Operating System

Lab Report

Lab 07

Group Members Name & Reg #:	**Muhammad Haris Irfan** **(FA18-BCE-090)**
Class	Operating System (**BCE-5B**)
Instructor's Name	Sir Wasif

In Lab Tasks

Question 1: Activity 1:

In this activity, you are required to perform tasks given below:

1. Print something and Check id of the parent process

2. Create a child process and print child process id in parent process

3. Create a child process and print child process id in child process

Solution:

1) The Terminal Output is given below,

```
8 printf( "Parent ID is: %d\n ",getpid());
```

```
Parent ID is: 7179
```

2) The Terminal Output is given below,

```
13 if(forkvalue !=0)
14 {
15 printf("%d is the parent, my child's pid is %d \n", getpid(),forkvalue);
16 }
```

```
7179 is the parent, my child's pid is 7180
```

3) The Terminal Output is given below,

checkid.c
~/Documents/C++

checkid | checkid.c

```
1 #include <stdio.h>
2 #include <unistd.h>
3 #include <sys/wait.h>
4
5 int main()
6 {
7 int forkvalue;
8 printf( "Parent ID is: %d\n ",getpid());
9 printf( "Going to fork %d\n", getpid());
10
11 forkvalue=fork();
12
13 if(forkvalue !=0)
14 {
15 printf("%d is the parent, my child's pid is %d \n", getpid(),forkvalue);
16 }
17
18 else
19 {
20  printf(" %d, I am the child. \n", getpid());
21 }
22
23 printf( "%d, I am the Parent. \n",getpid());
24 return 0;
25 }
```

C | Tab Width: 8 | Ln 17, Col 1 | INS

haris@haris-VirtualBox: ~/Documents/C++

```
haris@haris-VirtualBox:~/Documents/C++$ gedit checkid.c
haris@haris-VirtualBox:~/Documents/C++$ g++ checkid.c -o checkid
haris@haris-VirtualBox:~/Documents/C++$ ./checkid
Parent ID is: 7179
 Going to fork 7179
7179 is the parent, my child's pid is 7180
7179, I am the Parent.
haris@haris-VirtualBox:~/Documents/C++$  7180, I am the child.
7180, I am the Parent.
```

Question 2: Activity 2:

1. Create a process and make it an orphan.

Hint: To, illustrate this insert a sleep statement into the child's code. This ensured that the parent process terminated before its child.

Steps to create an orphan process:

1. print something and get its pid and ppid
2. create a child process
3. Now as a parent process print parent id and id of child process
4. Make child sleep for 5 seconds
5. Now while child is sleeping parent will terminate. Print parent id of child to make sure it is orphaned (PPID has been changed)

Solution:

1) The Terminal Output is given below,

```
6 printf( "PID: %d and PPID: %d\n ",getpid(),getppid());
```

```
PID: 22165 and PPID: 2942
```

2) The Terminal Output is given below,

```
11 if(pid !=0)
12 {
13 printf("%d is the parent pid and ppid is %d \n", getpid(),getppid());
14 printf("Child pid is %d\n",pid);
15 }
16
```

```
 22165 is the parent pid and ppid is 2942
Child pid is 22166
```

3) The Terminal Output is given below,

```
11 if(pid !=0)
12 {
13 printf("%d is the parent pid and ppid is %d \n", getpid(),getppid());
14 printf("Child pid is %d\n",pid);
15 }
16
```

```
 22165 is the parent pid and ppid is 2942
Child pid is 22166
```

4) The Terminal Output is given below,

```
19  sleep(5);
```

5) The Terminal Output is given below,

Open — task2.c — ~/Documents/C++ — Save

```
1 #include <stdio.h>
2 #include <unistd.h>
3 int main()
4 {
5 int pid;
6 printf( "PID: %d and PPID: %d\n ",getpid(),getppid());
7
8
9 pid=fork();
10
11 if(pid !=0)
12 {
13 printf("%d is the parent pid and ppid is %d \n", getpid(),getppid());
14 printf("Child pid is %d\n",pid);
15 }
16
17 else
18 {
19  sleep(5);
20  printf(" I am the child with pid %d and ppid %d. \n", getpid(),getppid());
21 }
22
23 printf( "%d PID is ending . \n",getpid());
24 return 0;
25 }
```

INS

haris@haris-VirtualBox: ~/Documents/C++

```
haris@haris-VirtualBox:~/Documents/C++$ g++ task2.c -o task2
haris@haris-VirtualBox:~/Documents/C++$ ./task2
PID: 22165 and PPID: 2942
 22165 is the parent pid and ppid is 2942
Child pid is 22166
22165 PID is ending .
haris@haris-VirtualBox:~/Documents/C++$   I am the child with pid 22166 and ppid
 1144.
22166 PID is ending .
```

Question 3: Activity 3:

Create a process and make it a Zombie.

