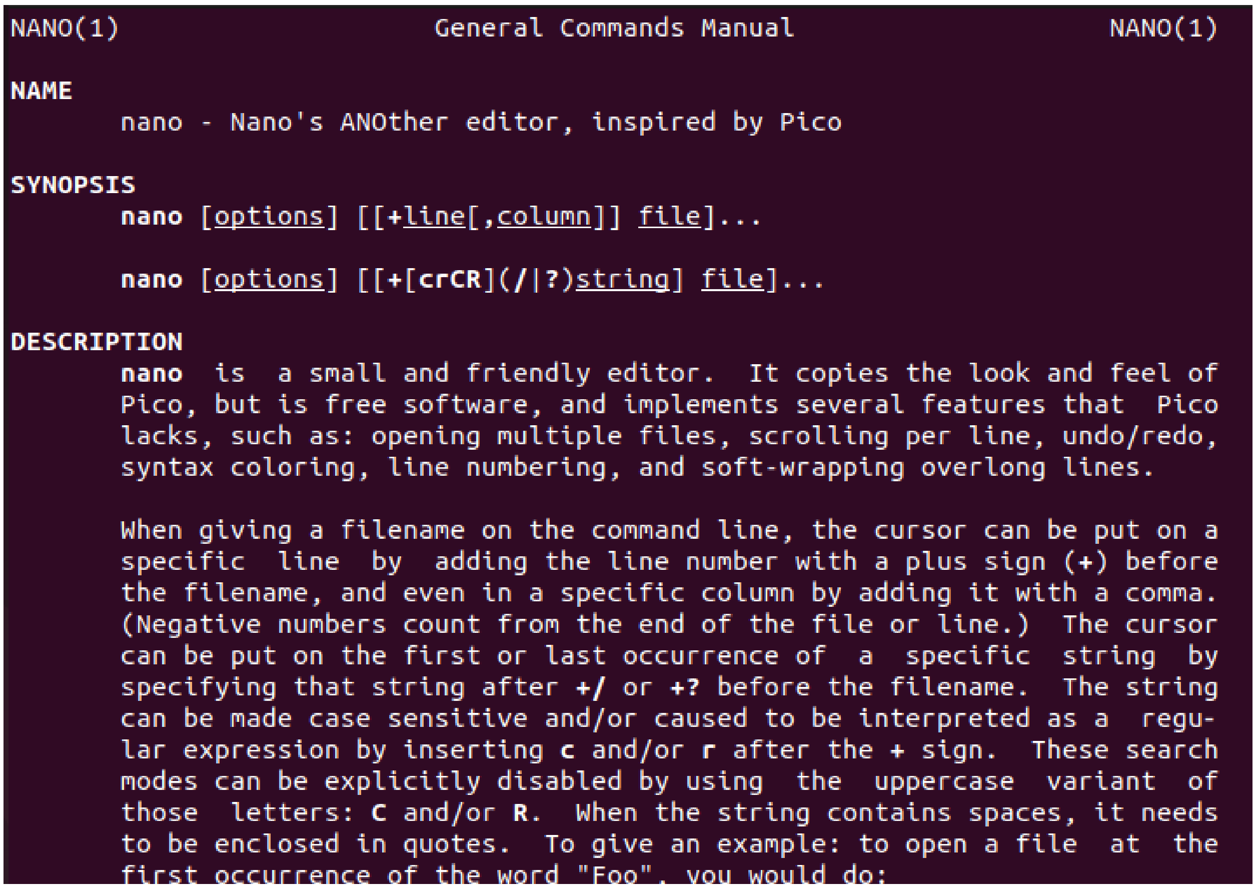
```
kits@kits-VirtualBox:~/tenge$ ls -1
fiele
file
file3
kits@kits-VirtualBox:~/tenge$
```



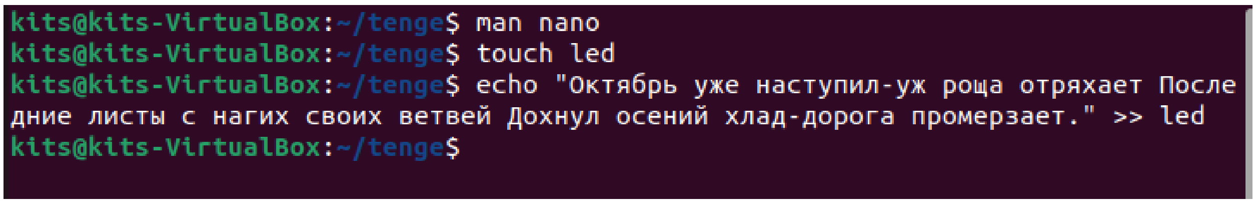
```
kits@kits-VirtualBox:~/tenge$ ps
    PID TTY          TIME CMD
  36510 pts/0    00:00:00 bash
  36688 pts/0    00:00:00 ps
```



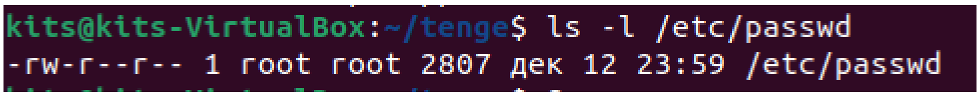
```
kits        1328     742  0 81986  6872   0 13:22 ?        00:00:00 /usr/libexe
kits        1334     742  0 87316 17240   0 13:22 ?        00:00:00 /usr/libexe
kits        1361     956  0 202644 35952  0 13:22 ?        00:00:00 /usr/libexe
kits        1378    1294  0 43032  5064   0 13:22 ?        00:00:00 /usr/libexe
kits        1379     956  0 58065  4908   0 13:22 ?        00:00:00 /usr/libexe
kits        1380    1294  0 88980 21904   0 13:22 ?        00:00:01 /usr/libexe
kits        1389     742  0 61477  5912   0 13:22 ?        00:00:00 /usr/libexe
kits        1407     742  0 87751  9640   0 13:22 ?        00:00:00 /usr/libexe
kits        1450    1294  0 43032  5552   0 13:22 ?        00:00:01 /usr/libexe
