**Process Creation and Execution**

**LAB # 06**

****

**Spring 2021**

**CSE204L Operating System**

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Mian Ibad Ali Shah**

August 16, 2021

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Lab Objective(s):**

* This lab examines aspects of Process and multiprocessing.
* The primary objective of this lab is to implement the Process Management.

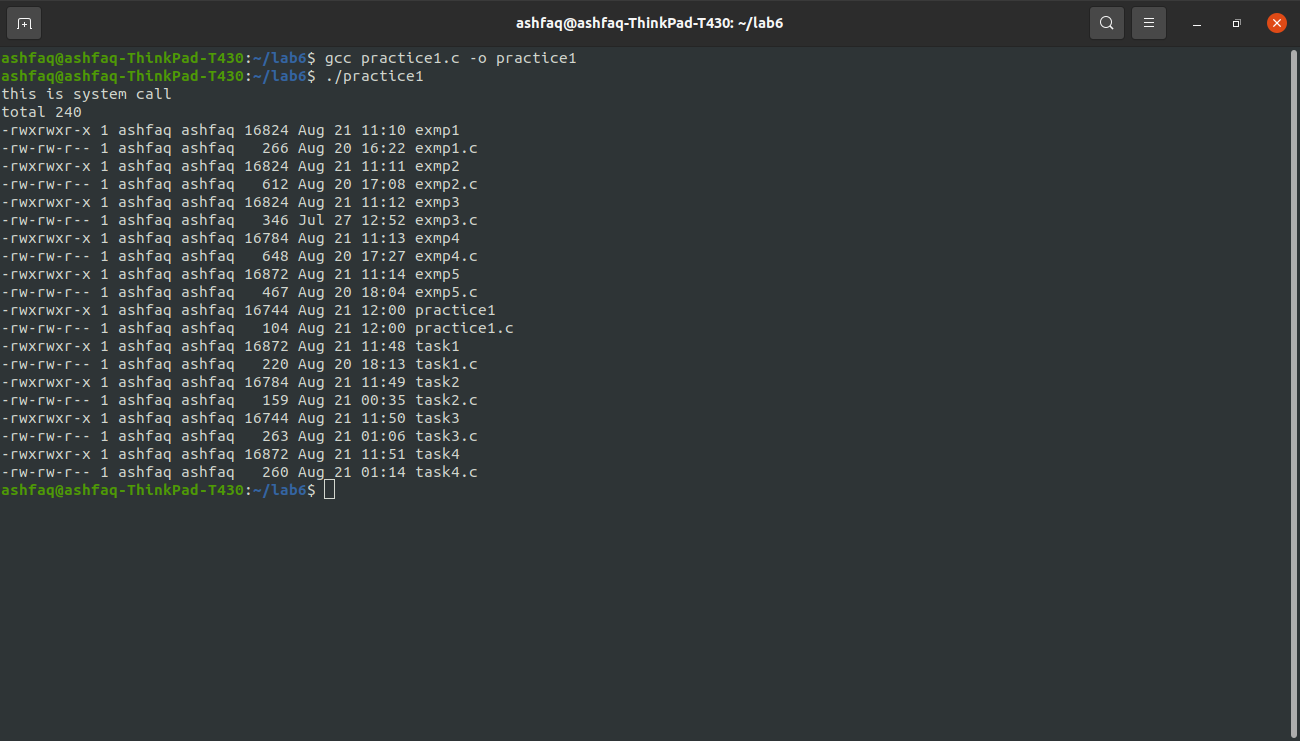
**Practice #01:**

* System call.

**Code:**



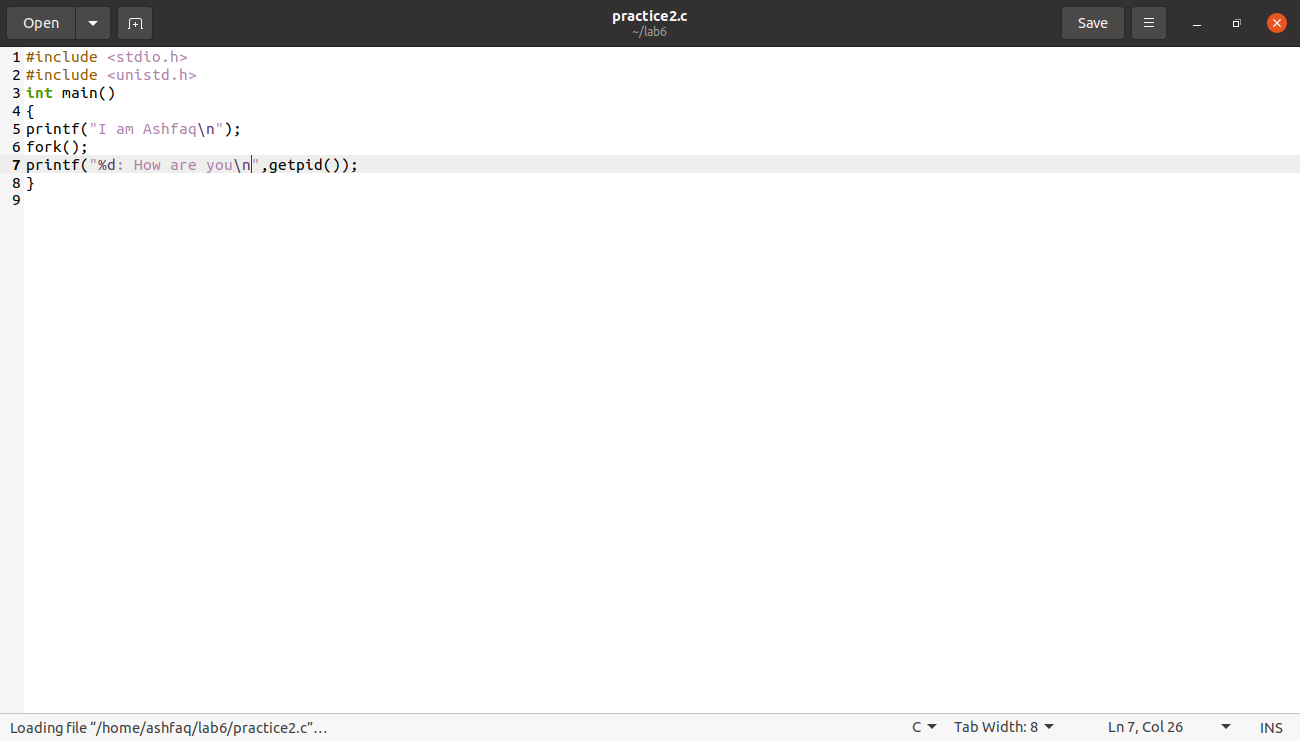
**Output:**



**Practice #02:**

* Using fork function.