1. Execute fork to create a child
2. In parent process (using if statement) Create an infinite loop so that it never terminates and never executes wait ()
3. Make parent sleep for 100 sec
4. Terminate child process exit using exit.

Now this child is a zombie because no parent is waiting for him

To view status of processes, use the following commands on command line:

1. Execute the c code in background using ./abc &
2. View process status using 'ps -lf'
3. You will see zombie process with state 'Z' in STAT column
4. Get parent id and kill parent process using "kill (parent id)" command
5. again execute 'ps -lf'
6. Note that Zombie is gone now

Solution:

1) The Terminal Output is given below,

```
9 pid=fork();
```

2) The Terminal Output is given below,

```
13 while (1)
14 {
15 sleep(100);
16 }
```

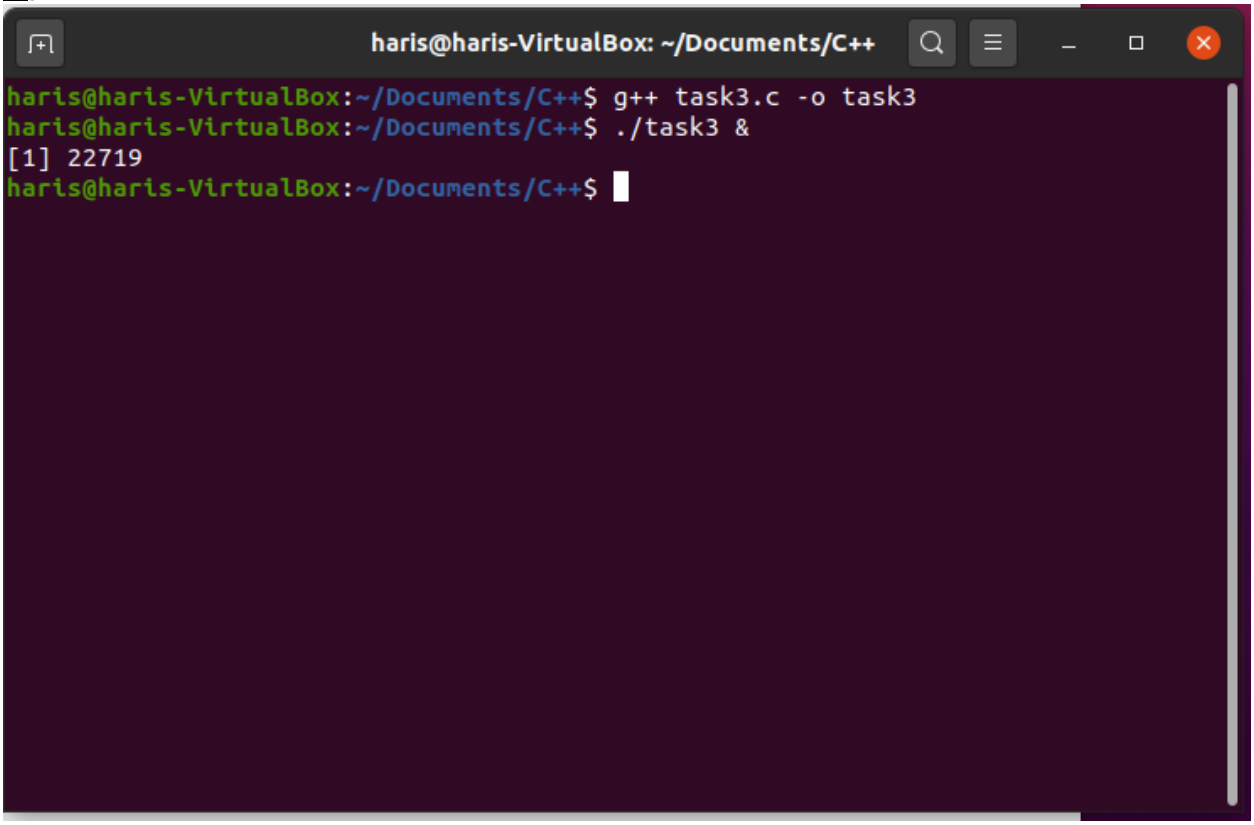
3) The Terminal Output is given below,

```
15 sleep(100);
```

