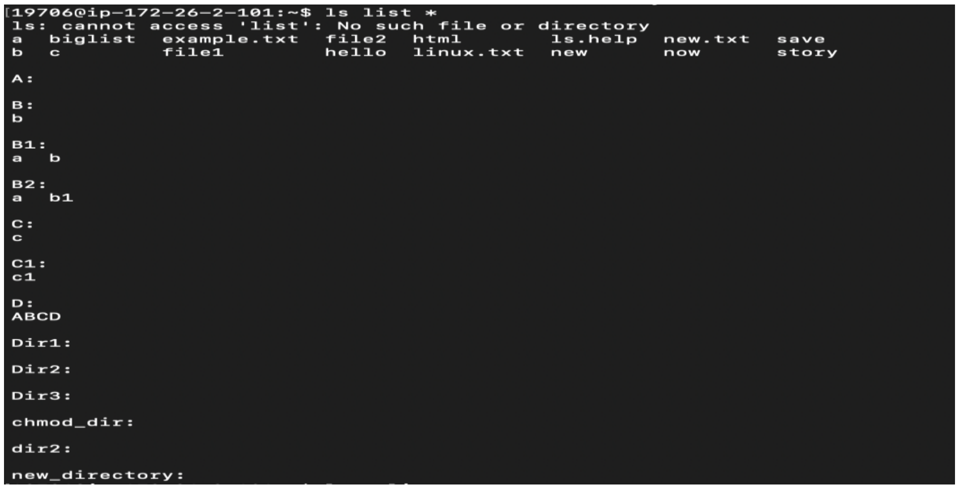
The lab is shorter this week because I want to talk about your project

Question no.1
Go to this website and finish all the steps in Tutorial Four. This tutorial talks about the difference between * and ?
http://www.ee.surrey.ac.uk/Teaching/Unix/unix4.html Links to an external site.

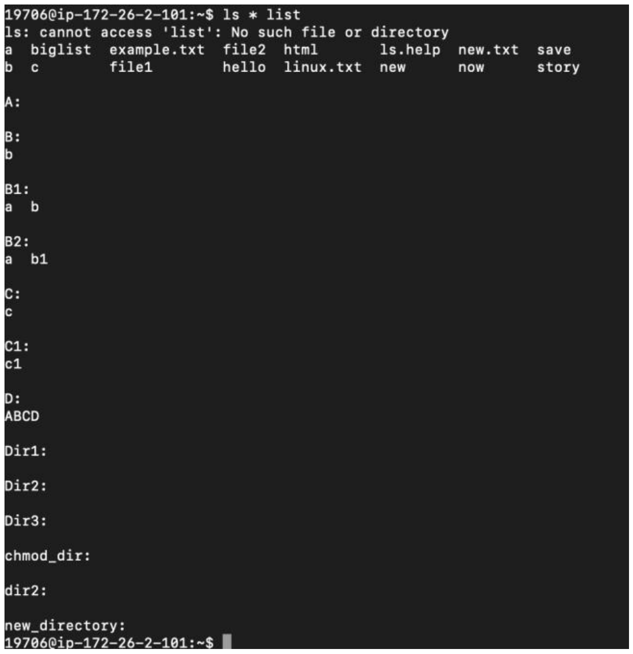
- ⇨ Wildcards:
  - i) *: This character represents any number of characters in a filename or path. When used in a command, the * character matches zero or more characters in a filename or path.

    a) ls list *:



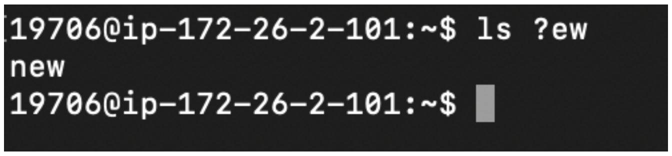
```
19706@ip-172-26-2-101:~$ ls list *
ls: cannot access 'list': No such file or directory
a   biglist   example.txt   file2   html        ls.help   new.txt   save
b   c         file1         hello   linux.txt   new       now       story

A:

B:
b

B1:
a   b

B2:
a   b1

C:
c

C1:
c1

D:
ABCD

Dir1:

Dir2:

Dir3:

chmod_dir:

dir2:

new_directory:
```

b) ls *list:



```
19706@ip-172-26-2-101:~$ ls * list
ls: cannot access 'list': No such file or directory
a  biglist  example.txt  file2  html       ls.help  new.txt  save
b  c        file1        hello  linux.txt  new      now      story

A:

B:
b

B1:
a  b

B2:
a  b1

C:
c

C1:
c1

D:
ABCD

Dir1:

Dir2:

Dir3:

chmod_dir:

dir2:

new_directory:
19706@ip-172-26-2-101:~$
```