**Code:**



**Output:**



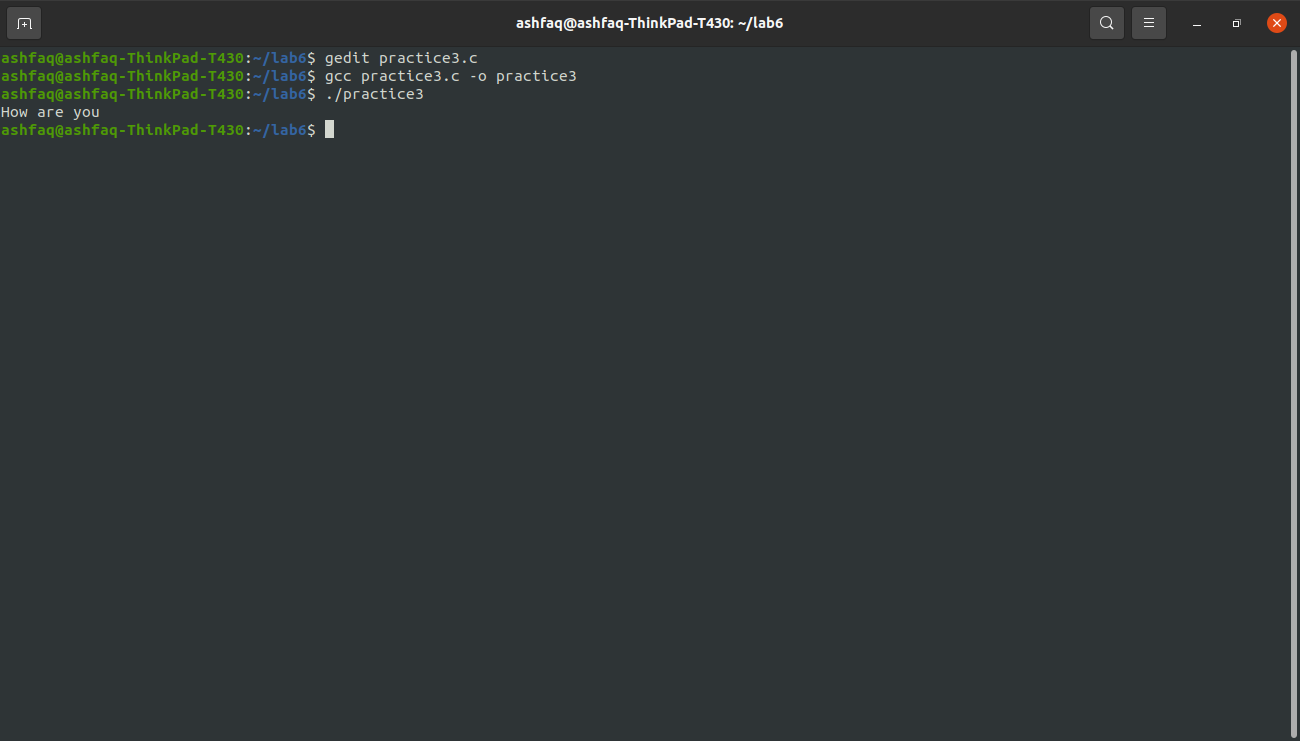
**Practice #03:**

* Fork function having no statement.