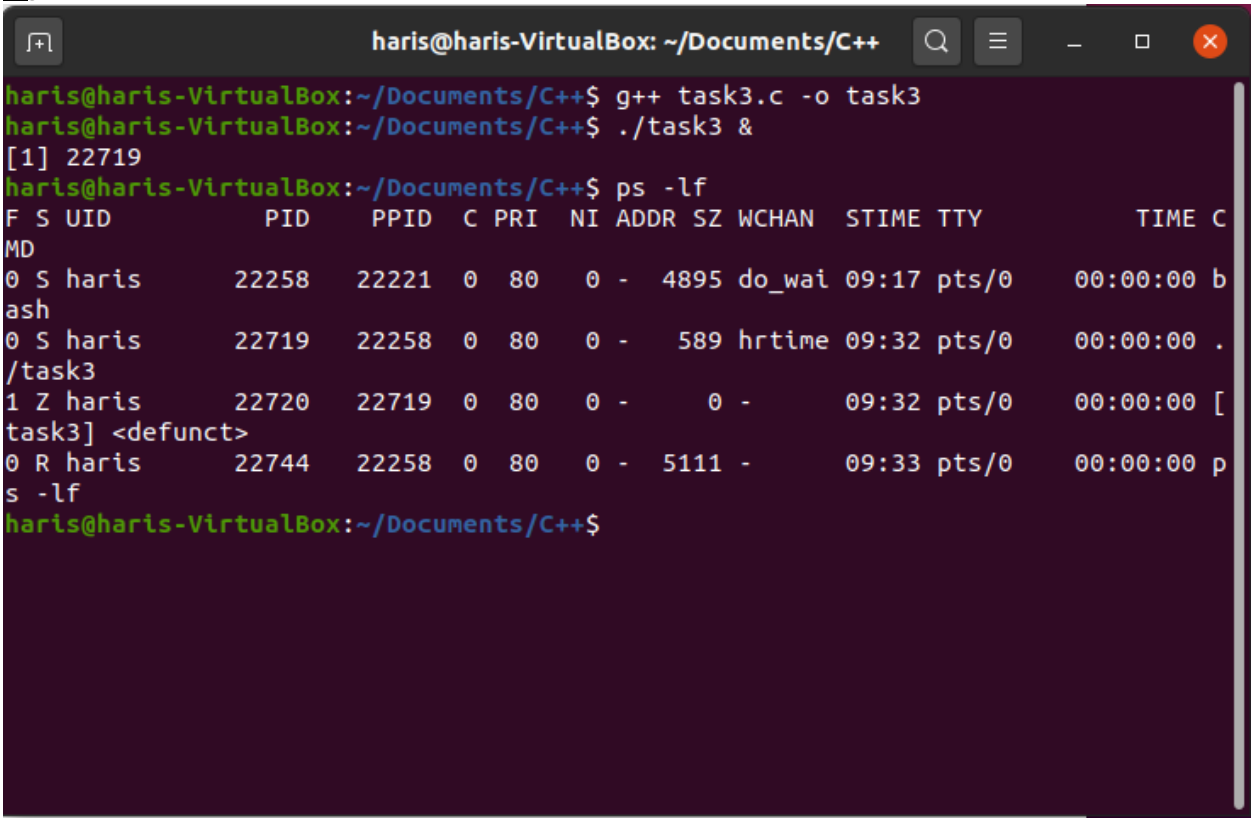
4) The Terminal Output is given below,

```
20 exit(0);
```