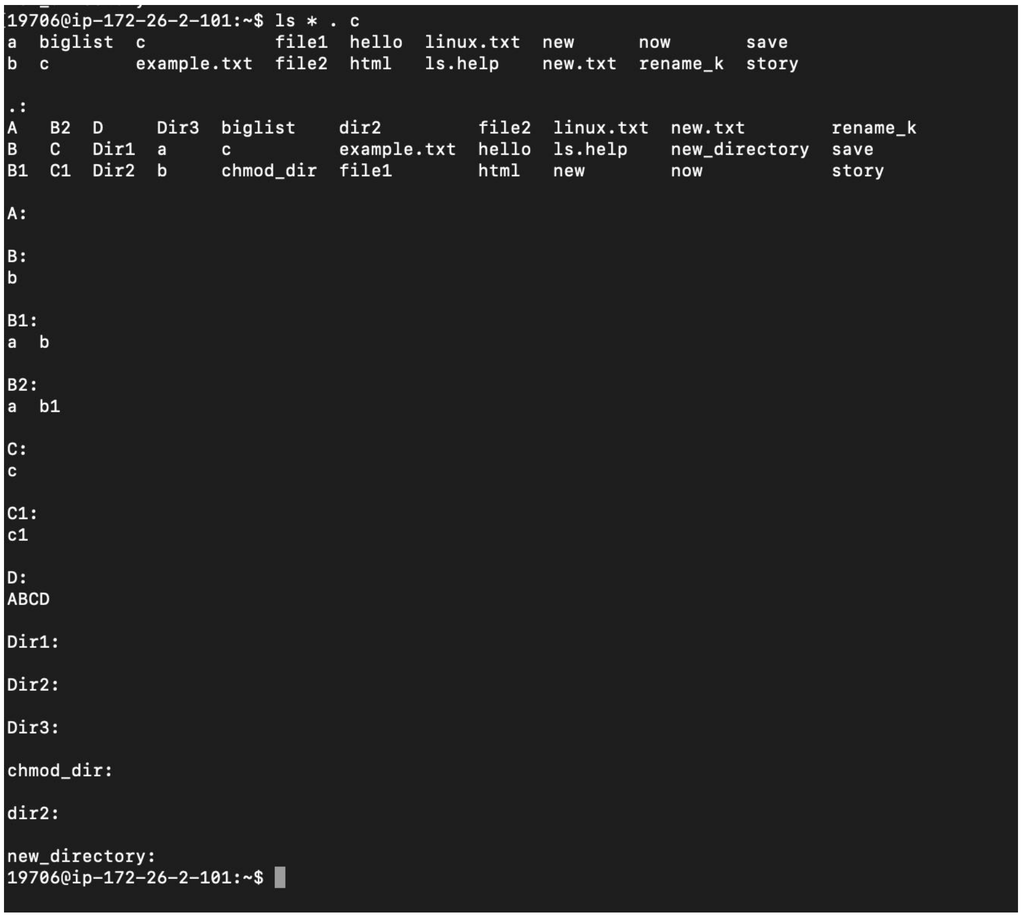
ii) ?: This character helps represents a single character in a filename or path. When used in a command, the "?" character matches any single character in a filename or path.



```
19706@ip-172-26-2-101:~$ ls ?ew
new
19706@ip-172-26-2-101:~$
```

⇨ Filename convention: The names of the file should not begin with a special character because all the special characters have a special meaning in the Linux terminal.

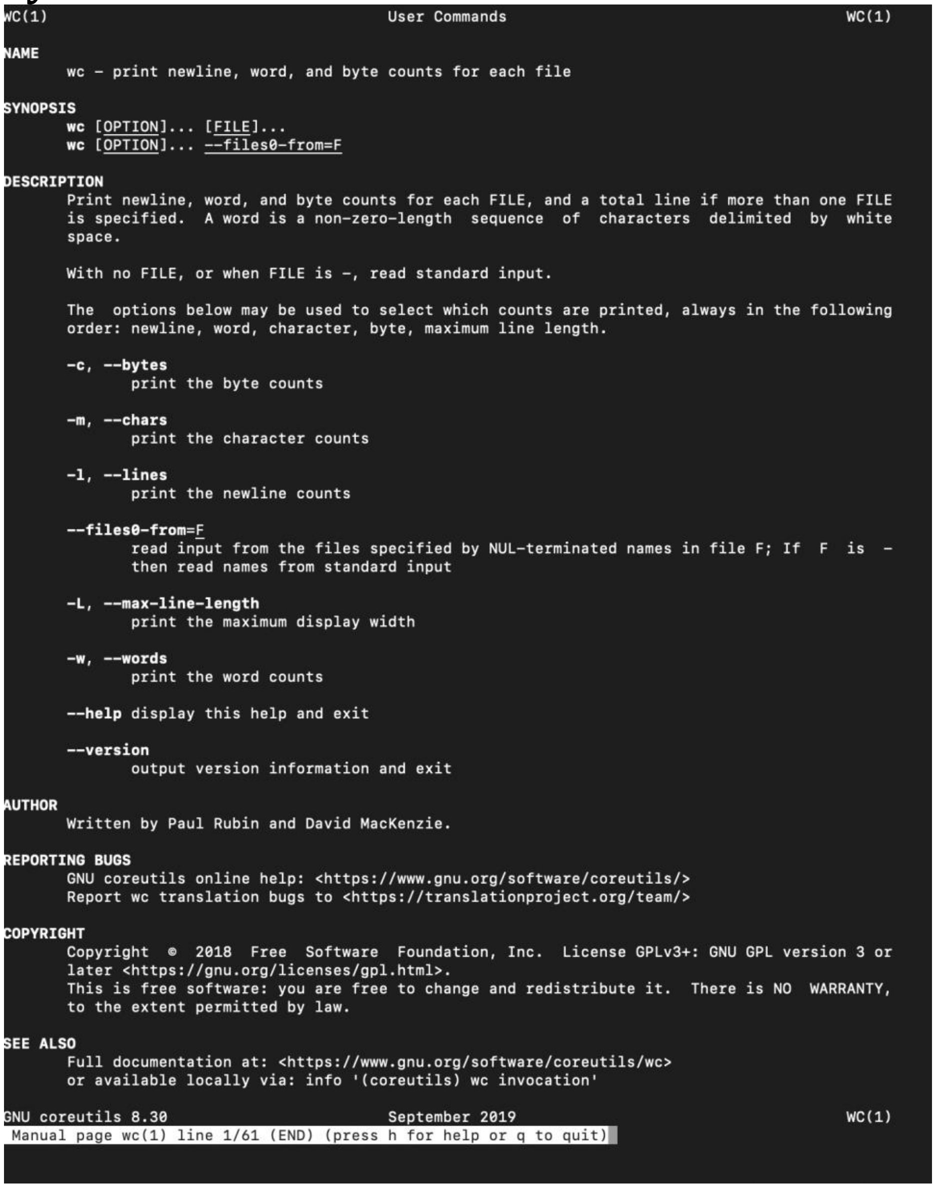
i) ls *.c: This command will list all files and directories in the current directory that have a name with one or more characters, followed by a space, and then the .c extension.



```
19706@ip-172-26-2-101:~$ ls * . c
a  biglist  c            file1  hello  linux.txt  new      now       save
b  c        example.txt  file2  html   ls.help    new.txt  rename_k  story

.:
A   B2  D     Dir3  biglist    dir2         file2  linux.txt  new.txt        rename_k
B   C   Dir1  a     c          example.txt  hello  ls.help    new_directory  save
B1  C1  Dir2  b     chmod_dir  file1        html   new        now            story

A:

B:
b

B1:
a  b

B2:
a  b1

C:
c

C1:
c1

D:
ABCD

Dir1:

Dir2:

Dir3:

chmod_dir:

dir2:

new_directory:
19706@ip-172-26-2-101:~$
```