**Code:**



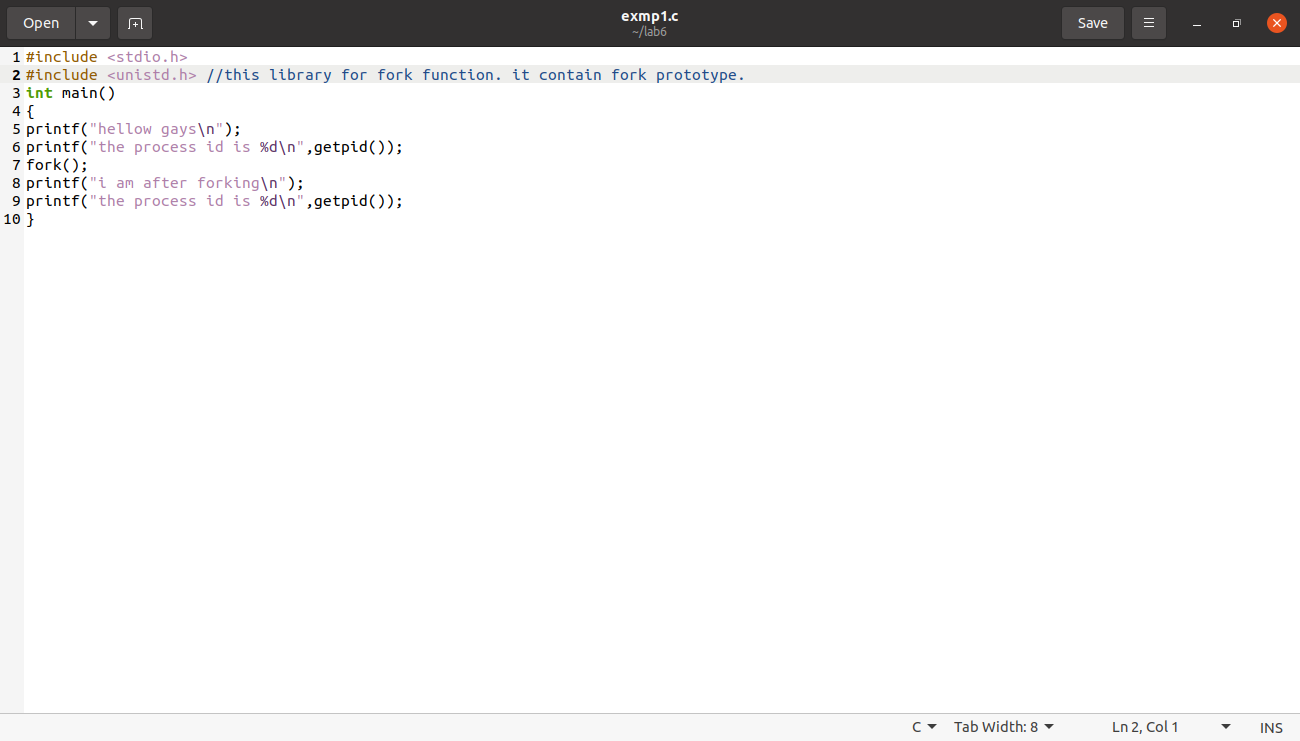
**Output:**



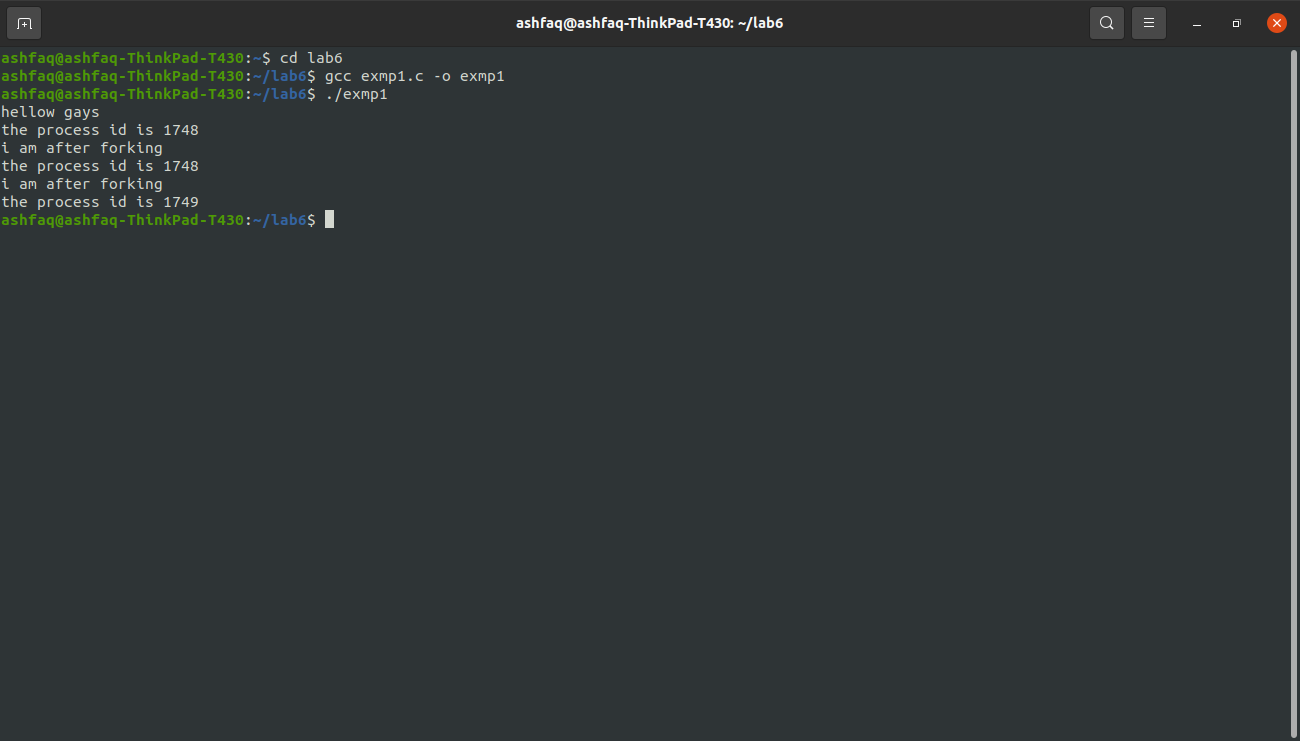
**Example # 01:**

* Implementation of fork function.

**Code:**



**Output:**



**Observations:**

By using fork function we created child process from parent process.

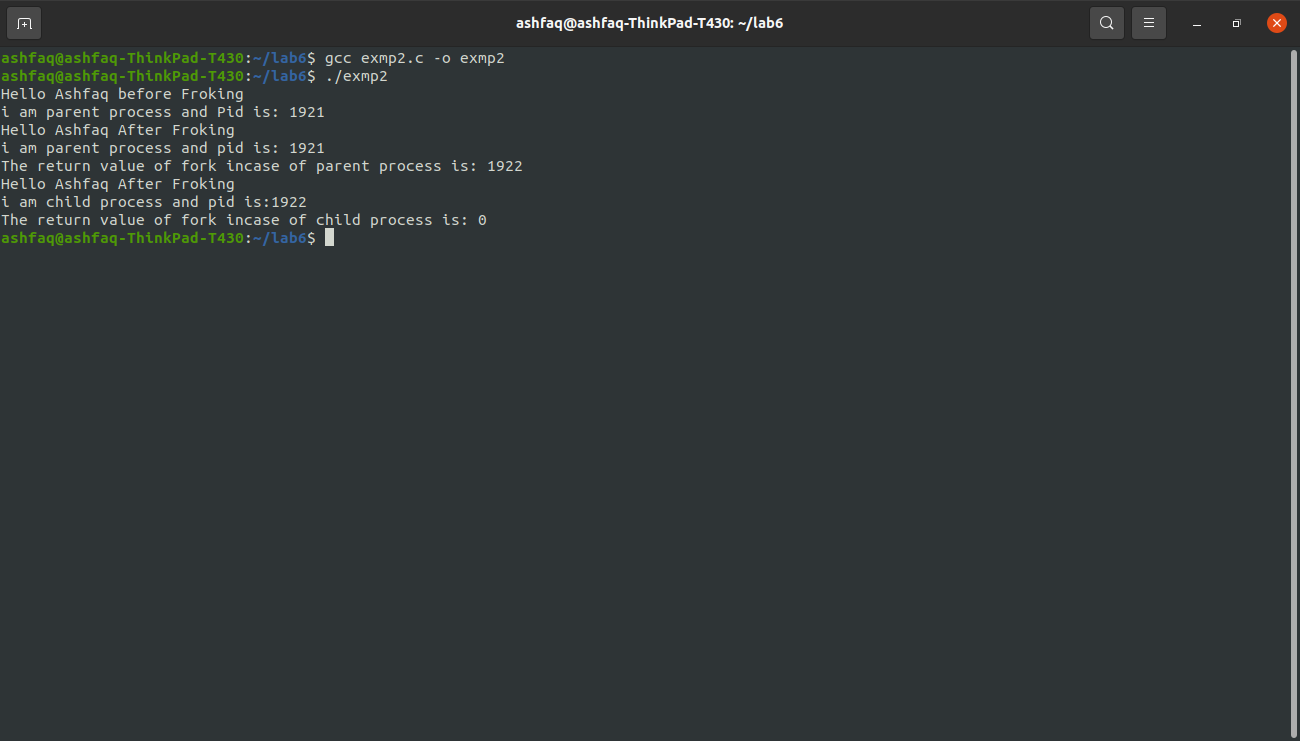
**Example # 02:**

* Each process prints a message identifying itself.