kits        1478     742  0 88135 21556   0 13:22 ?        00:00:00 /usr/libexe
colord      1486       1  0 151673 10908  0 13:22 ?        00:00:00 /usr/libexe
kits        1512     742  0 652038 17516  0 13:22 ?        00:00:00 /usr/bin/gj
kits        1522     988  0 700576 41552  0 13:22 ?        00:00:02 gjs /usr/sh
kits        1574     956  0 127574 23416  0 13:23 ?        00:00:00 update-noti
kits        1786     742  0 186733 60188  0 13:26 ?        00:00:13 /usr/bin/na
root        5929       1  0 181775 34400  0 13:45 ?        00:00:00 /usr/lib/sn
root       36366       2  0     0     0   0 15:46 ?        00:00:00 [kworker/u2
kits       36376     742  0 143164 54144  0 15:47 ?        00:00:01 /usr/bin/ge
root       36483       2  0     0     0   0 15:56 ?        00:00:01 [kworker/0:
kits       36492     742  0 142542 50436  0 15:56 ?        00:00:03 /usr/libexe
kits       36510   36492  0  4949  5488   0 15:56 pts/0    00:00:00 bash
root       36550       2  0     0     0   0 16:02 ?        00:00:00 [kworker/u2
root       36589       2  0     0     0   0 16:16 ?        00:00:00 [kworker/u2
kits       36612     742  0 77209 26888   0 16:20 ?        00:00:00 /snap/snapd
kits       36712   36510  0  5331  3728   0 16:23 pts/0    00:00:00 ps -eF
```



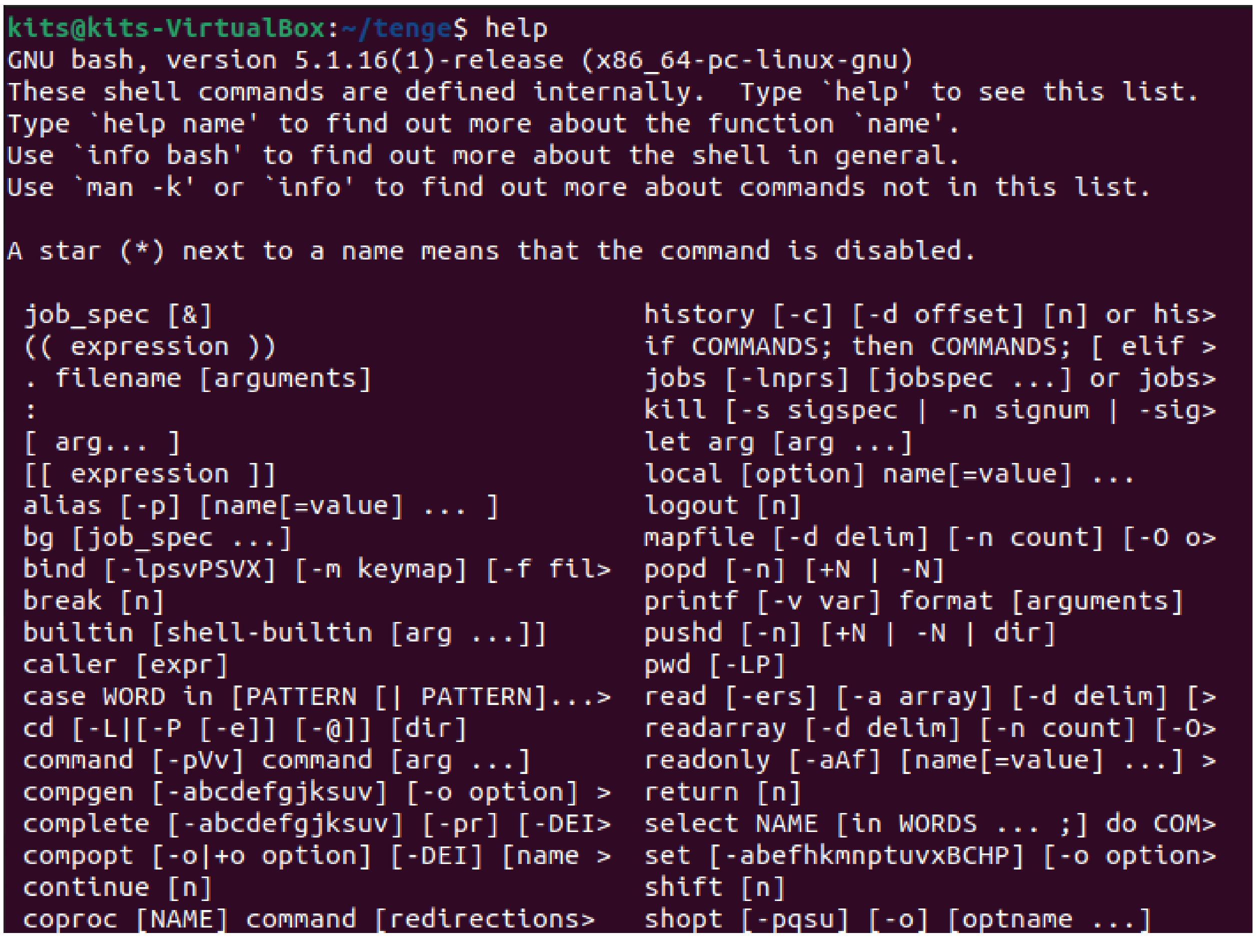
```
NANO(1)                      General Commands Manual                     NANO(1)

NAME
       nano - Nano's ANOther editor, inspired by Pico

SYNOPSIS
       nano [options] [[+line[,column]] file]...

       nano [options] [[+[crCR](/|?)string] file]...

DESCRIPTION
       nano  is  a small and friendly editor.  It copies the look and feel of
       Pico, but is free software, and implements several features that  Pico
       lacks, such as: opening multiple files, scrolling per line, undo/redo,
       syntax coloring, line numbering, and soft-wrapping overlong lines.

       When giving a filename on the command line, the cursor can be put on a
       specific  line  by  adding the line number with a plus sign (+) before
       the filename, and even in a specific column by adding it with a comma.