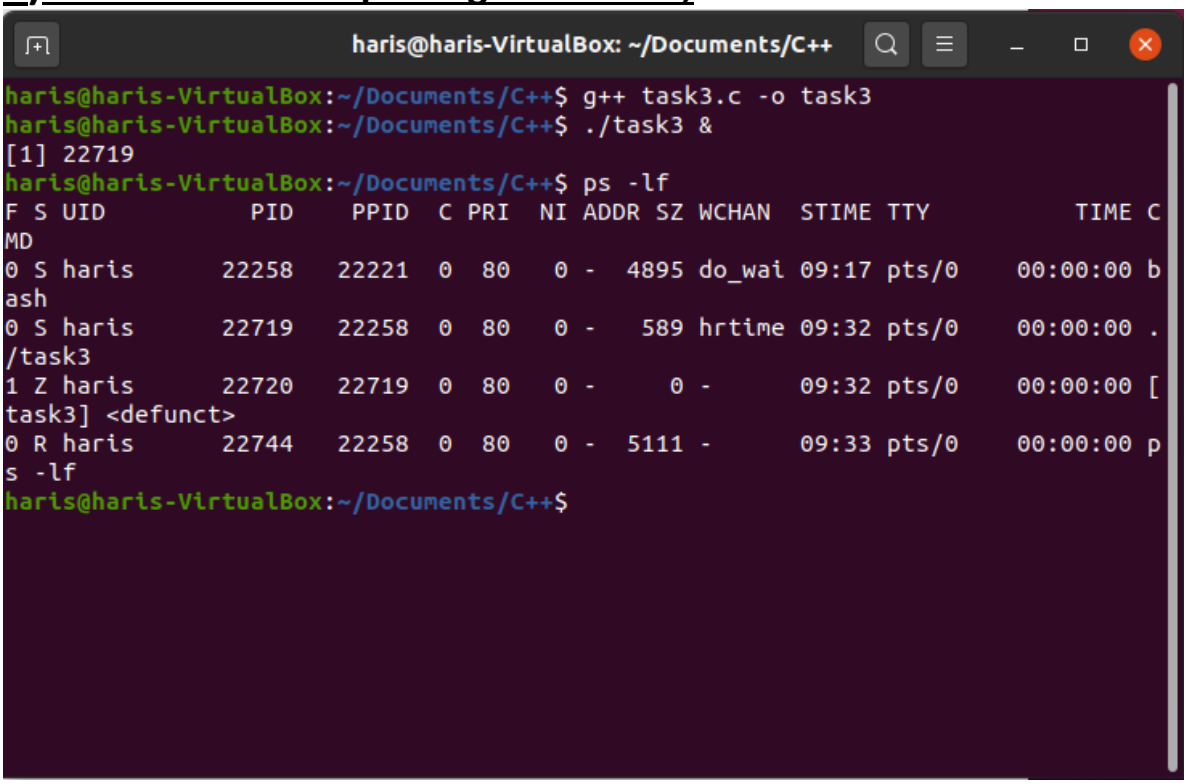
1) The Terminal Output is given below,



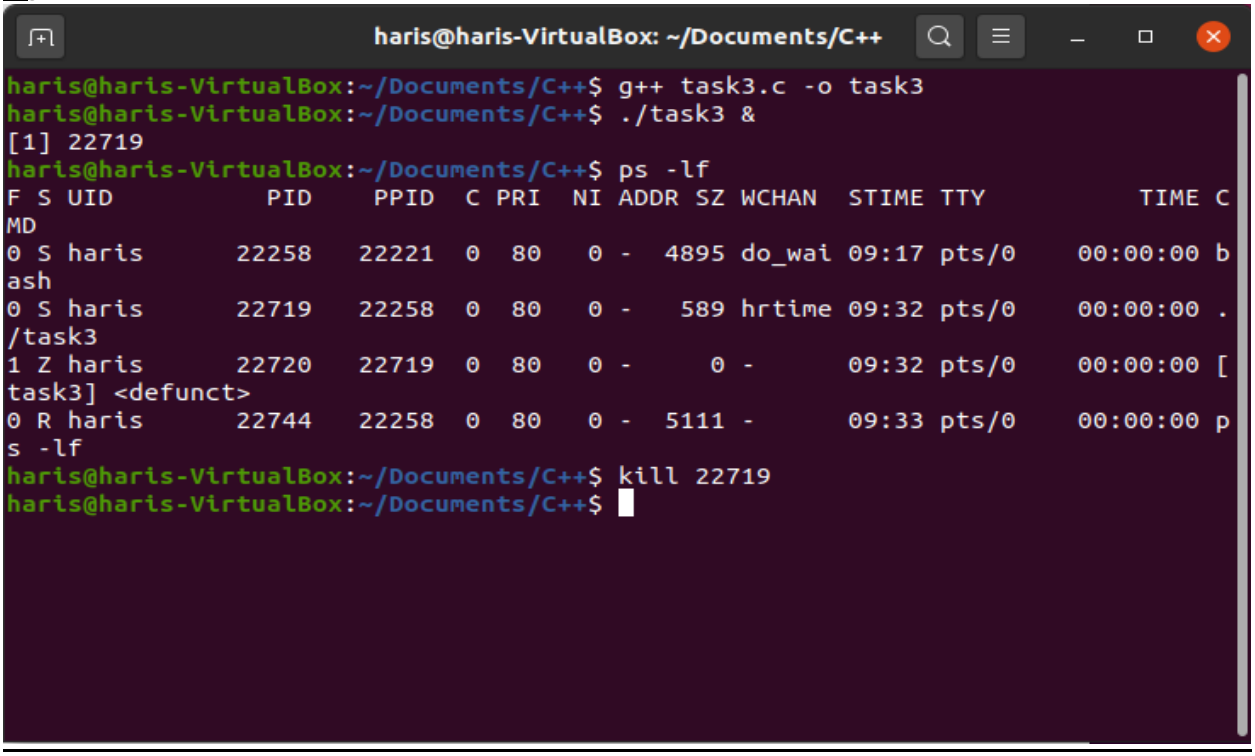
2) The Terminal Output is given below,



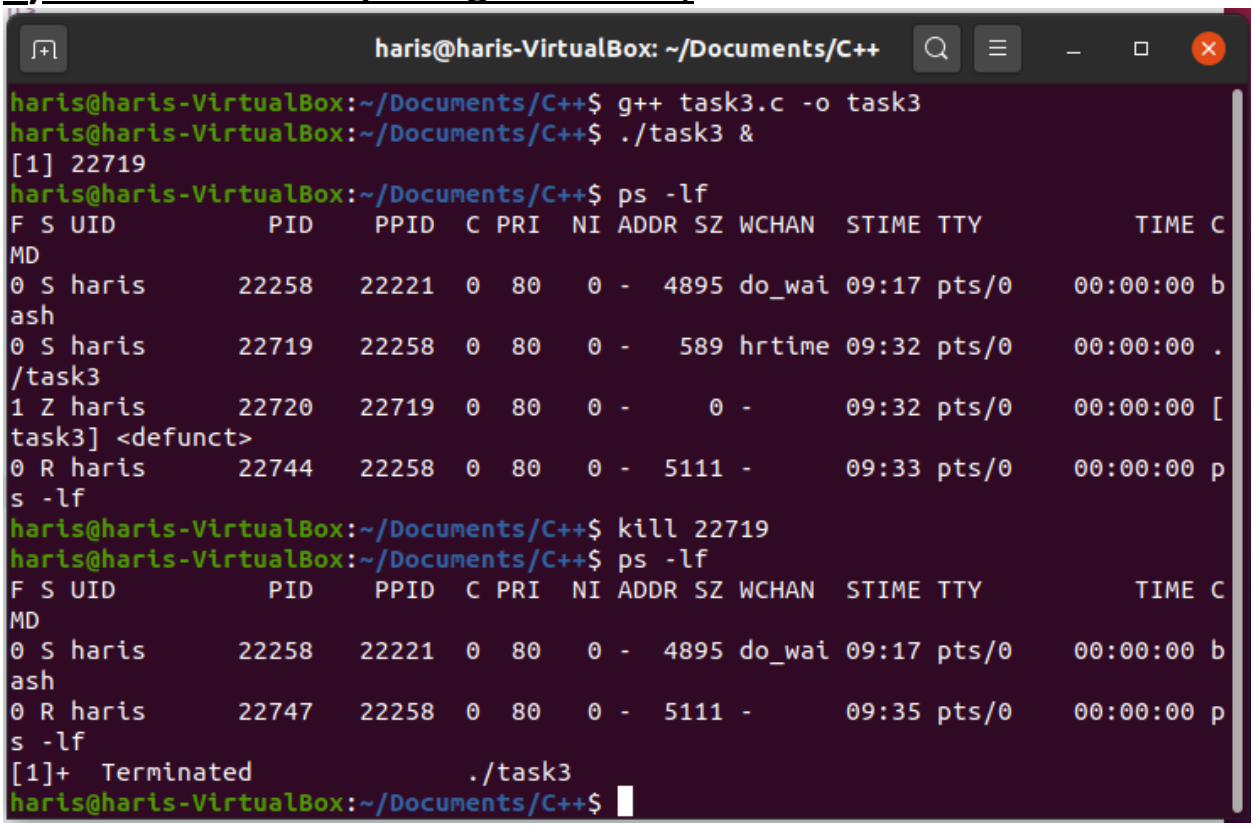