**Code:**



**Output:**



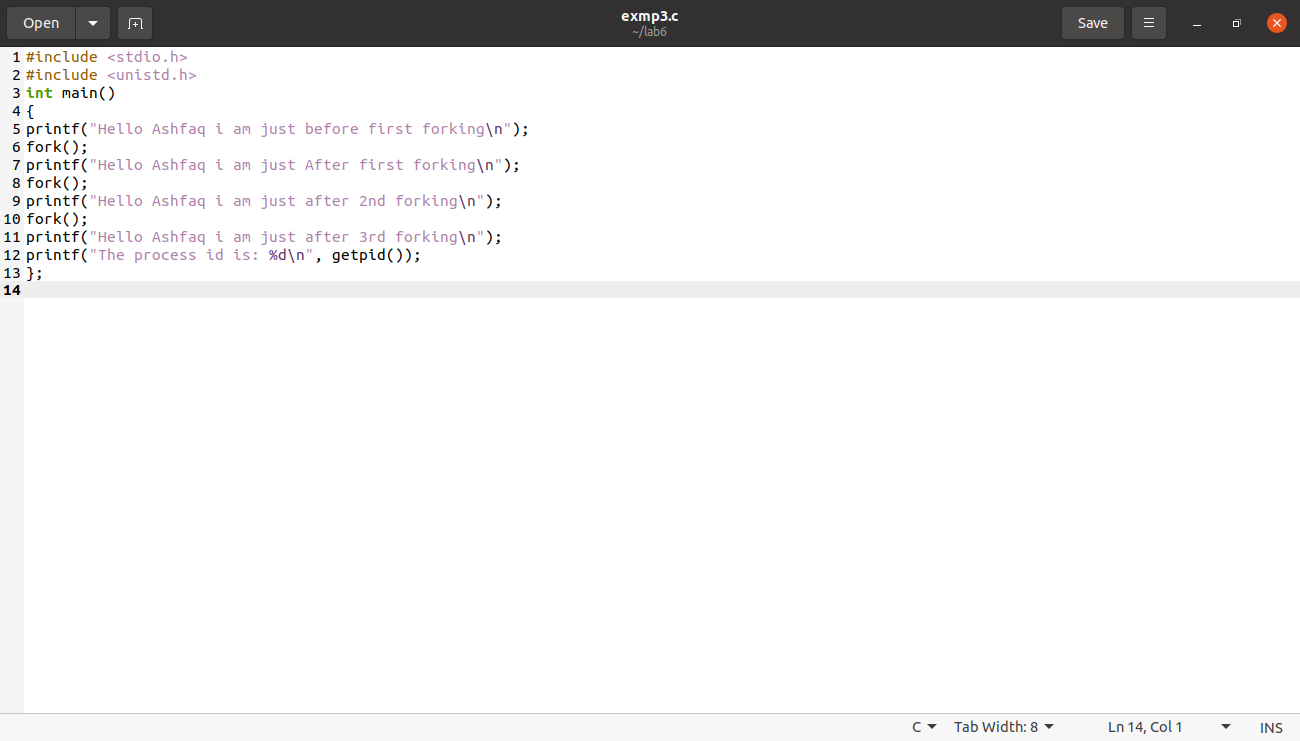
**Observations:**

Already explained in comment.

**Example # 03:**

* Multiple forks:

