

Addis Ababa University

Addis Ababa Institute of Technology

School of Information Technology and Engineering

*Lab Report*

**Course Name**: System Programming

**Course Code**: ITSE-3133

**Name**: Aymen Mohammednur

**Id**: ATR/5985/11

**Section**: Software 02

**Instructor Name**: Ms. Arisema Mezgebe

**Submission Date**: Sunday June 13/2021

**Table of Contents**

[**Part I: Process Creation and Process Id** 1](#_Toc74427851)

[*Practical 1* 1](#_Toc74427852)

[*Questions* 1](#_Toc74427853)

[*Practical 2* 2](#_Toc74427854)

[*Questions* 2](#_Toc74427855)

[*Practical 3* 3](#_Toc74427856)

[*Questions* 4](#_Toc74427857)

[**Part II: Process Termination and Waiting for Children** 5](#_Toc74427858)

[*Practical 4* 5](#_Toc74427859)

[*Questions* 5](#_Toc74427860)

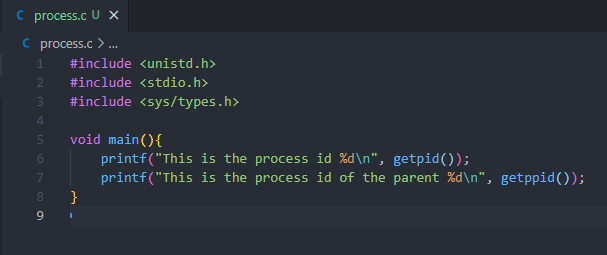
[**Part III: Command Line Arguments** 7](#_Toc74427861)

[**Part IV: Running other programs using execv system call** 8](#_Toc74427862)

[*Questions* 8](#_Toc74427863)

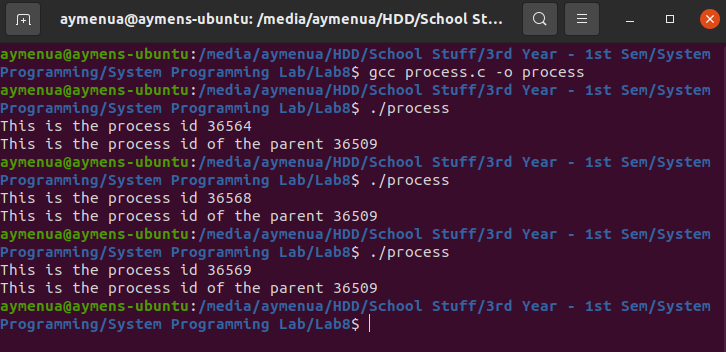
# **Part I: Process Creation and Process Id**