3) The Terminal Output is given below,



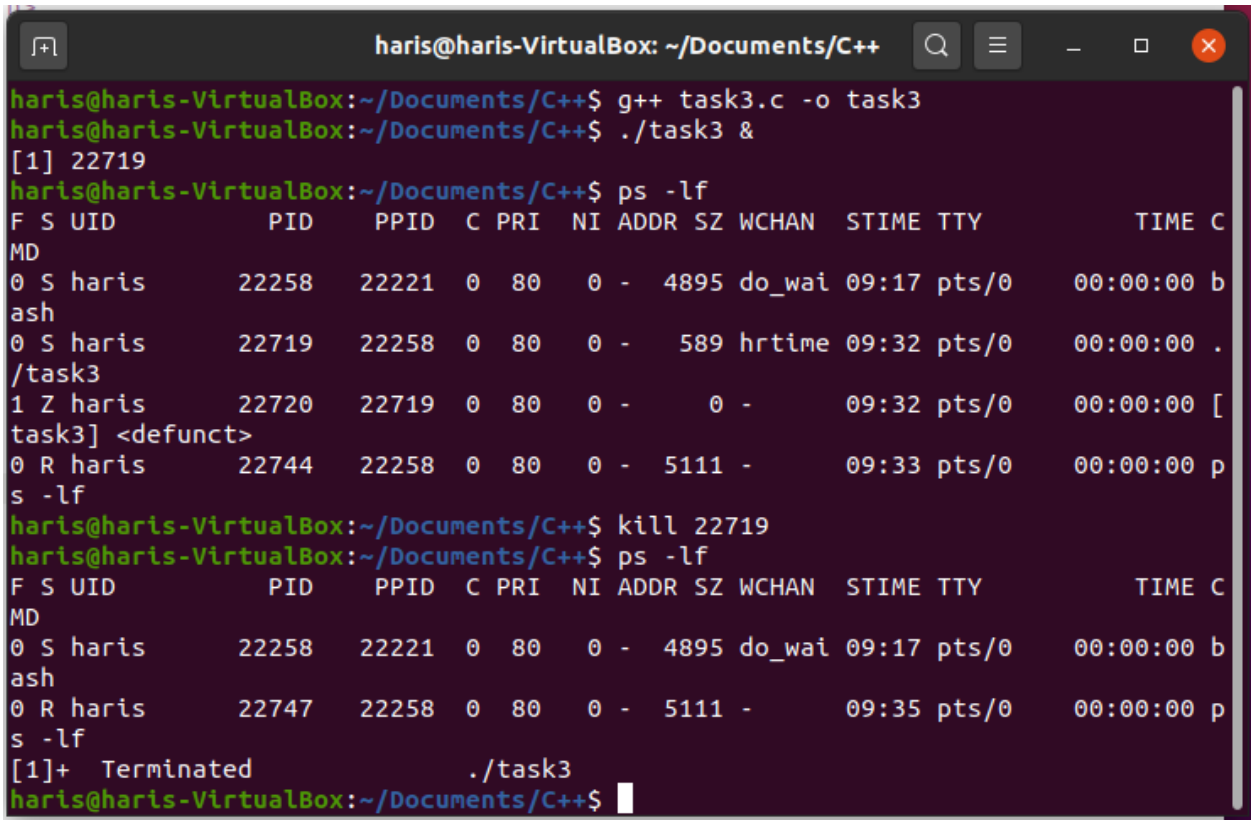
4) The Terminal Output is given below,



5) The Terminal Output is given below,



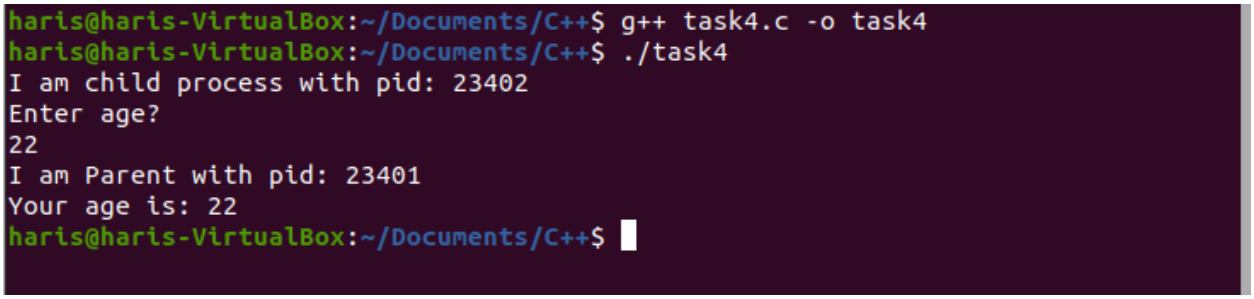
6) The Terminal Output is given below,



Question 4: Activity 4:

Write a C/C++ program in which a parent process creates a child process using a fork() system call. The child process takes your age as input and parent process prints the age.