**Code:**

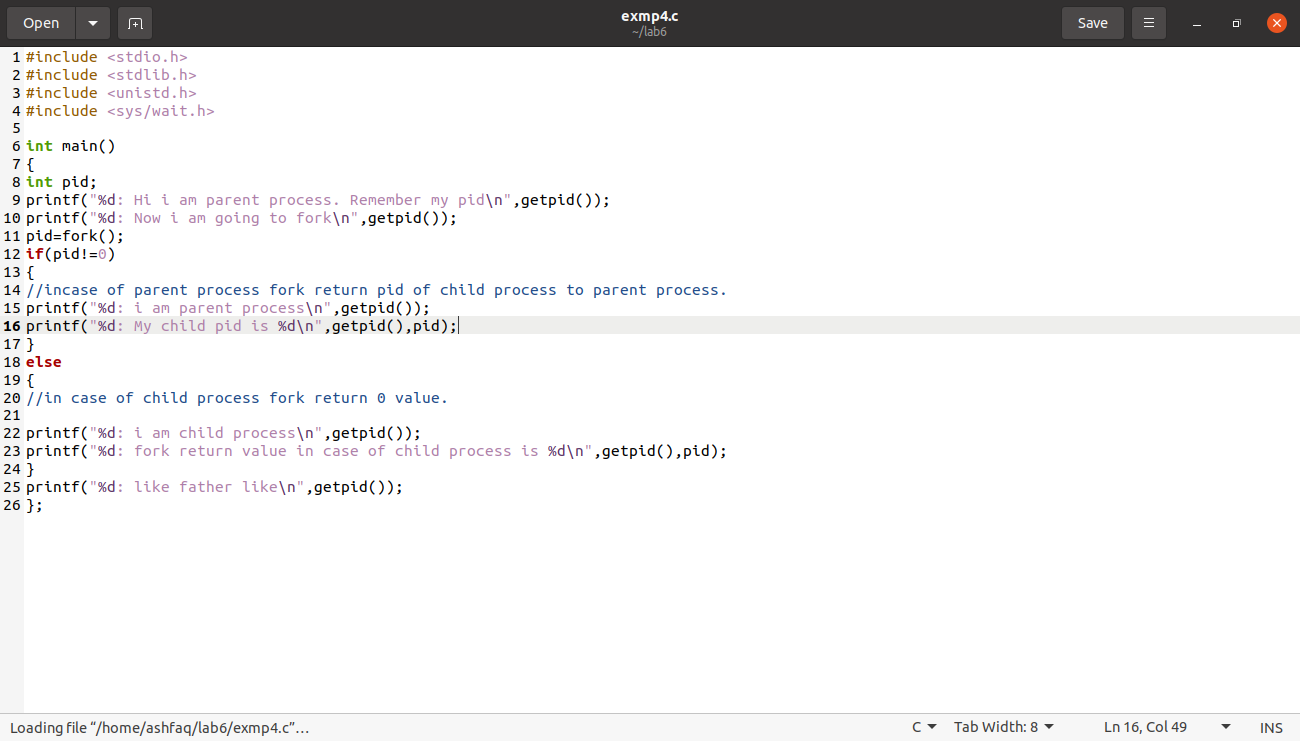


**Output:**



**Example # 04:**

**Code:**



**Output:**



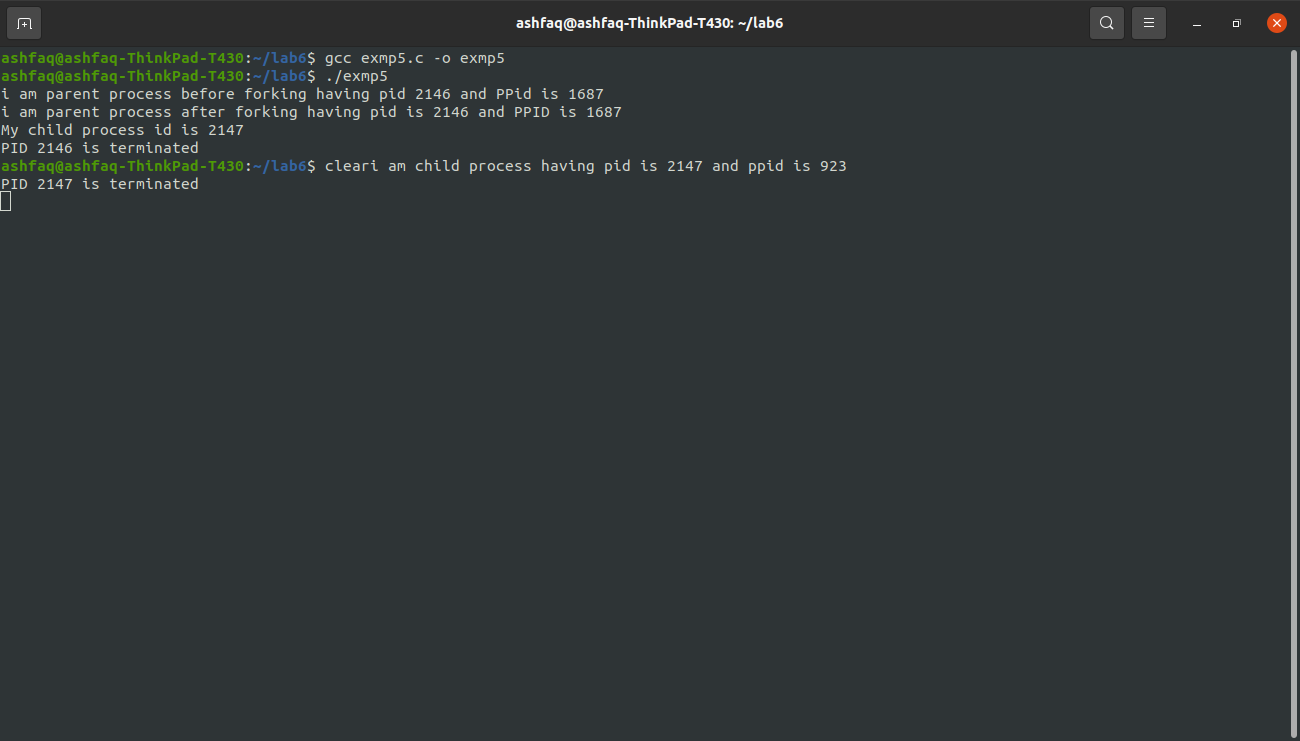
**Example # 05:**

* When a parent dies before its child.

**Code:**



**Output:**



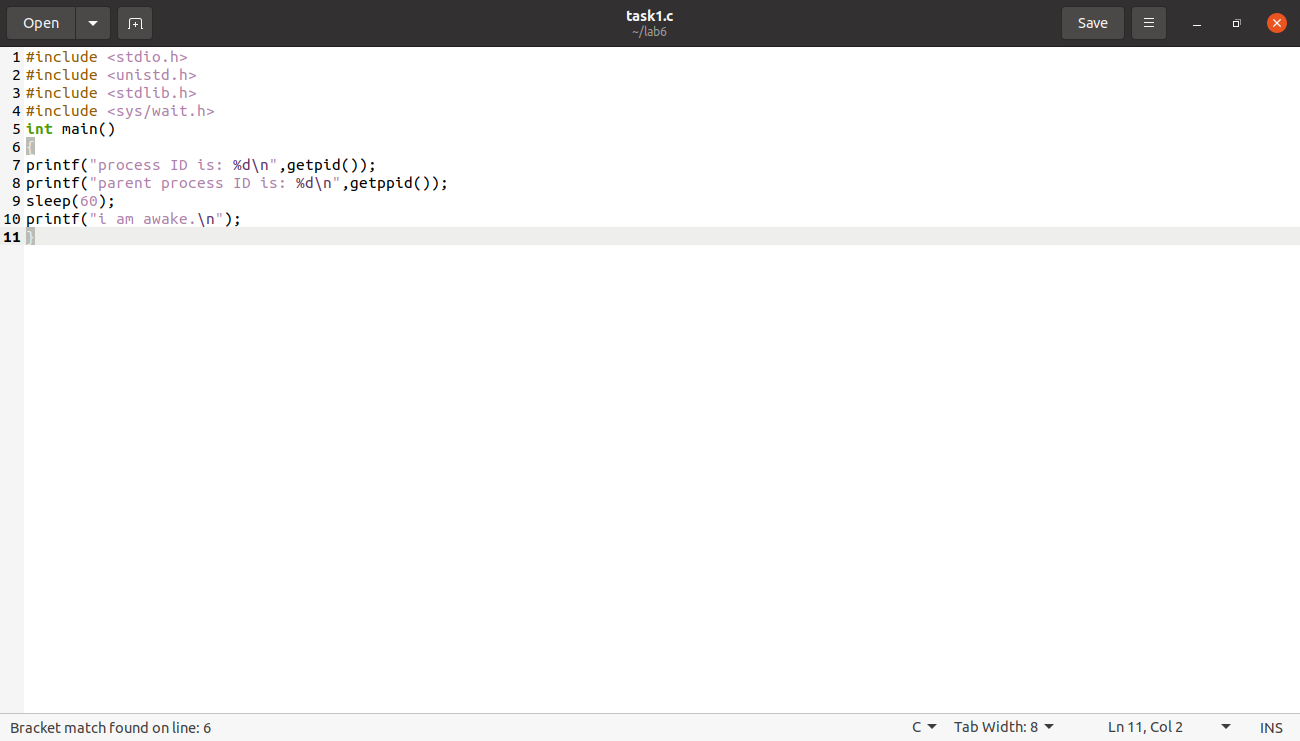
**Observations:**

As child process go to the sleep the parent process will execute and then die before child process execution. So the parent process of child will be initialized to init process having PID 1. But here PID is 923, I tried execution many time but it did not give us PID 1.

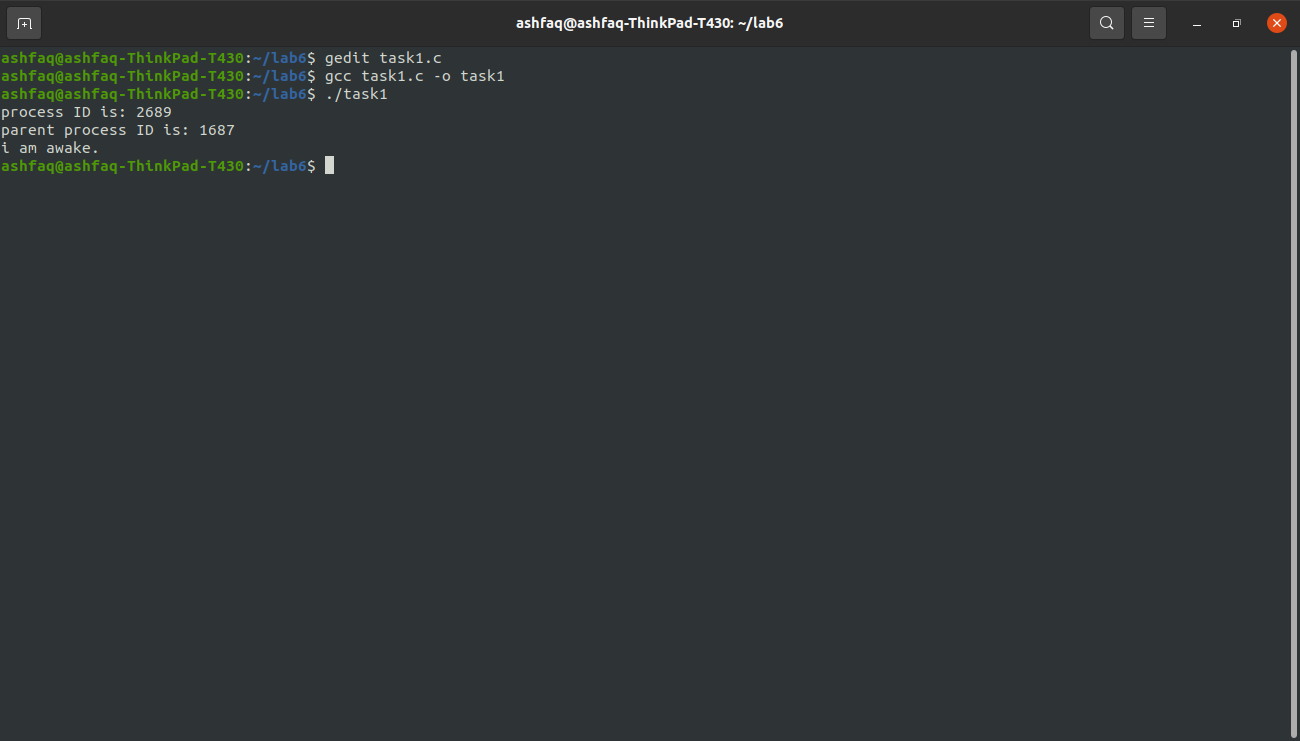
**Task # 01:**

* Run the following program twice. Both times as a background process, i.e., suffix it with an ampersand "&". Once both processes are running as background processes, view the *process table* using ps -l UNIX command. Observe the *process state*, *PID (process ID)* etc. Repeat this experiment to observe the changes, if any. Write your observation about the Process ID and state of the process