       (Negative numbers count from the end of the file or line.)  The cursor
       can be put on the first or last occurrence of  a  specific  string  by
       specifying that string after +/ or +? before the filename.  The string
       can be made case sensitive and/or caused to be interpreted as a  regu-
       lar expression by inserting c and/or r after the + sign.  These search
       modes can be explicitly disabled by using  the  uppercase  variant  of
       those  letters: C and/or R.  When the string contains spaces, it needs
       to be enclosed in quotes.  To give an example: to open a file  at  the
       first occurrence of the word "Foo", you would do:
```



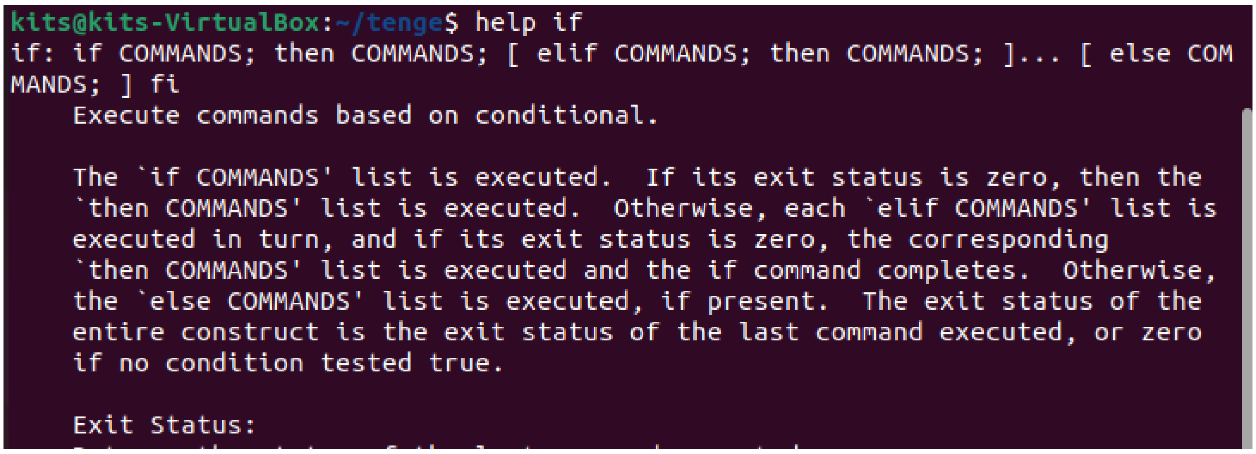
```
kits@kits-VirtualBox:~/tenge$ man nano
kits@kits-VirtualBox:~/tenge$ touch led
kits@kits-VirtualBox:~/tenge$ echo "Октябрь уже наступил-уж роща отряхает После
дние листы с нагих своих ветвей Дохнул осений хлад-дорога промерзает." >> led
kits@kits-VirtualBox:~/tenge$
```



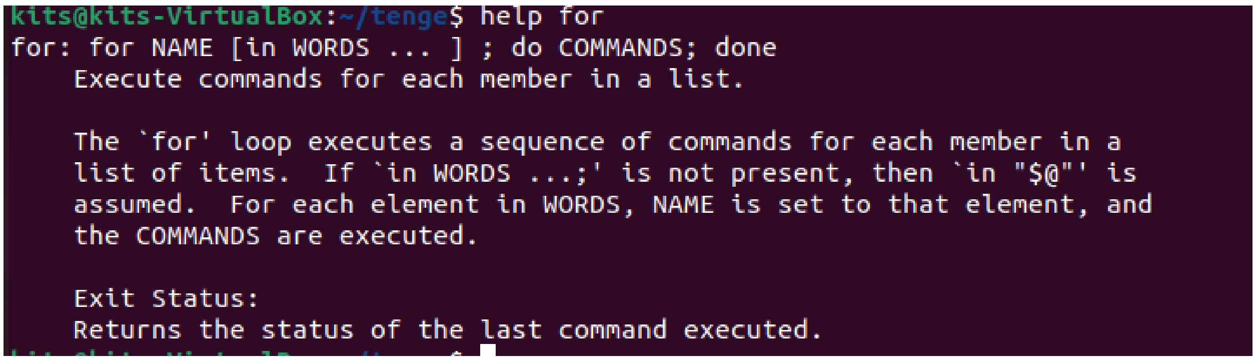
```
kits@kits-VirtualBox:~/tenge$ ls -l /etc/passwd
-rw-r--r-- 1 root root 2807 дек 12 23:59 /etc/passwd
```



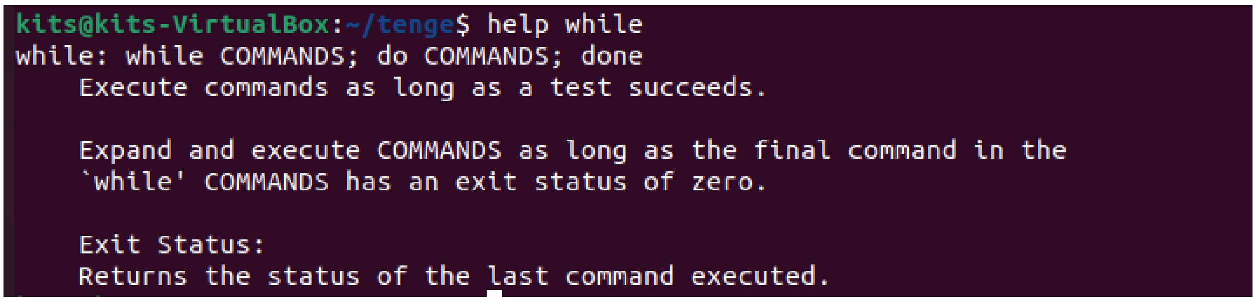
```
kits@kits-VirtualBox:~/tenge$ help
GNU bash, version 5.1.16(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                           history [-c] [-d offset] [n] or his>
 (( expression ))                       if COMMANDS; then COMMANDS; [ elif >