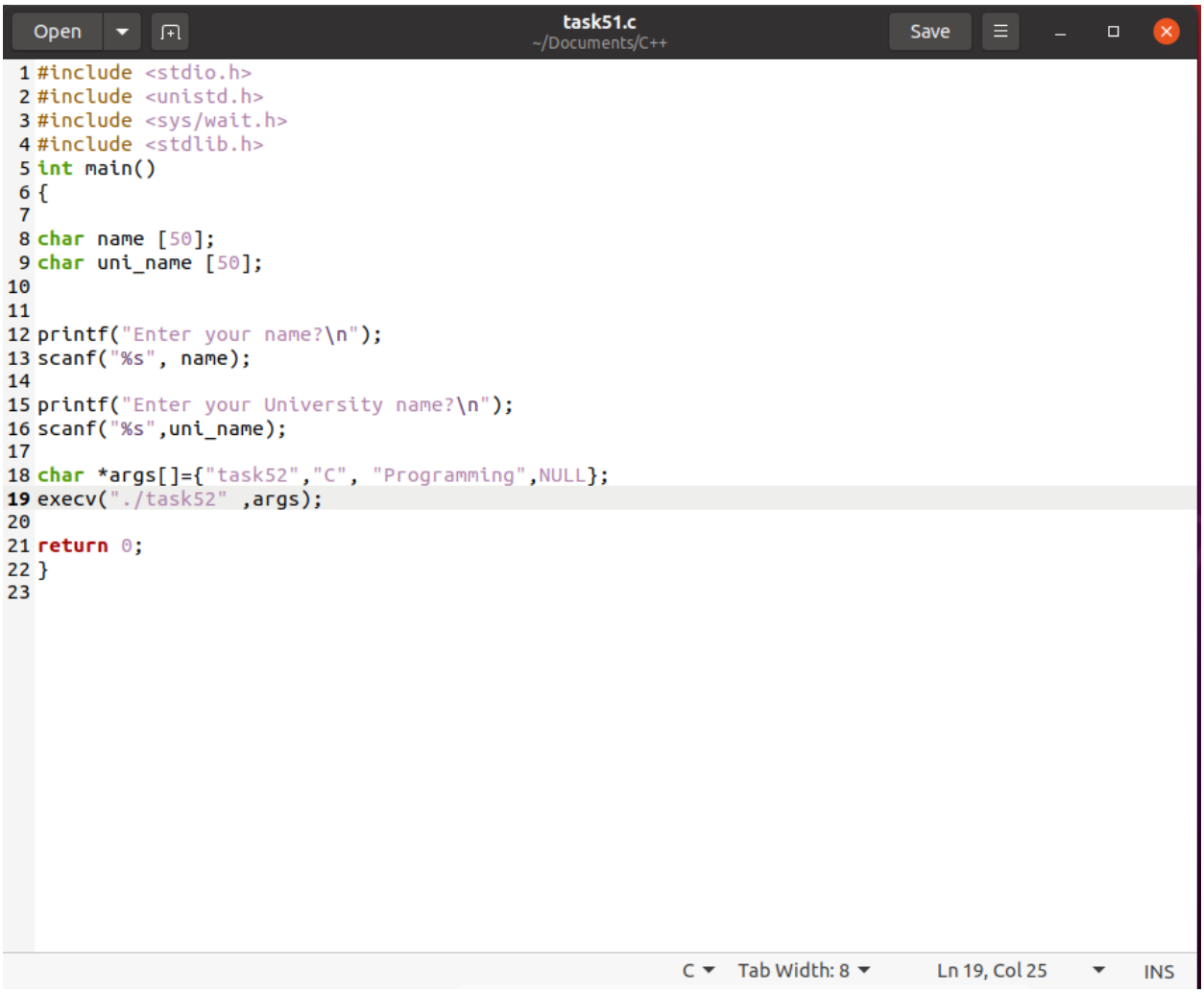
Solution:


haris@haris-VirtualBox:~/Documents/C++$ g++ task4.c -o task4
haris@haris-VirtualBox:~/Documents/C++$ ./task4
I am child process with pid: 23402
Enter age?
22
I am Parent with pid: 23401
Your age is: 22
haris@haris-VirtualBox:~/Documents/C++$