**Code:**



**Output:**

**

**Observations:**

After the execution of the program the child process and parent got their separate IDS as both processes are different from each other

**Sleep function**

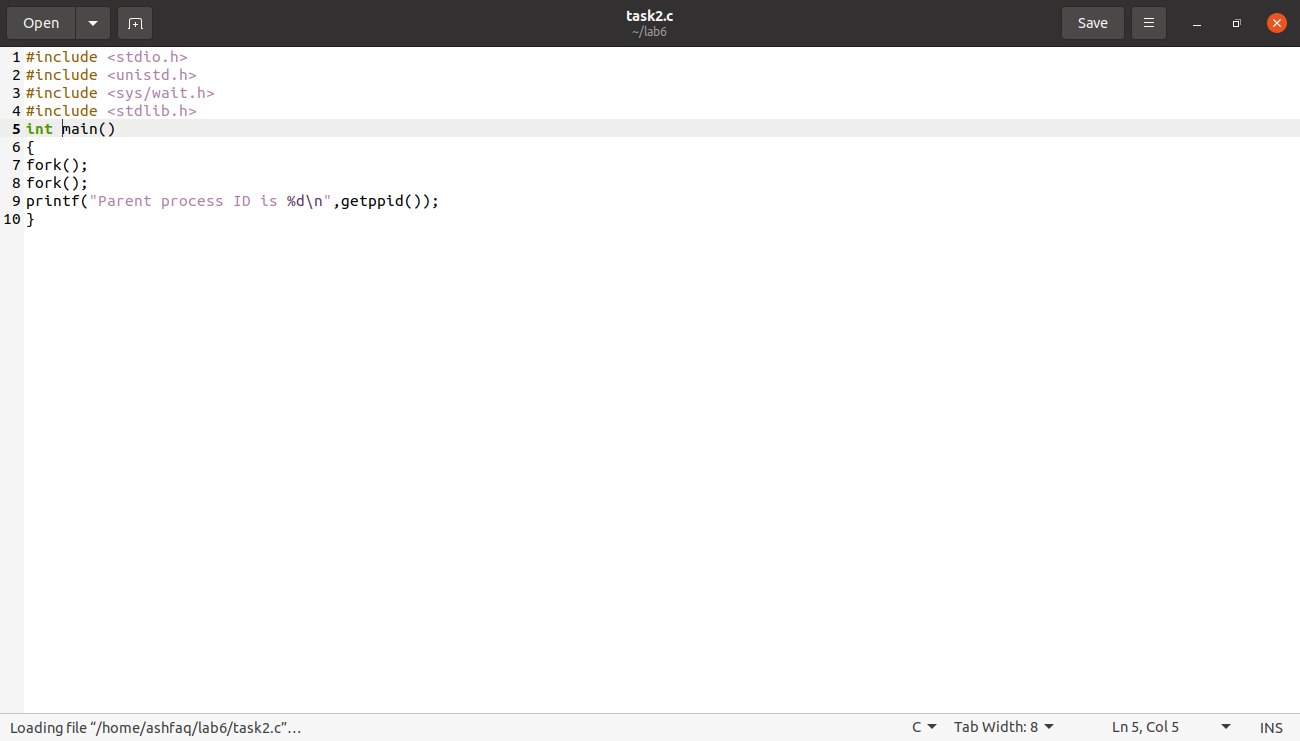
The function which is used suspend the execution of current process For specific no of second that we are passing in the parameter of the sleep function here we passed 60s as a parameter in sleep function

So this program is suspended for 60s after this time I will execute the last print statement i:e I am awake

**Task # 02:**

* Run the following program and observe the *number of times* and the *order* in which the print statement is executed. The fork( ) creates a child that is a duplicate of the parent process. The child process begins from the fork( ). All the statements after the call to fork ( ) are executed by the parent process and also by the child process.  Draw a family tree of processes and explain the results you observed.

**Code:**



**Output:**



**Observations:**

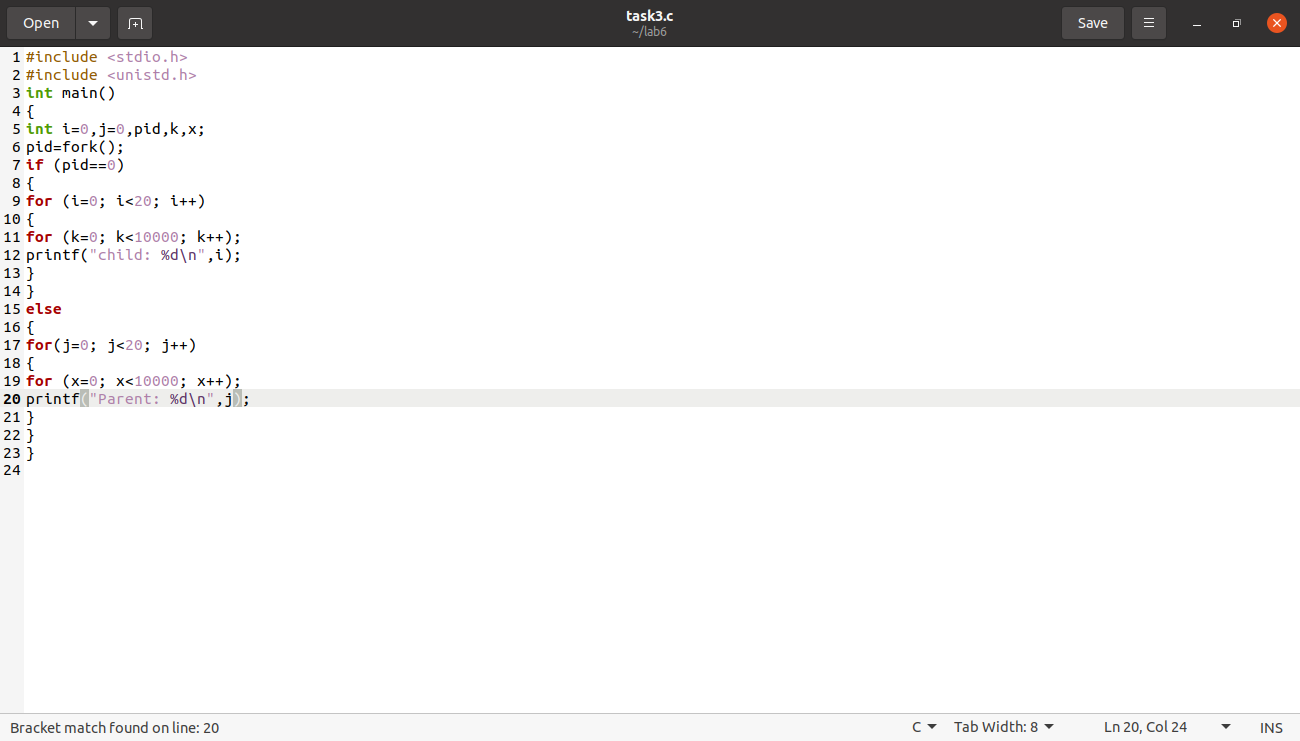
Parent process id is same except the first parent process id