 . filename [arguments]                 jobs [-lnprs] [jobspec ...] or jobs>
 :                                      kill [-s sigspec | -n signum | -sig>
 [ arg... ]                             let arg [arg ...]
 [[ expression ]]                       local [option] name[=value] ...
 alias [-p] [name[=value] ... ]         logout [n]
 bg [job_spec ...]                      mapfile [-d delim] [-n count] [-O o>
 bind [-lpsvPSVX] [-m keymap] [-f fil>  popd [-n] [+N | -N]
 break [n]                              printf [-v var] format [arguments]
 builtin [shell-builtin [arg ...]]      pushd [-n] [+N | -N | dir]
 caller [expr]                          pwd [-LP]
 case WORD in [PATTERN [| PATTERN]...>  read [-ers] [-a array] [-d delim] [>
 cd [-L|[-P [-e]] [-@]] [dir]           readarray [-d delim] [-n count] [-O>
 command [-pVv] command [arg ...]       readonly [-aAf] [name[=value] ...] >
 compgen [-abcdefgjksuv] [-o option] >  return [n]
 complete [-abcdefgjksuv] [-pr] [-DEI>  select NAME [in WORDS ... ;] do COM>
 compopt [-o|+o option] [-DEI] [name >  set [-abefhkmnptuvxBCHP] [-o option>
 continue [n]                           shift [n]
 coproc [NAME] command [redirections>   shopt [-pqsu] [-o] [optname ...]
```



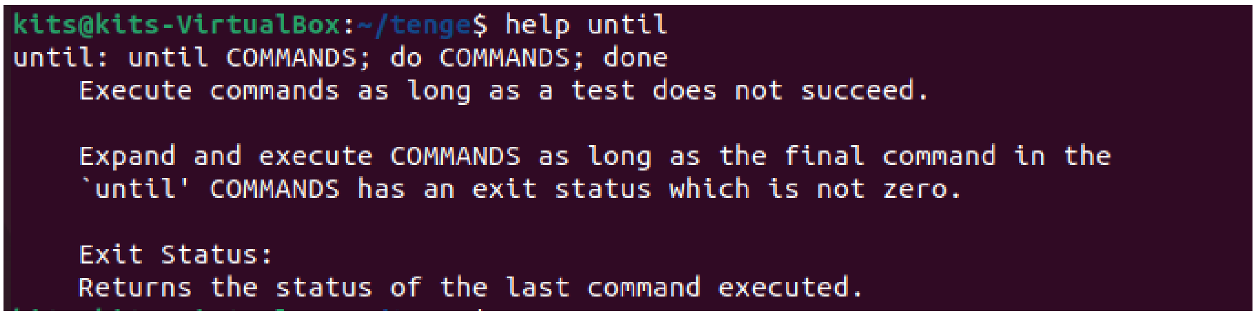
```
kits@kits-VirtualBox:~/tenge$ help if
if: if COMMANDS; then COMMANDS; [ elif COMMANDS; then COMMANDS; ]... [ else COM
MANDS; ] fi
    Execute commands based on conditional.