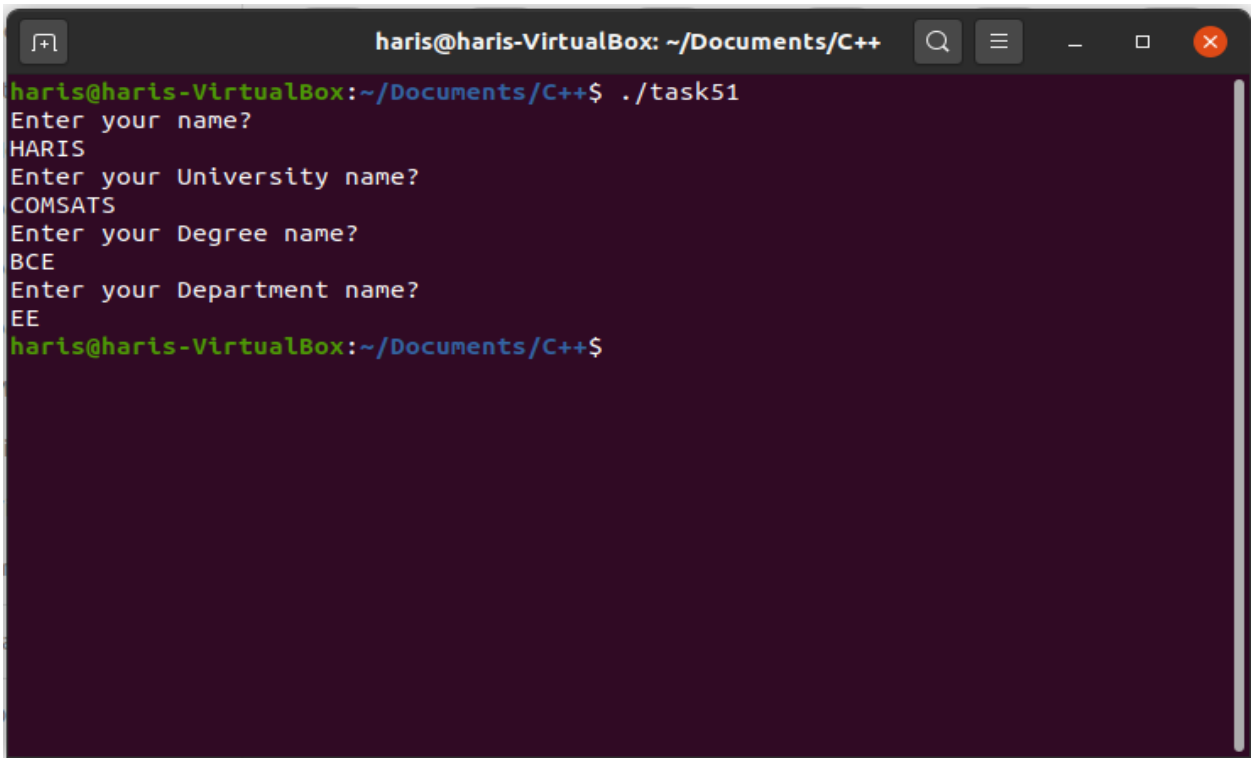
Question 5: Activity 5:

1. Write a C/C++ program that asks user to enter his name and his university name. Within the same program, execute another program that asks the user to enter his degree name and department name.
2. Hint: write 2 separate programs and execute using execv()

Solution:




Open
task52.c
~/Documents/C++
Save
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdlib.h>
int main()
{
char d_name [50];
char dept_name [50];
printf("Enter your Degree name?\n");
scanf("%s", d_name);
printf("Enter your Department name?\n");
scanf("%s",dept_name);
return 0;
}
C
Tab Width: 8
Ln 15, Col 32
INS


haris@haris-VirtualBox: ~/Documents/C++
haris@haris-VirtualBox:~/Documents/C++$ ./task51
Enter your name?
HARIS
Enter your University name?
COMSATS
Enter your Degree name?
BCE
Enter your Department name?
EE
haris@haris-VirtualBox:~/Documents/C++$

_____THE END_____