So parent process repeats itself after first execution

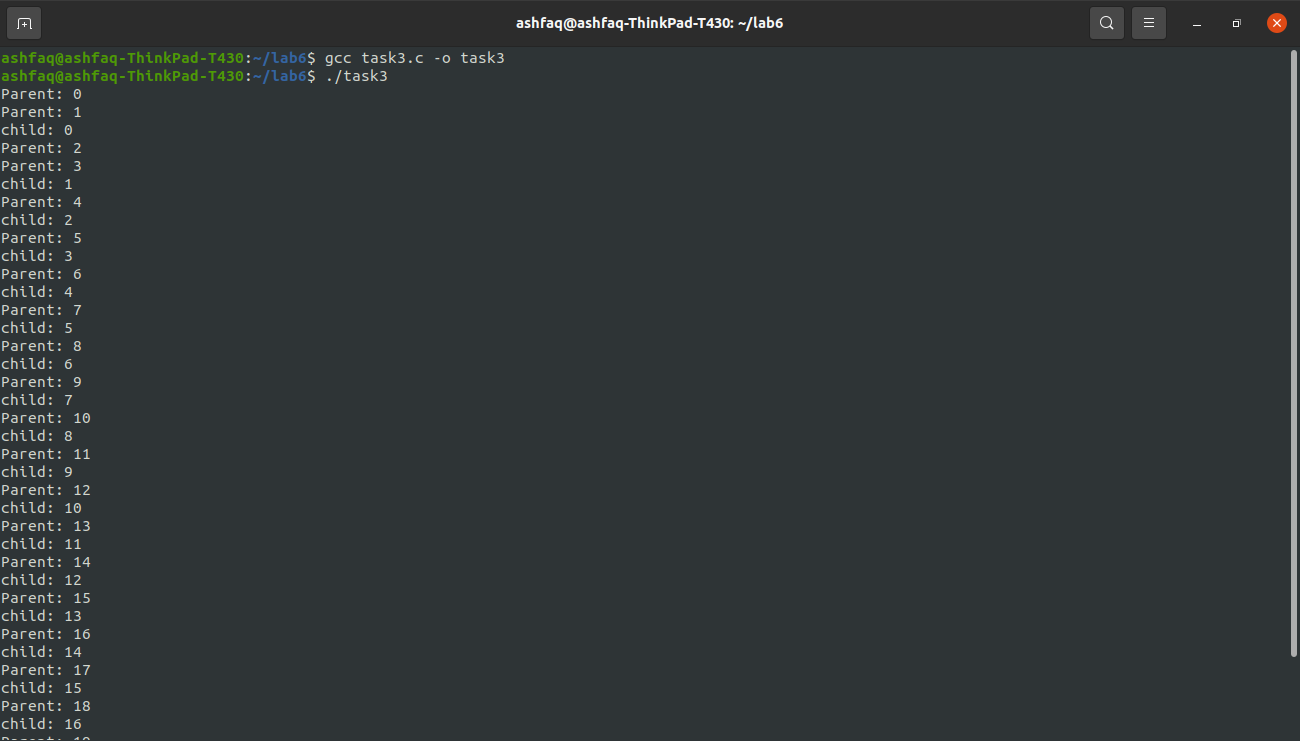
**Task # 03:**

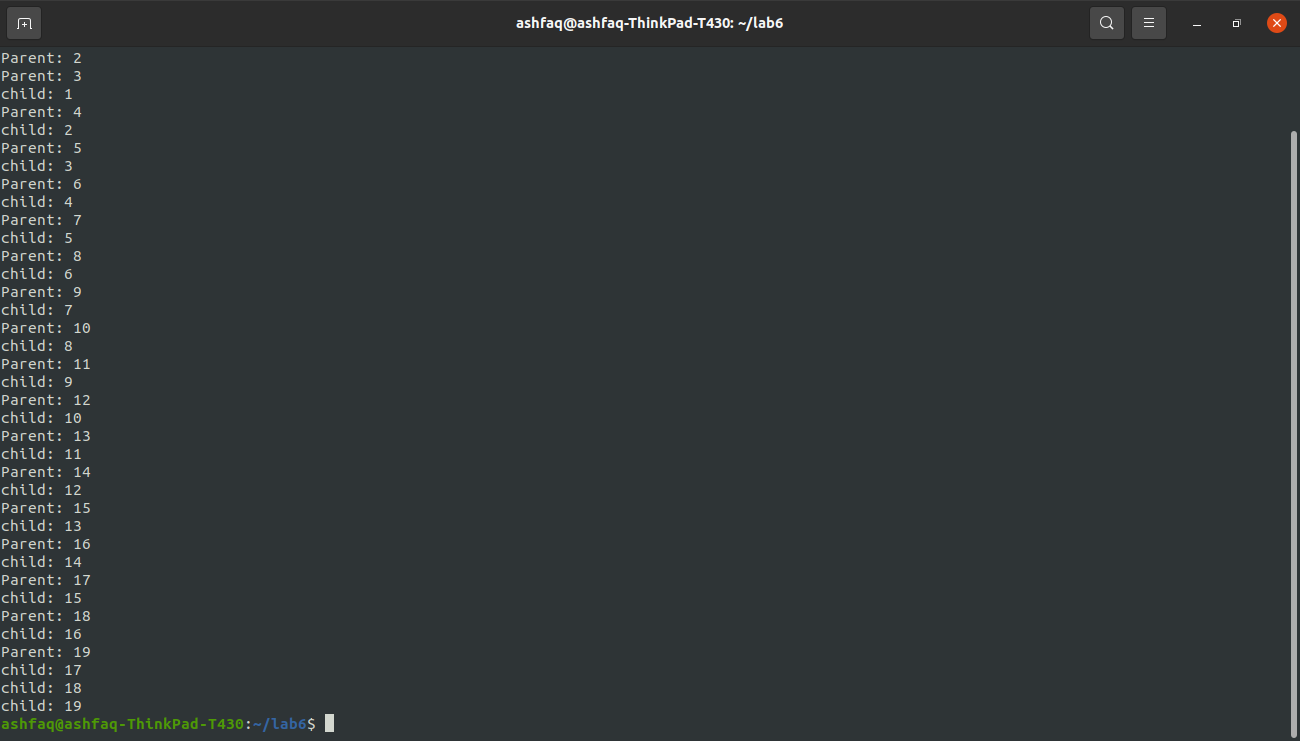
* Run the following program and observe the result of time slicing used by UNIX.

