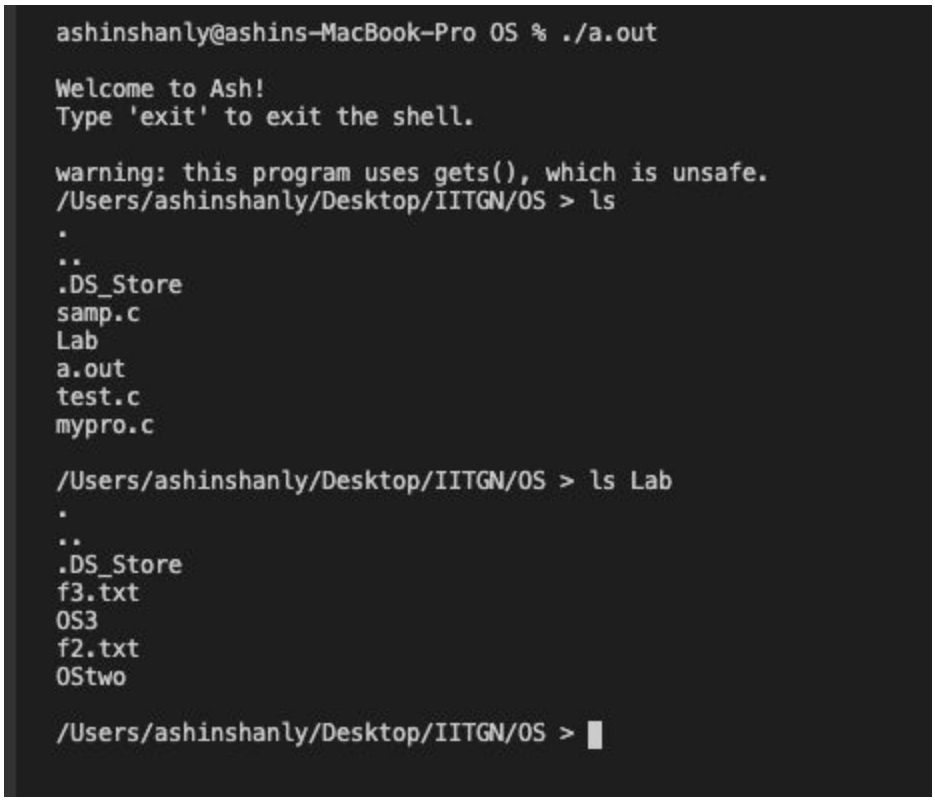
OS ASSIGNMENT-1

Submitted by: KS Ashin Shanly - 20250019

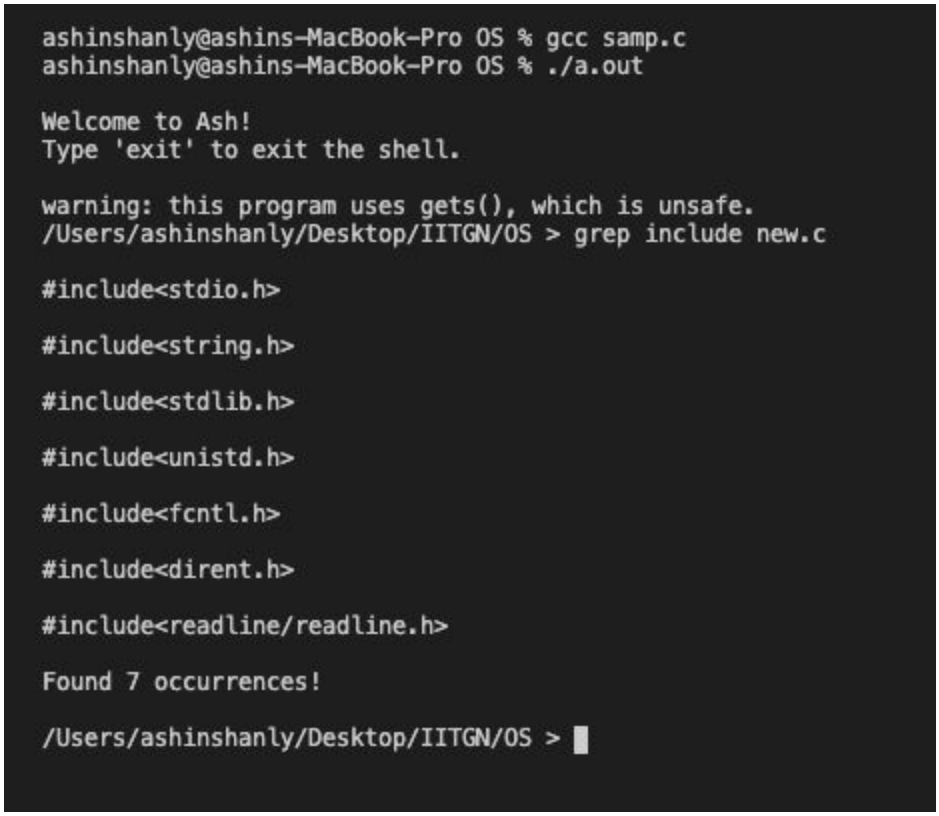
- **Ls command**

'ls' is a command to list computer files in Unix and Unix-like operating systems. It is specified by POSIX and the Single UNIX Specification. When invoked without any arguments, 'ls' lists the files in the current working directory. The directory is first opened using **opendir()** function in C and read using **readdir()** function. The names of the sub-directores and files are then printed as the output.