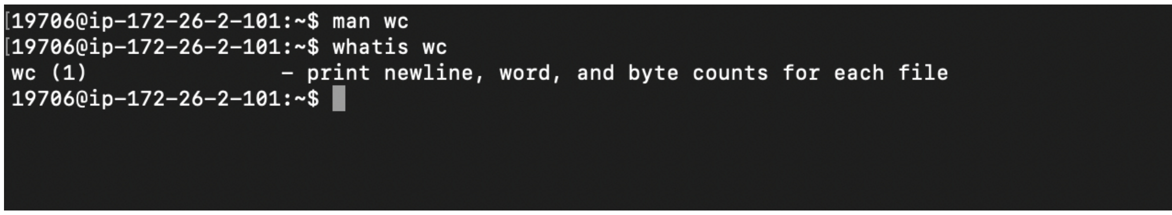
⇨ Online manuals:

i) man wc: It is used to display the manual pages for Unix commands, functions, and system calls.



```
WC(1)                            User Commands                           WC(1)

NAME
       wc - print newline, word, and byte counts for each file

SYNOPSIS
       wc [OPTION]... [FILE]...
       wc [OPTION]... --files0-from=F

DESCRIPTION
       Print newline, word, and byte counts for each FILE, and a total line if more than one FILE
       is specified.  A word is a non-zero-length  sequence  of  characters  delimited  by  white
       space.

       With no FILE, or when FILE is -, read standard input.

       The  options below may be used to select which counts are printed, always in the following
       order: newline, word, character, byte, maximum line length.

       -c, --bytes
              print the byte counts

       -m, --chars
              print the character counts

       -l, --lines
              print the newline counts

       --files0-from=F
              read input from the files specified by NUL-terminated names in file F; If  F  is  -
              then read names from standard input

       -L, --max-line-length
              print the maximum display width

       -w, --words
              print the word counts

       --help display this help and exit

       --version
              output version information and exit

AUTHOR
       Written by Paul Rubin and David MacKenzie.

REPORTING BUGS
       GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
       Report wc translation bugs to <https://translationproject.org/team/>

COPYRIGHT
       Copyright  ©  2018  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.

SEE ALSO
       Full documentation at: <https://www.gnu.org/software/coreutils/wc>
       or available locally via: info '(coreutils) wc invocation'

GNU coreutils 8.30                September 2019                         WC(1)
 Manual page wc(1) line 1/61 (END) (press h for help or q to quit)
```

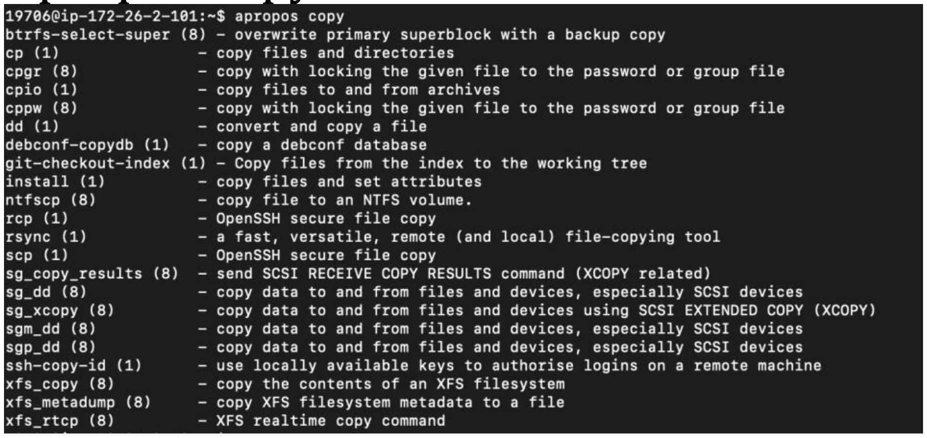
ii) whatis wc: It is used to display a brief one-line description of a command.



```
[19706@ip-172-26-2-101:~$ man wc
[19706@ip-172-26-2-101:~$ whatis wc
 wc (1)               - print newline, word, and byte counts for each file
 19706@ip-172-26-2-101:~$ 
```

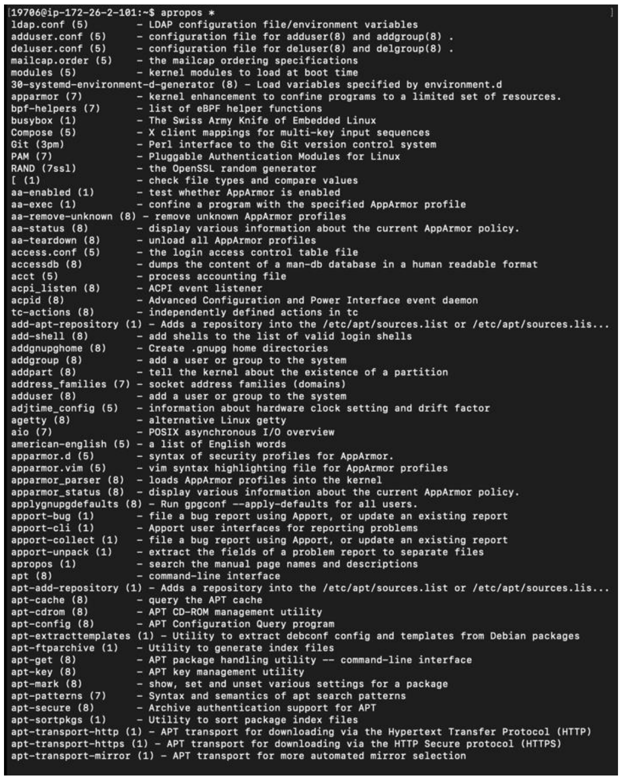
⇨ Apropos: It is used to search the manual pages for commands and keywords related to a specific topic.

