List of Commands:

Please make note of the following before proceeding:

- You have root access in the container, hence your home is /root
- Semicolon (;) is being used to separate out individual commands i.e. type a command and then press enter and then type what follows semicolon and so on

1. pwd and cd:

a. Absolute paths: Try below and check where you have landed via pwd
   i. cd / ; pwd
   ii. cd /root/ ; pwd
   iii. cd /home/ ; pwd
   iv. cd /home/labDirectory/ ; pwd

b. Assuming you are in the /home folder; try the 3 commands below. All will end up in the same folder, check via pwd. The first is relative path, the latter two are absolute path. Remember to use “tab” for auto filling. “~” refers to the user's home directory and ".." is parent directory
   i. cd labDirectory ; pwd
   ii. cd /home/labdirectory/; pwd
   iii. cd ~/../home/labdirectory; pwd

c. Go to /home/labdirectory. Now try below in sequence and use pwd to figure out where you landed in each instance. Note “.” is current directory
   i. cd . ; pwd
   ii. cd ../ ; pwd
   iii. cd ../../ ; pwd
   iv. cd ; pwd
   v. cd - ; pwd

2. ls

a. Ensure you are in /home/labdirectory. Then try the following and check out what the options mean and what displays.
   i. ls
   ii. ls -a
   iii. ls -l
   iv. ls -al
   v. ls -R

b. You can also pass multiple directories and files separated by space. The first two arguments are directories but the last is a file!
   i. ls /etc /var /etc/passwd

c. Via web or man pages, find out what these options mean. Choose a relevant folder (e.g. /home/labdirectory) and checkout -X, -S, -t options

3. mkdir

a. Create a folder in /home/labdirectory. Ensure you are in the right folder first. The first command is using relative path. Next command uses absolute path.
   i. mkdir newfolder
   ii. mkdir /home/labdirectory/anotherfolder
b. Try recursion via -p option. Compare and contrast as to what happens between first and second command
   i. mkdir dir1/dir2
   ii. mkdir -p dir1/dir2
c. Try something more complex as shown in class
   i. mkdir -p Music/{Jazz/Blues,Folk,Disco,Rock/{Gothic,Punk,Progressive},Classical/Baroque/Early}
d. If you want to work with spaces in directory or file names, you need to escape them. Strongly encourage not to use spaces, instead go with _ or -. Nonetheless a few examples. Use ls to check what happened, after every command.
   i. mkdir "hello world"
   ii. mkdir 'hello again'
   iii. mkdir hello\ there

4. echo and >

a. In /home/labdirectory, try the following commands to understand echo. > is used to redirect out to a file, we will cover more on this in next class. You can redirect output of commands to a file as well via >
   i. echo "This is a test"
   ii. echo "This is a test" > file1
   iii. ls > file2
b. How to introduce new lines? -e option enables interpretation of backslash escapes. Try with and without.
   i. echo “start \n \n \n \n end”
   ii. echo -e “start \n \n \n \n end”

5. cat

a. cat just prints contents of a file, after any concatenation. * is a wildcard, we will learn more about it in next class. In this case, it selects all files that start with “file” i.e. file1 and file2
   i. cat file1
   ii. cat file1 file2
   iii. cat file*
b. cat can also be used with redirection.Can you figure out what > vs >> does?
   i. cat file1 > newfile; cat newfile
   ii. cat file2 > newfile; cat newfile
   iii. cat file2 >> newfile; cat newfile
c. You can use cat to number the lines
   i. cat -n /etc/lsb-release

6. **mv**
    a. Move a file and also reverse the move. Check what happened via ls after each command
        i. mv file1 dir1 ; ls ; ls dir1
        ii. mv dir1/file1 . ; ls ; ls dir1
    b. Can use this command to also copy files
        i. cat file1; mv file1 file2 ; ls ; cat file2

7. **cp**
    a. Undo previous move via cp
        i. cp file2 file1 ; cat file1; cat file2
    b. Can move files inside a directory also. In the second command, the file1 is overwritten. The third command below will overwrite content of file1 in dir1 with content of file2. What does -n do?
        i. echo “file1 content” > file1; echo “file2 content” > file2
        ii. cp file1 dir1/ ; cat dir1/file1
        iii. cp file2 dir1/file1 ; cat dir1/file1
        iv. cp -n file1 dir1/file1 ; cat dir1/file1
        v. cp file1 dir1/file1 ; cat dir1/file1
    c. Before overwriting, confirm
        i. cp -i file1 file2
    d. Recursive copy. Note there are already files in dir1, folder Music is added in there along with rest
        i. cp -R Music dir1

8. rm and rmdir

a. Check out below commands to remove files! In first command all files starting with file will get removed. In second command, rm will not remove Music folder, since it is not empty. Third command will remove though, since we are asking it to remove recursively. Fourth command also will not work since dir1 is not empty. 5th will work since the test folder is empty. 6th command asks before removing the file (type n or y)
    i. rm file*
    ii. rm Music
    iii. rm -r Music ; ls
    iv. rm -d dir1
    v. mkdir empty; rm -d empty
    vi. echo “create a file” > file; rm -i file

9. Miscellaneous

a. To clear the screen
    i. clear
b. List some top entries of a file. The first command lists top 10; Next two will list only the top 2.

i. echo -e " 1 \n 2 \n 3 \n 4 \n 5 \n 6 \n 7 \n 8 \n 9 \n 10 \n 11 \n 12" > file 1; head file1
ii. head -n 2 file1
iii. head -2 file1
iv. tail -n 3 file2

c. Check out man pages of various commands
   i. man ls; man cat ; man which; man ip
d. which shows the full path of the executable of that command
   i. which ls
   ii. which cp
   iii. which ip