Output of the execution of 'ls' in the program is as below.



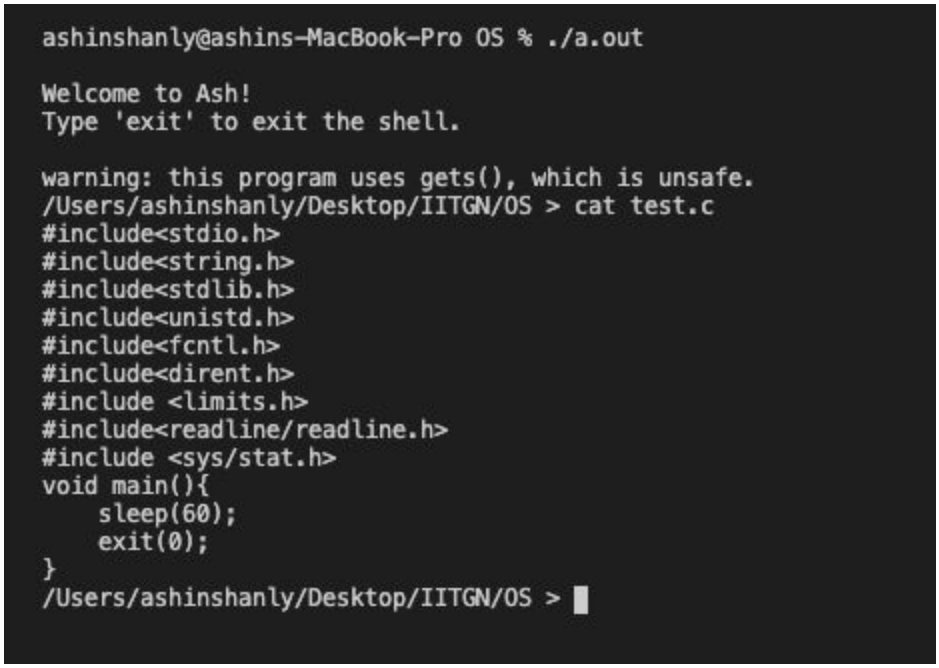
- **Grep command**

Grep is a Linux / Unix command-line tool used to search for a string of characters in a specified file. It will open the specified file and read it. Thereafter, it will check for the occurrence of the substring passed as argument as it is reading and will print the line if a match occurs. In this program, the function will also print the total number of occurrences of the string pattern in a specified file.