i) Apropos copy:



```
19706@ip-172-26-2-101:~$ apropos copy
btrfs-select-super (8) - overwrite primary superblock with a backup copy
cp (1)               - copy files and directories
cpgr (8)             - copy with locking the given file to the password or group file
cpio (1)             - copy files to and from archives
cppw (8)             - copy with locking the given file to the password or group file
dd (1)               - convert and copy a file
debconf-copydb (1)   - copy a debconf database
git-checkout-index (1) - Copy files from the index to the working tree
install (1)          - copy files and set attributes
ntfscp (8)           - copy file to an NTFS volume.
rcp (1)              - OpenSSH secure file copy
rsync (1)            - a fast, versatile, remote (and local) file-copying tool
scp (1)              - OpenSSH secure file copy
sg_copy_results (8)  - send SCSI RECEIVE COPY RESULTS command (XCOPY related)
sg_dd (8)            - copy data to and from files and devices, especially SCSI devices
sg_xcopy (8)         - copy data to and from files and devices using SCSI EXTENDED COPY (XCOPY)
sgm_dd (8)           - copy data to and from files and devices, especially SCSI devices
sgp_dd (8)           - copy data to and from files and devices, especially SCSI devices
ssh-copy-id (1)      - use locally available keys to authorise logins on a remote machine
xfs_copy (8)         - copy the contents of an XFS filesystem
xfs_metadump (8)     - copy XFS filesystem metadata to a file
xfs_rtcp (8)         - XFS realtime copy command
```

ii) Apropos* :



```
19706@ip-172-26-2-101:~$ apropos *
ldap.conf (5)        - LDAP configuration file/environment variables
adduser.conf (5)     - configuration file for adduser(8) and addgroup(8) .
deluser.conf (5)     - configuration file for deluser(8) and delgroup(8) .
mailcap.order (5)    - the mailcap ordering specifications
modules (5)          - kernel modules to load at boot time
30-systemd-environment-d-generator (8) - Load variables specified by environment.d
apparmor (7)         - kernel enhancement to confine programs to a limited set of resources.
bpf-helpers (7)      - list of eBPF helper functions
busybox (1)          - The Swiss Army Knife of Embedded Linux
Compose (5)          - X client mappings for multi-key input sequences
Git (3pm)            - Perl interface to the Git version control system
PAM (7)              - Pluggable Authentication Modules for Linux
RAND (7ssl)          - the OpenSSL random generator
[ (1)                - check file types and compare values
aa-enabled (1)       - test whether AppArmor is enabled
aa-exec (1)          - confine a program with the specified AppArmor profile
aa-remove-unknown (8) - remove unknown AppArmor profiles
aa-status (8)        - display various information about the current AppArmor policy.
aa-teardown (8)      - unload all AppArmor profiles
access.conf (5)      - the login access control table file
accessdb (8)         - dumps the content of a man-db database in a human readable format
acct (5)             - process accounting file
acpi_listen (8)      - ACPI event listener
acpid (8)            - Advanced Configuration and Power Interface event daemon
tc-actions (8)       - independently defined actions in tc
add-apt-repository (1) - Adds a repository into the /etc/apt/sources.list or /etc/apt/sources.lis...
add-shell (8)        - add shells to the list of valid login shells
addgnupghome (8)     - Create .gnupg home directories
addgroup (8)         - add a user or group to the system
addpart (8)          - tell the kernel about the existence of a partition
address_families (7) - socket address families (domains)
adduser (8)          - add a user or group to the system
adjtime_config (5)   - information about hardware clock setting and drift factor
agetty (8)           - alternative Linux getty
aio (7)              - POSIX asynchronous I/O overview
american-english (5) - a list of English words
apparmor.d (5)       - syntax of security profiles for AppArmor.
apparmor.vim (5)     - vim syntax highlighting file for AppArmor profiles
apparmor_parser (8)  - loads AppArmor profiles into the kernel
apparmor_status (8)  - display various information about the current AppArmor policy.
applygnupgdefaults (8) - Run gpgconf --apply-defaults for all users.
apport-bug (1)       - file a bug report using Apport, or update an existing report
apport-cli (1)       - Apport user interfaces for reporting problems
apport-collect (1)   - file a bug report using Apport, or update an existing report
apport-unpack (1)    - extract the fields of a problem report to separate files
apropos (1)          - search the manual page names and descriptions
apt (8)              - command-line interface
apt-add-repository (1) - Adds a repository into the /etc/apt/sources.list or /etc/apt/sources.lis...
apt-cache (8)        - query the APT cache
apt-cdrom (8)        - APT CD-ROM management utility
apt-config (8)       - APT Configuration Query program
apt-extracttemplates (1) - Utility to extract debconf config and templates from Debian packages
apt-ftparchive (1)   - Utility to generate index files
apt-get (8)          - APT package handling utility -- command-line interface
apt-key (8)          - APT key management utility
apt-mark (8)         - show, set and unset various settings for a package
apt-patterns (7)     - Syntax and semantics of apt search patterns
apt-secure (8)       - Archive authentication support for APT
apt-sortpkgs (1)     - Utility to sort package index files
apt-transport-http (1) - APT transport for downloading via the Hypertext Transfer Protocol (HTTP)
apt-transport-https (1) - APT transport for downloading via the HTTP Secure protocol (HTTPS)
apt-transport-mirror (1) - APT transport for more automated mirror selection
```

➔ The difference between the * and ? are:
The asterisk (*) represents any number of characters, including zero. For example, the command ls *.txt will list all files in the current directory that end in .txt.