**Code:**



**Output:**

**

**

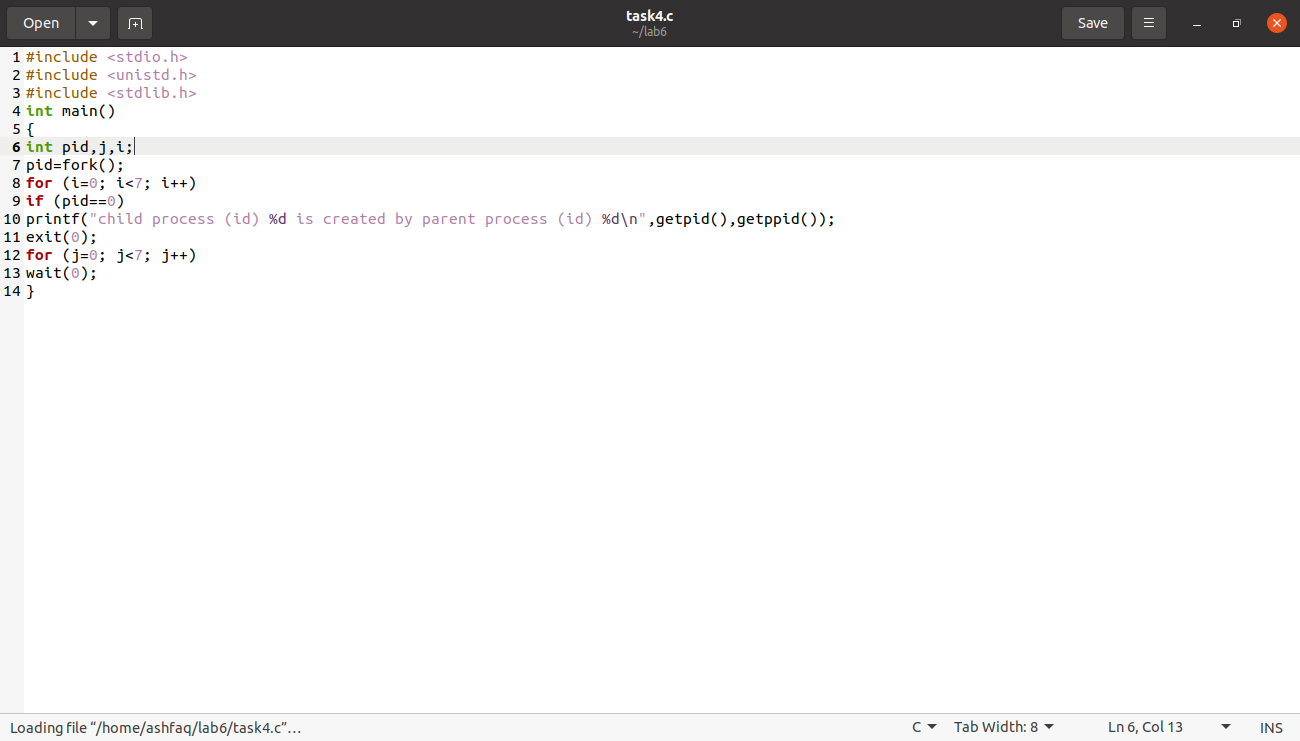
**Observations:**

Execution of program 20 times then first parent process is executes then their Childs process is executes

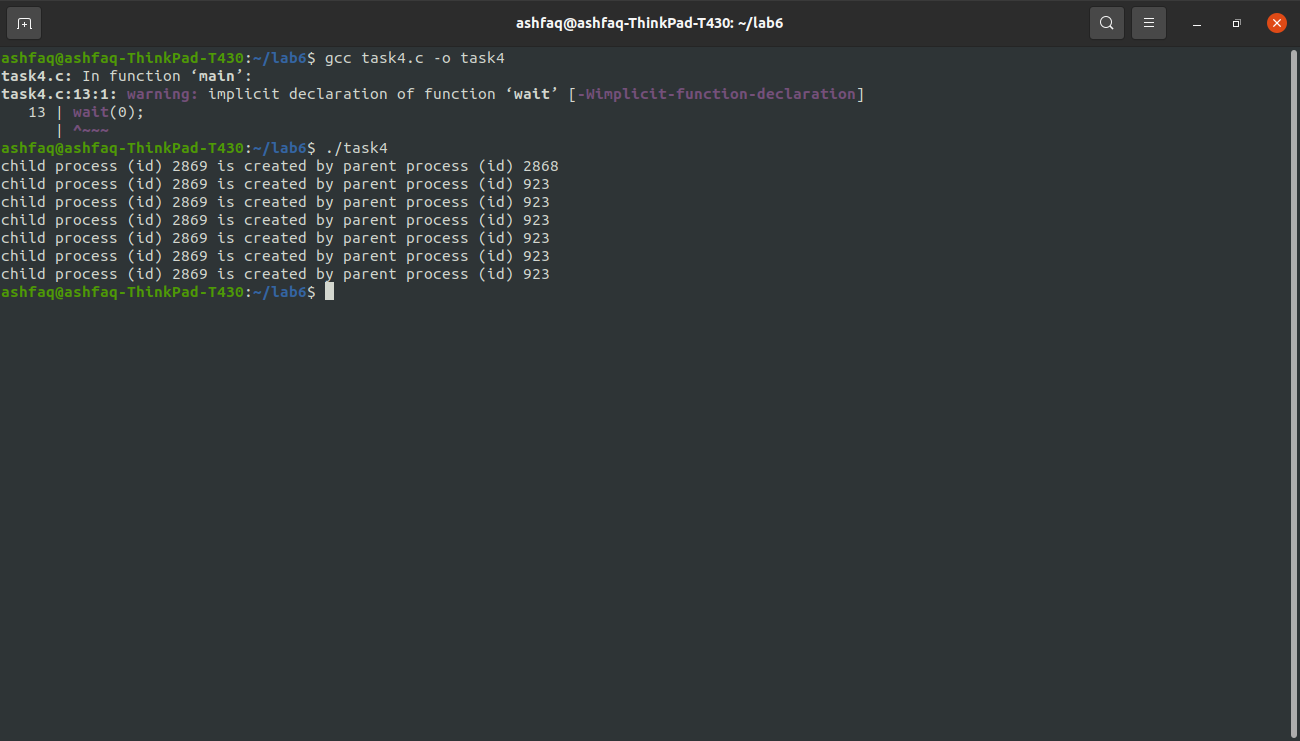
**Task # 04:**

* Create process fan as shown in figure 1 (a) and fill the figure 1 (a) with actual IDs.

**Code:**



**Output:**

**

**Observations:**

* Fan filled with actual PIDs.

THE END