- **Cat command**

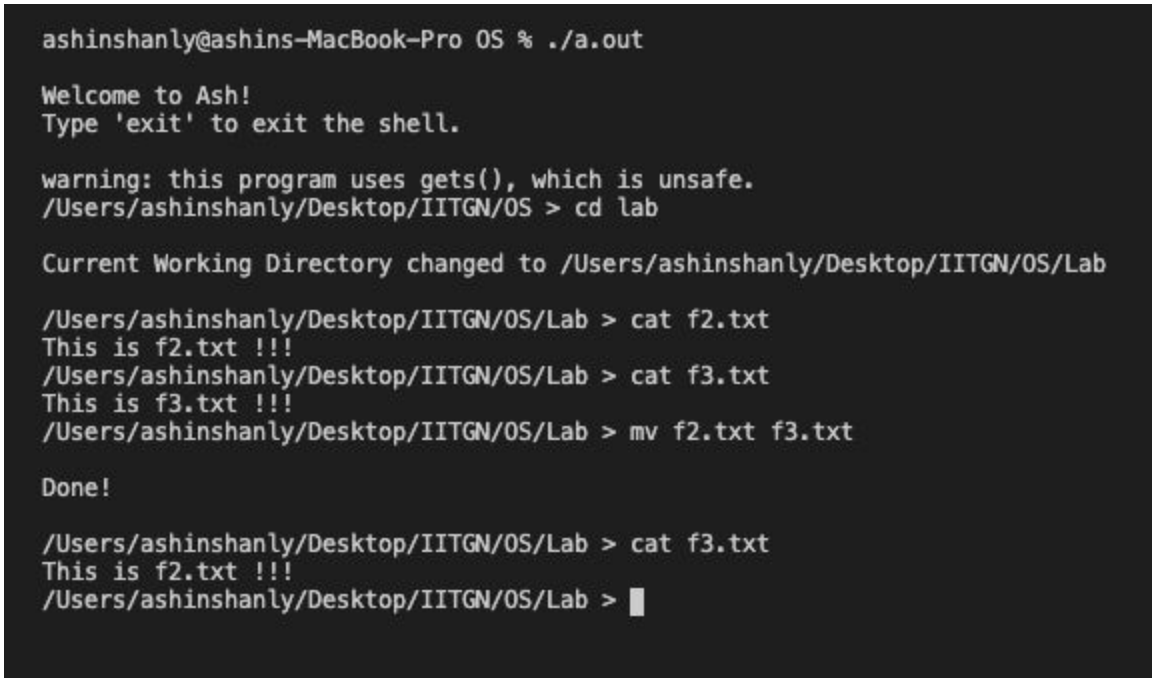
Cat(concatenate) command is used to read data from a file and give their content as output. It helps us to create, view and concatenate files. The argument passed along with cat is the file name to view. Here, the file test.c is first opened and read, after reading each line it is immediately printed as the output. This continues until the end of test.c(here) is reached.



- **Mv command**

mv stands for move. mv is used to move one or more files or directories from one place to another in a file system like UNIX. It has two distinct functions:
(i) It renames a file or folder.
(ii) It moves group of files to different directory.
No additional space is consumed on a disk during renaming. This command normally works silently means no prompt for confirmation.

Here, after executing the command f3.txt will also point to f2.txt now, and the original content of f3.txt is lost. This is accomplished by using the **rename()** function in C, which renames a file to another name.



- **<u>Cp command</u>**

cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. cp command requires at least two filenames in its arguments.

Here, f2.txt's contents will be copied to f3.txt by overwriting it's original contents. F2.txt will be first opened and read, after reading each line that line will be written to f3.txt. This continues until the end of f2.txt.

```
ashinshanly@ashins-MacBook-Pro OS % ./a.out

Welcome to Ash!
Type 'exit' to exit the shell.

warning: this program uses gets(), which is unsafe.
/Users/ashinshanly/Desktop/IITGN/OS > cd lab

Current Working Directory changed to /Users/ashinshanly/Desktop/IITGN/OS/Lab

/Users/ashinshanly/Desktop/IITGN/OS/Lab > cat f2.txt
This is f2.txt !!!
/Users/ashinshanly/Desktop/IITGN/OS/Lab > cat f3.txt
This is f3.txt !!!
/Users/ashinshanly/Desktop/IITGN/OS/Lab > cp f2.txt f3.txt

Done!

/Users/ashinshanly/Desktop/IITGN/OS/Lab > cat f2.txt
This is f2.txt !!!
/Users/ashinshanly/Desktop/IITGN/OS/Lab > cat f3.txt
This is f2.txt !!!
/Users/ashinshanly/Desktop/IITGN/OS/Lab >
```