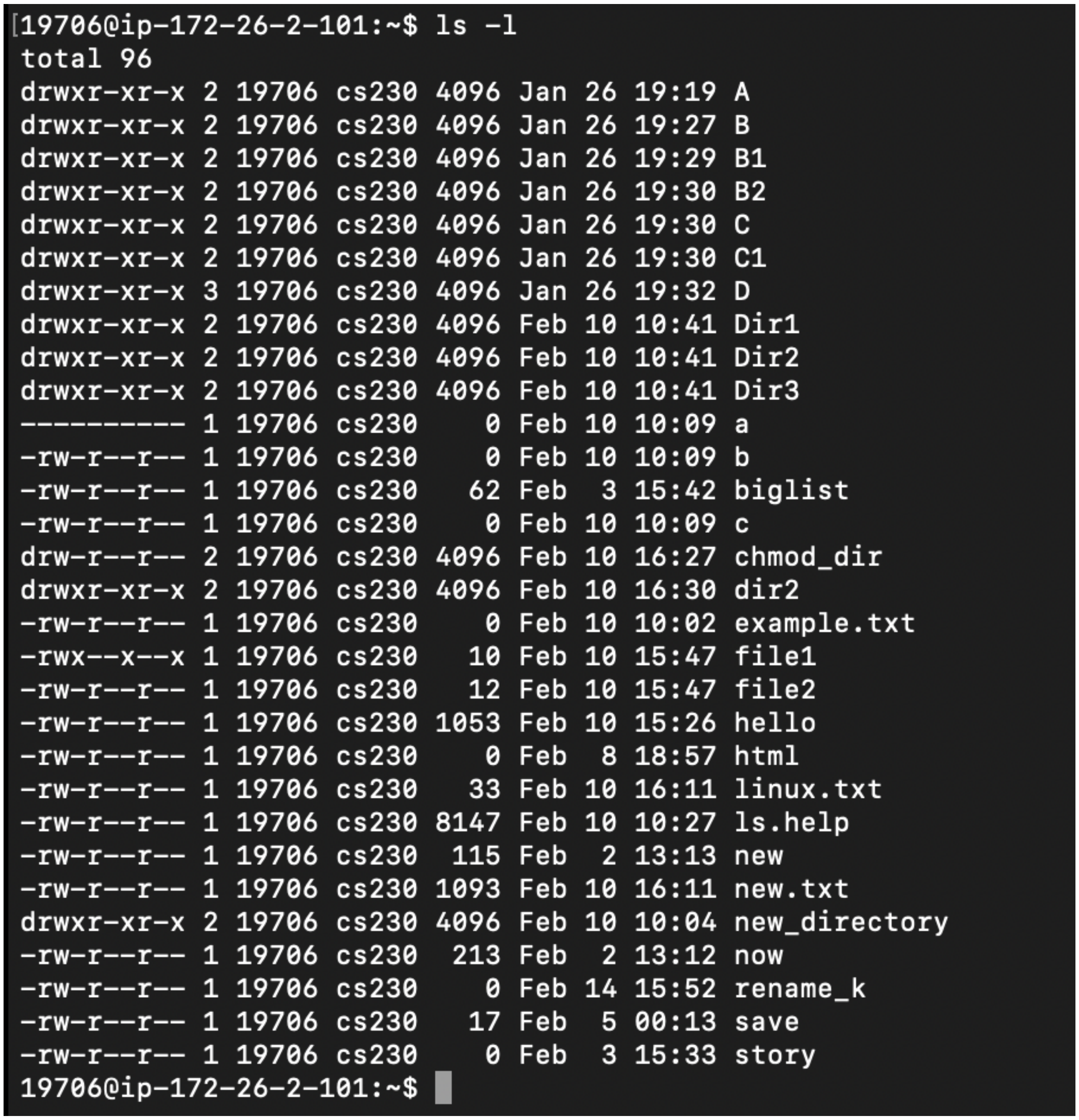
The question mark (?) represents any single character. For example, the command ls ?.txt will list all files in the current directory that have a single character before the .txt extension.

The main difference between the two is that the asterisk can represent any number of characters, while the question mark represents only a single character.

Question no.2

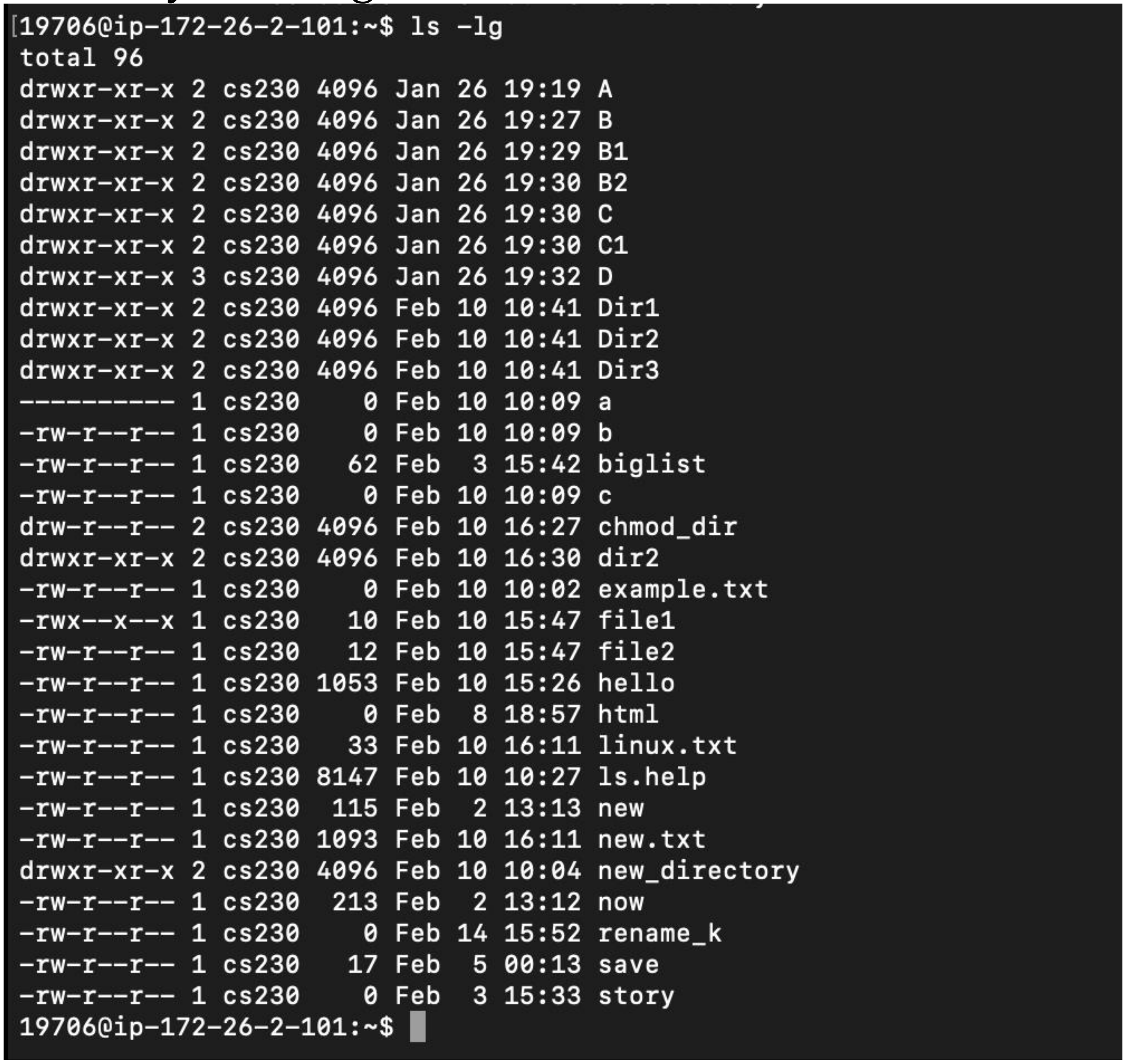
Go to this website and finish all the steps in Tutorial Five. This tutorial talks about these commands: fg, bg, jobs, ps, kill
http://www.ee.surrey.ac.uk/Teaching/Unix/unix5.html.