    The `if COMMANDS' list is executed.  If its exit status is zero, then the
    `then COMMANDS' list is executed.  Otherwise, each `elif COMMANDS' list is
    executed in turn, and if its exit status is zero, the corresponding
    `then COMMANDS' list is executed and the if command completes.  Otherwise,
    the `else COMMANDS' list is executed, if present.  The exit status of the
    entire construct is the exit status of the last command executed, or zero
    if no condition tested true.

    Exit Status:
```



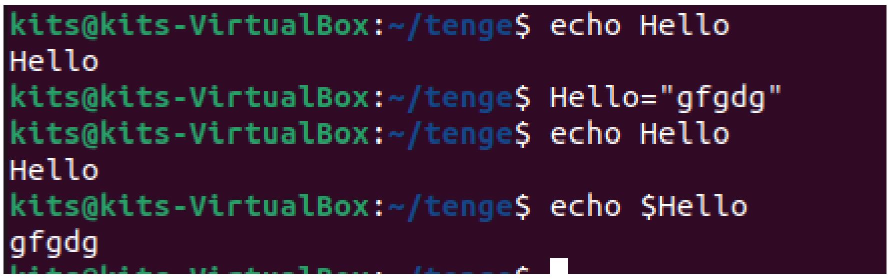
```
kits@kits-VirtualBox:~/tenge$ help for
for: for NAME [in WORDS ... ] ; do COMMANDS; done
    Execute commands for each member in a list.

    The `for' loop executes a sequence of commands for each member in a
    list of items.  If `in WORDS ...;' is not present, then `in "$@"' is
    assumed.  For each element in WORDS, NAME is set to that element, and
    the COMMANDS are executed.

    Exit Status:
    Returns the status of the last command executed.
```



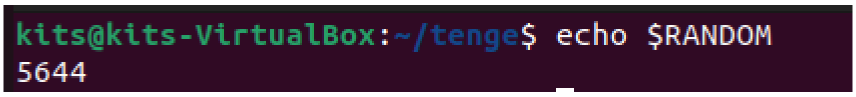
```
kits@kits-VirtualBox:~/tenge$ help while
while: while COMMANDS; do COMMANDS; done
    Execute commands as long as a test succeeds.

    Expand and execute COMMANDS as long as the final command in the
    `while' COMMANDS has an exit status of zero.

    Exit Status:
    Returns the status of the last command executed.
```



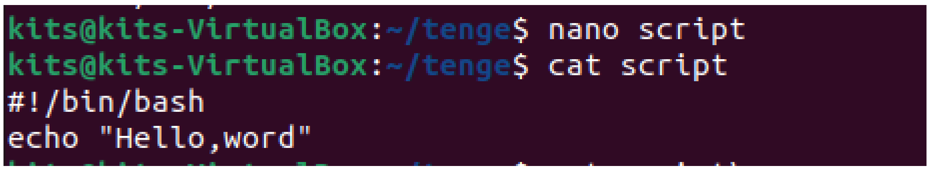
```
kits@kits-VirtualBox:~/tenge$ help until
until: until COMMANDS; do COMMANDS; done
    Execute commands as long as a test does not succeed.

    Expand and execute COMMANDS as long as the final command in the
    `until' COMMANDS has an exit status which is not zero.

    Exit Status:
    Returns the status of the last command executed.
```



```
kits@kits-VirtualBox:~/tenge$ echo Hello
Hello
kits@kits-VirtualBox:~/tenge$ Hello="gfgdg"
kits@kits-VirtualBox:~/tenge$ echo Hello
Hello
kits@kits-VirtualBox:~/tenge$ echo $Hello
gfgdg
```



```
kits@kits-VirtualBox:~/tenge$ echo $RANDOM
5644
```



```
kits@kits-VirtualBox:~/tenge$ nano script
kits@kits-VirtualBox:~/tenge$ cat script
#!/bin/bash
echo "Hello,word"
```