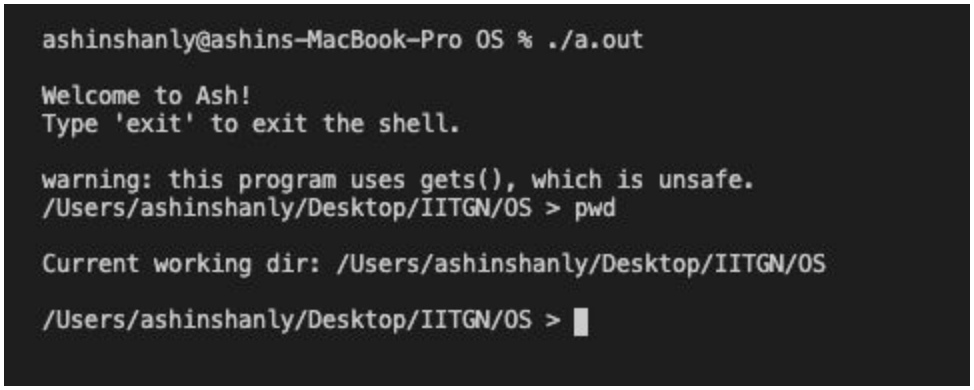
- **Cd command**

cd (change directory) command in linux is used to change the current working directory. **chdir()** function in C is used to accomplish this task. “Cd ..” will change the current working directory as the parent directory.

```
ashinshanly@ashins-MacBook-Pro OS % ./a.out

Welcome to Ash!
Type 'exit' to exit the shell.

warning: this program uses gets(), which is unsafe.
/Users/ashinshanly/Desktop/IITGN/OS > pwd

Current working dir: /Users/ashinshanly/Desktop/IITGN/OS

/Users/ashinshanly/Desktop/IITGN/OS > cd lab

Current Working Directory changed to /Users/ashinshanly/Desktop/IITGN/OS/Lab

/Users/ashinshanly/Desktop/IITGN/OS/Lab > cd ..

Current Working Directory changed to /Users/ashinshanly/Desktop/IITGN/OS

/Users/ashinshanly/Desktop/IITGN/OS >
```