- ⇨ Ls -l: it is used to list the contents of a directory in a long format



```
19706@ip-172-26-2-101:~$ ls -l
total 96
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:19 A
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:27 B
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:29 B1
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:30 B2
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:30 C
drwxr-xr-x 2 19706 cs230 4096 Jan 26 19:30 C1
drwxr-xr-x 3 19706 cs230 4096 Jan 26 19:32 D
drwxr-xr-x 2 19706 cs230 4096 Feb 10 10:41 Dir1
drwxr-xr-x 2 19706 cs230 4096 Feb 10 10:41 Dir2
drwxr-xr-x 2 19706 cs230 4096 Feb 10 10:41 Dir3
---------- 1 19706 cs230    0 Feb 10 10:09 a
-rw-r--r-- 1 19706 cs230    0 Feb 10 10:09 b
-rw-r--r-- 1 19706 cs230   62 Feb  3 15:42 biglist
-rw-r--r-- 1 19706 cs230    0 Feb 10 10:09 c
drw-r--r-- 2 19706 cs230 4096 Feb 10 16:27 chmod_dir
drwxr-xr-x 2 19706 cs230 4096 Feb 10 16:30 dir2
-rw-r--r-- 1 19706 cs230    0 Feb 10 10:02 example.txt
-rwx--x--x 1 19706 cs230   10 Feb 10 15:47 file1
-rw-r--r-- 1 19706 cs230   12 Feb 10 15:47 file2
-rw-r--r-- 1 19706 cs230 1053 Feb 10 15:26 hello
-rw-r--r-- 1 19706 cs230    0 Feb  8 18:57 html
-rw-r--r-- 1 19706 cs230   33 Feb 10 16:11 linux.txt
-rw-r--r-- 1 19706 cs230 8147 Feb 10 10:27 ls.help
-rw-r--r-- 1 19706 cs230  115 Feb  2 13:13 new
-rw-r--r-- 1 19706 cs230 1093 Feb 10 16:11 new.txt
drwxr-xr-x 2 19706 cs230 4096 Feb 10 10:04 new_directory
-rw-r--r-- 1 19706 cs230  213 Feb  2 13:12 now
-rw-r--r-- 1 19706 cs230    0 Feb 14 15:52 rename_k
-rw-r--r-- 1 19706 cs230   17 Feb  5 00:13 save
-rw-r--r-- 1 19706 cs230    0 Feb  3 15:33 story
19706@ip-172-26-2-101:~$
```