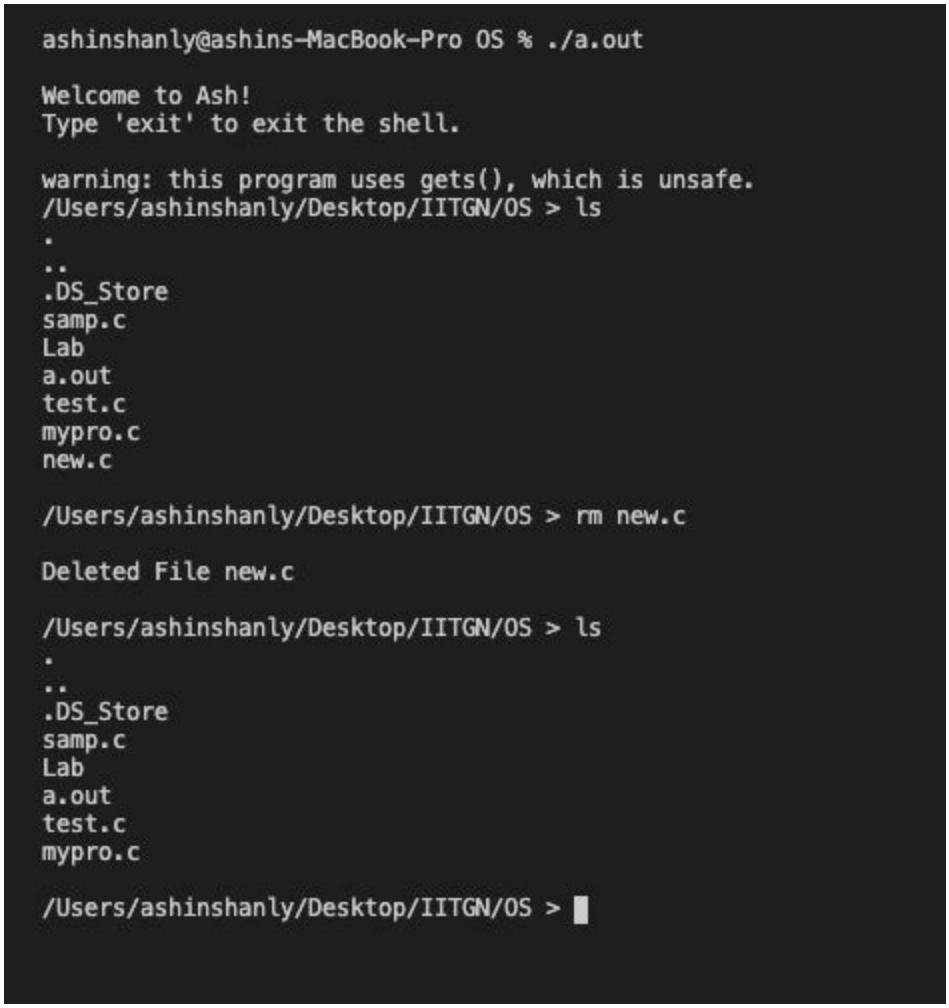
- **Pwd command**

pwd‘ stands for ‘Print Working Directory‘. As the name states, command ‘pwd‘ prints the current working directory or simply the directory user is, at present. It prints the current directory name with the complete path starting from root (/). The C function **getcwd()** is used for this purpose.



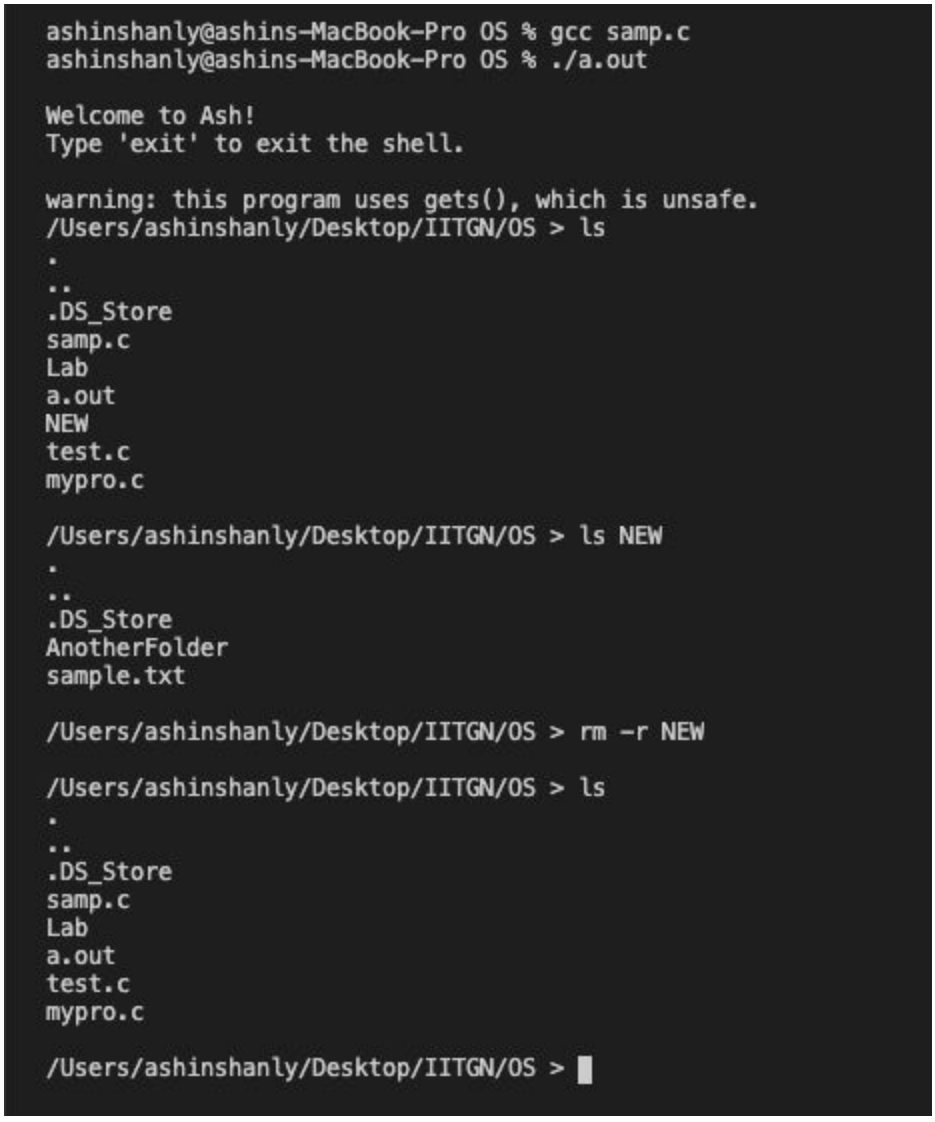
- **Rm command**

‘rm’ stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX. To be more precise, rm removes references to objects from the filesystem, where those objects might have had multiple references (for example, a file with two different names). By default, it does not remove non-empty directories. The C function **remove()** to implement this command.



- **Rm -r command**

This command will recursively delete all the sub-directories and files in a directory specified. After deleting all the contents, it will delete the mentioned directory itself. This is implemented using **ntfw()** function in C.

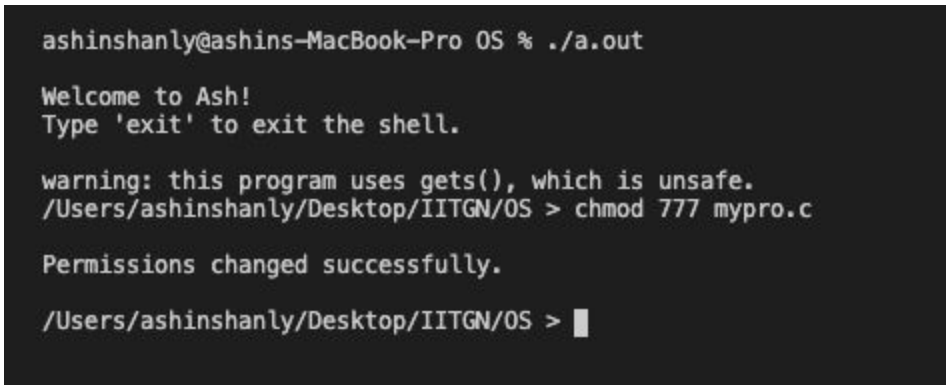


- **<u>Chmod command</u>**

In Unix-like operating systems, the chmod command is used to change the access mode of a file.
The name is an abbreviation of change mode.
The modes indicates which permissions are to be granted or removed from the specified classes. There are three basic modes which correspond to the basic permissions:

r -> Permission to read the file.
w -> Permission to write (or delete) the file.
x -> Permission to execute the file.

It is implemented using the **chmod()** function in C. The modes are passed as integers here.



- **mkdir command**

'mkdir' command is used to create a new directory. The C function mkdir() is used to implement the same. Here in the example, a new empty directory named "NEW" is created by passing the command "mkdir NEW".

- **<u>Run programs in the background using '&'</u>**

Entering the executable program name followed by a '&' at it's end will start executing the program in the background. This is accomplished by **forking** a child process and then replacing that with the program specified using the **execvp()** function in C. This way, the parent can continue executing while the child runs in the background. The child will exit after it is done without executing the remaining lineso of code.

```
ashinshanly@ashins-MacBook-Pro OS % ./a.out

Welcome to Ash!
Type 'exit' to exit the shell.

warning: this program uses gets(), which is unsafe.
/Users/ashinshanly/Desktop/IITGN/OS > cat test.c
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include<fcntl.h>
#include<dirent.h>
#include <limits.h>
#include<readline/readline.h>
#include <sys/stat.h>
void main(){
    sleep(60);
    exit(0);
}
/Users/ashinshanly/Desktop/IITGN/OS > test&

/Users/ashinshanly/Desktop/IITGN/OS >
```

```
ashinshanly@ashins-MacBook-Pro ~ % ps
  PID TTY           TIME CMD
  649 ttys000    0:00.18 /bin/zsh -l
 1809 ttys000    0:00.00 ./a.out
 1816 ttys000    0:00.00 (test)
 1786 ttys001    0:00.06 -zsh
ashinshanly@ashins-MacBook-Pro ~ %
```

- **Inbuilt binaries**

```
ashinshanly@ashins-MacBook-Pro OS % ./a.out

Welcome to Ash!
Type 'exit' to exit the shell.

warning: this program uses gets(), which is unsafe.
/Users/ashinshanly/Desktop/IITGN/OS > ps
  PID TTY           TIME CMD
  649 ttys000    0:00.25 /bin/zsh -l
 2292 ttys000    0:00.00 ./a.out
 2236 ttys001    0:00.03 -zsh
over
/Users/ashinshanly/Desktop/IITGN/OS >
```

All the inbuilt binary commands such as ps, wget and pmap are inserted in an array of characters. When the user inputs a command, it is first checked to see if it matches any of the inbuilt binaries. If any match occurs they are directly executed if not then the other command names are checked and so on.

NOTE:

- If none of the commands match, the user is asked to check the command entered and retry!
- The program will terminate if the user inputs “exit”.