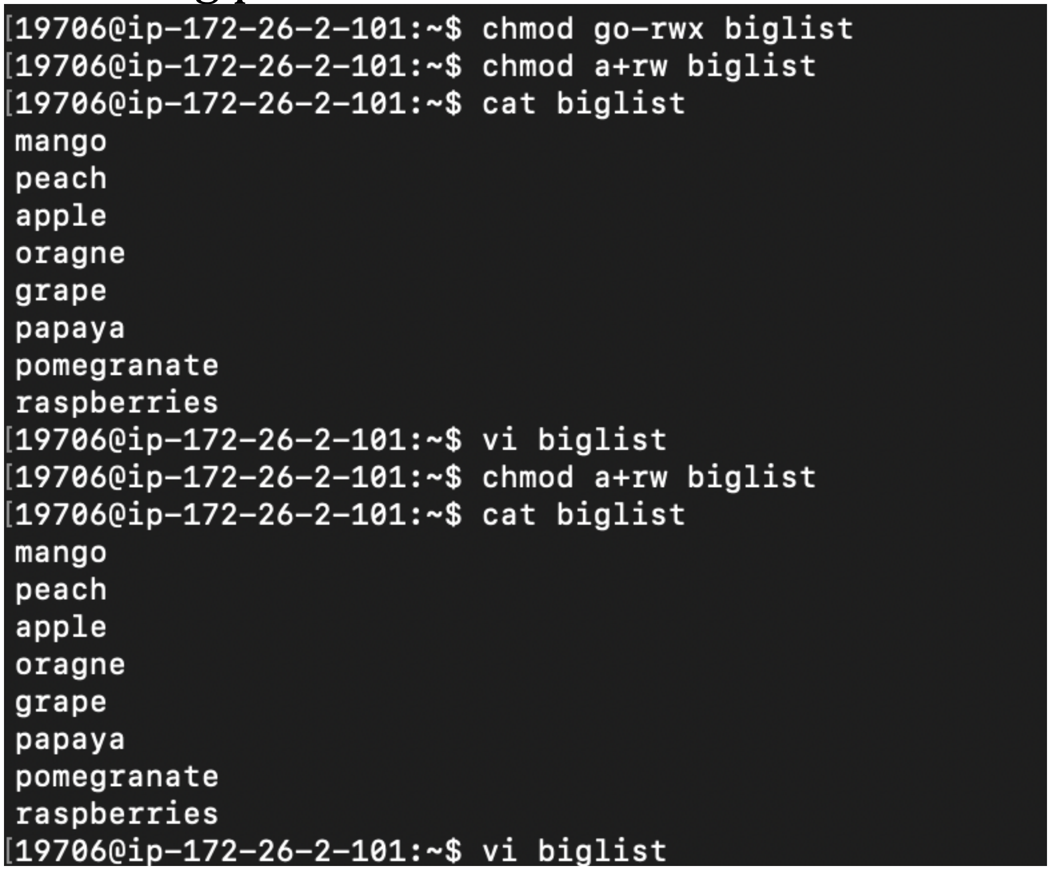
⇨ Ls-lg: It is also used to list the contents of a directory in a long format.



```
[19706@ip-172-26-2-101:~$ ls -lg
total 96
drwxr-xr-x 2 cs230 4096 Jan 26 19:19 A
drwxr-xr-x 2 cs230 4096 Jan 26 19:27 B
drwxr-xr-x 2 cs230 4096 Jan 26 19:29 B1
drwxr-xr-x 2 cs230 4096 Jan 26 19:30 B2
drwxr-xr-x 2 cs230 4096 Jan 26 19:30 C
drwxr-xr-x 2 cs230 4096 Jan 26 19:30 C1
drwxr-xr-x 3 cs230 4096 Jan 26 19:32 D
drwxr-xr-x 2 cs230 4096 Feb 10 10:41 Dir1
drwxr-xr-x 2 cs230 4096 Feb 10 10:41 Dir2
drwxr-xr-x 2 cs230 4096 Feb 10 10:41 Dir3
---------- 1 cs230    0 Feb 10 10:09 a
-rw-r--r-- 1 cs230    0 Feb 10 10:09 b
-rw-r--r-- 1 cs230   62 Feb  3 15:42 biglist
-rw-r--r-- 1 cs230    0 Feb 10 10:09 c
drw-r--r-- 2 cs230 4096 Feb 10 16:27 chmod_dir
drwxr-xr-x 2 cs230 4096 Feb 10 16:30 dir2
-rw-r--r-- 1 cs230    0 Feb 10 10:02 example.txt
-rwx--x--x 1 cs230   10 Feb 10 15:47 file1
-rw-r--r-- 1 cs230   12 Feb 10 15:47 file2
-rw-r--r-- 1 cs230 1053 Feb 10 15:26 hello
-rw-r--r-- 1 cs230    0 Feb  8 18:57 html
-rw-r--r-- 1 cs230   33 Feb 10 16:11 linux.txt
-rw-r--r-- 1 cs230 8147 Feb 10 10:27 ls.help
-rw-r--r-- 1 cs230  115 Feb  2 13:13 new
-rw-r--r-- 1 cs230 1093 Feb 10 16:11 new.txt
drwxr-xr-x 2 cs230 4096 Feb 10 10:04 new_directory
-rw-r--r-- 1 cs230  213 Feb  2 13:12 now
-rw-r--r-- 1 cs230    0 Feb 14 15:52 rename_k
-rw-r--r-- 1 cs230   17 Feb  5 00:13 save
-rw-r--r-- 1 cs230    0 Feb  3 15:33 story
19706@ip-172-26-2-101:~$ 
```

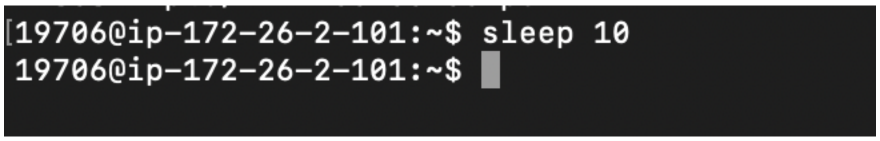
➔ The drwxr-xr-x, where d meaning is a directory, and -rw-r--r-- permissions, indicating that they are readable and writable by the owner, and readable by the group and others.

⇨ chmod: It is used to change the permissions of files and directories. There are three basic types of permissions: read (r), write (w), and execute (x). The letters "u", "g", and "o" represent the owner, group, and other users, respectively, and the symbols "+" and "-" represent adding or removing permission.



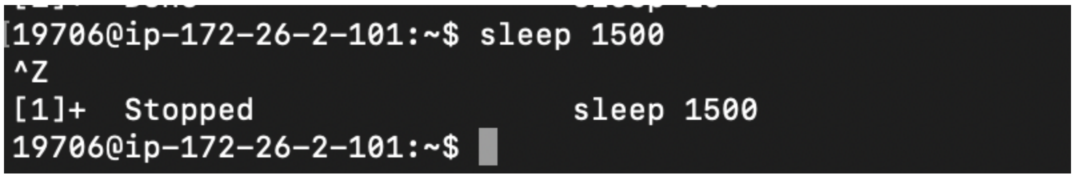
```
[19706@ip-172-26-2-101:~$ chmod go-rwx biglist
[19706@ip-172-26-2-101:~$ chmod a+rw biglist
[19706@ip-172-26-2-101:~$ cat biglist
mango
peach
apple
oragne
grape
papaya
pomegranate
raspberries
[19706@ip-172-26-2-101:~$ vi biglist
[19706@ip-172-26-2-101:~$ chmod a+rw biglist
[19706@ip-172-26-2-101:~$ cat biglist
mango
peach
apple
oragne
grape
papaya
pomegranate
raspberries
[19706@ip-172-26-2-101:~$ vi biglist
```

⇨ Sleep: It is used to pause a program or a script for a certain amount of time. The name "sleep" is derived from the fact that the command puts the program or script to sleep for a specified time.



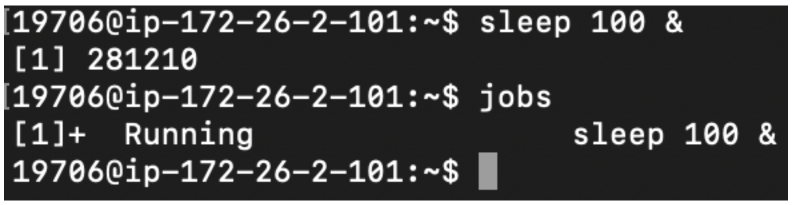
```
19706@ip-172-26-2-101:~$ sleep 10
19706@ip-172-26-2-101:~$ 
```

To terminal the sleep we can use the command "Ctrl +Z":



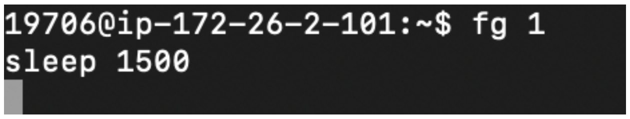
```
19706@ip-172-26-2-101:~$ sleep 1500
^Z
[1]+  Stopped                 sleep 1500
19706@ip-172-26-2-101:~$ 
```

⇨ Jobs: To run this command we need to execute the command "% sleep 10 &". After that, we can command the job and it will display a list of all active jobs, including the job number, status, and command. In this example, the job number is 1 and the status is "Running".



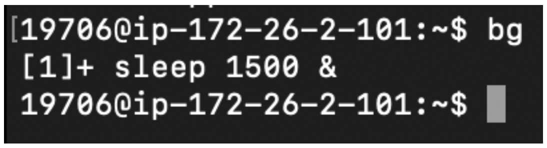
```
19706@ip-172-26-2-101:~$ sleep 100 &
[1] 281210
19706@ip-172-26-2-101:~$ jobs
[1]+  Running                 sleep 100 &
19706@ip-172-26-2-101:~$ 
```

i) fg: We can also use the job number to bring a job to the foreground or send it to the background.



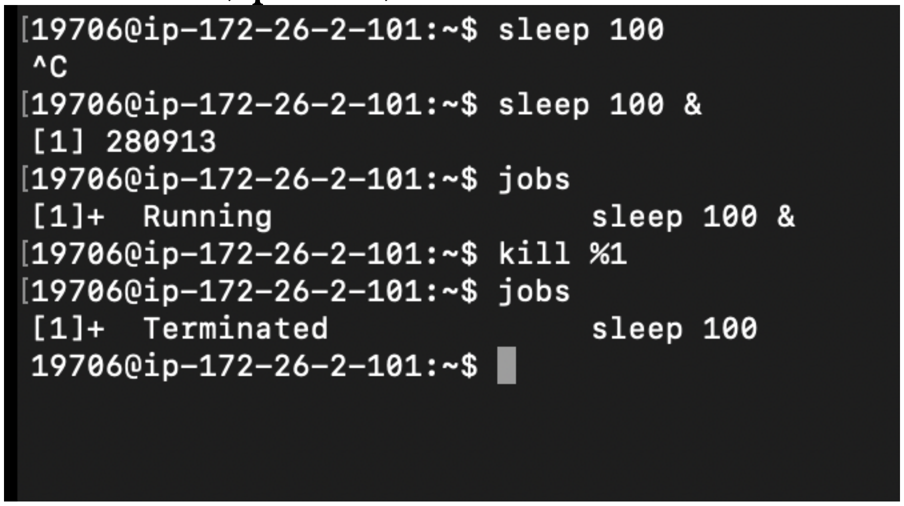
```
19706@ip-172-26-2-101:~$ fg 1
sleep 1500
```

ii) bg: To send the job back to the background, we can press 'Ctrl+Z' or 'Ctrl +C' to suspend the job, and then use the 'bg' command to resume it in the background.



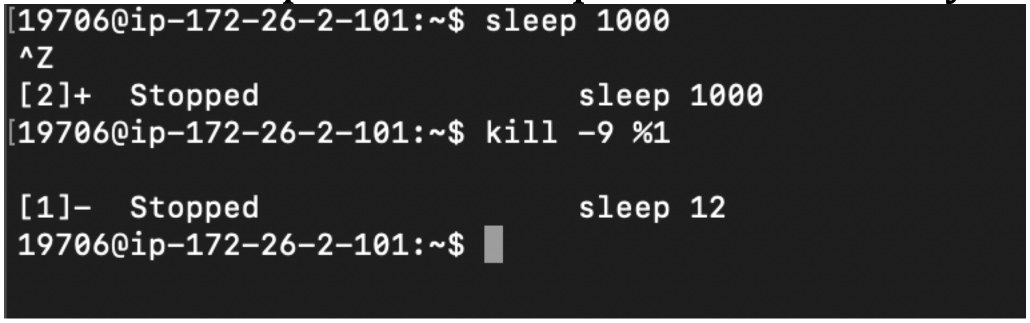
```
19706@ip-172-26-2-101:~$ bg
[1]+ sleep 1500 &
19706@ip-172-26-2-101:~$
```

⇨ Kill: It is used to send a signal to a process. The signal can be used to request that a process terminate, pause, or resume execution.



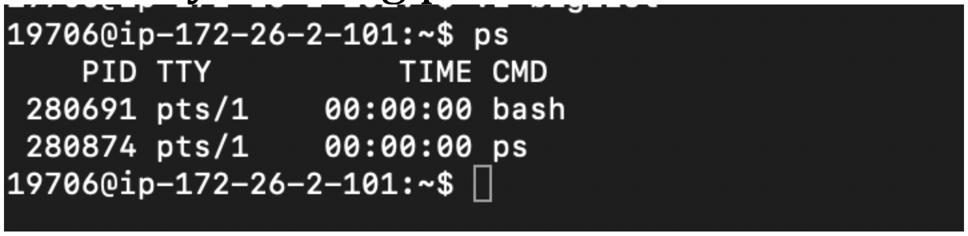
```
19706@ip-172-26-2-101:~$ sleep 100
^C
19706@ip-172-26-2-101:~$ sleep 100 &
[1] 280913
19706@ip-172-26-2-101:~$ jobs
[1]+  Running                 sleep 100 &
19706@ip-172-26-2-101:~$ kill %1
19706@ip-172-26-2-101:~$ jobs
[1]+  Terminated              sleep 100
19706@ip-172-26-2-101:~$
```

i) Kill -9: it helps to kill the process forcefully.



```
19706@ip-172-26-2-101:~$ sleep 1000
^Z
[2]+  Stopped                 sleep 1000
19706@ip-172-26-2-101:~$ kill -9 %1

[1]-  Stopped                 sleep 12
19706@ip-172-26-2-101:~$ 
```

⇨ Ps: It is used to display information about the currently running processes.



```
19706@ip-172-26-2-101:~$ ps
    PID TTY          TIME CMD
 280691 pts/1    00:00:00 bash
 280874 pts/1    00:00:00 ps
19706@ip-172-26-2-101:~$ 
```