## *Practical 1*



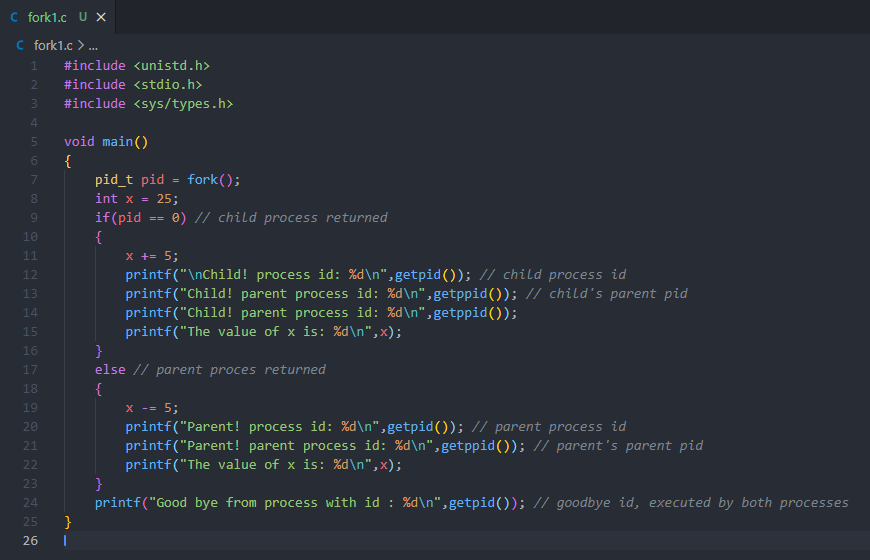
## *Questions*

1. Compile and run the above program called process.c. What is displayed (printed) by the program?



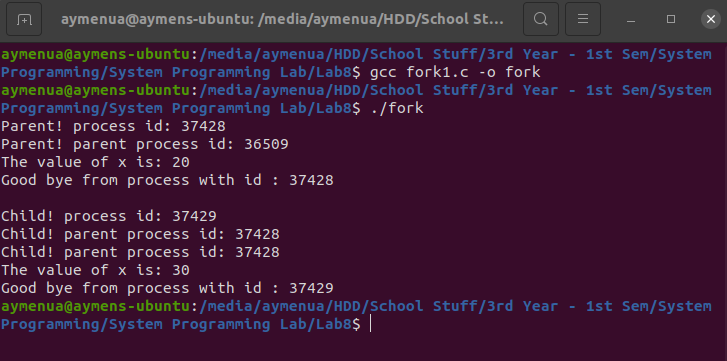
1. The process id on the first run is 36564.
2. The process id of the parent is 36509.
3. The parent process is the one which doesn’t change its value every time we run the program. It is the parent of the child calling process.

## *Practical 2*



## *Questions*

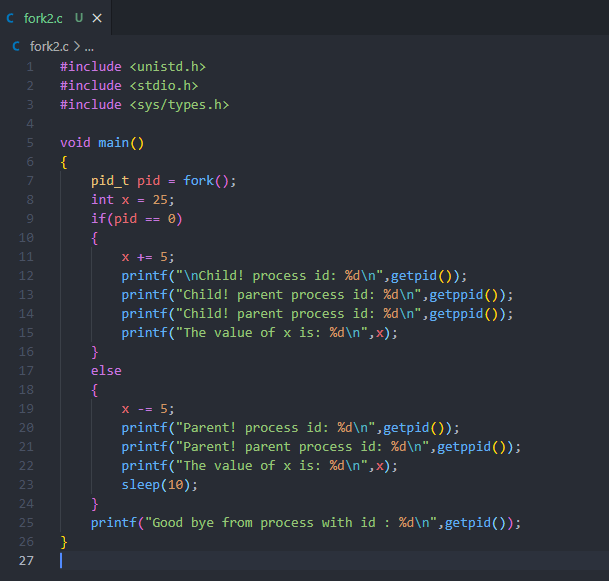
1. Compile and run the above program called fork1.c. What is displayed (printed) by the program?



1. Both processes executed their operations at the same time.
2. Process id of the parent is 37428 and process id of the child is 37429.
3. The last line (line 24) which is the printf statement is executed by both the child and parent process.

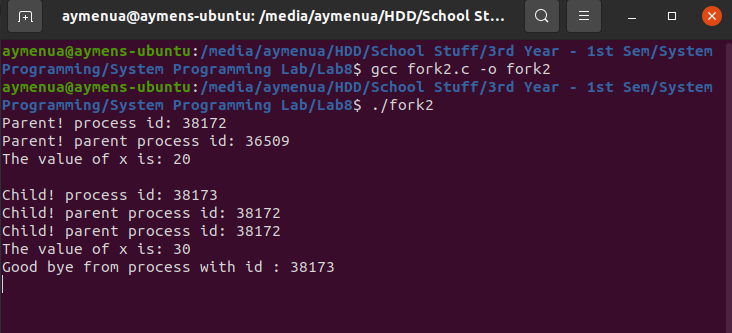
## *Practical 3*

The fork2.c code is shown below…



## *Questions*

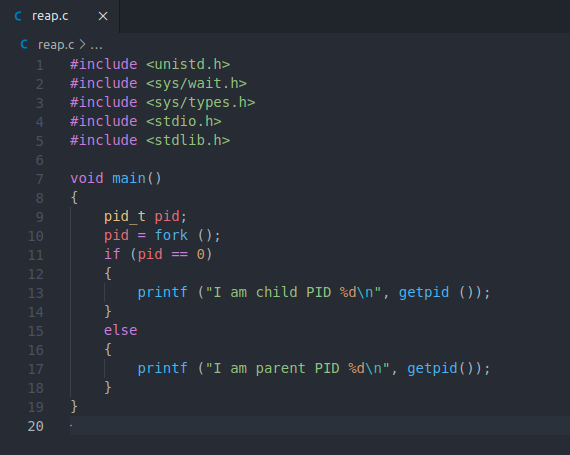
1. Compile and run the above program called fork2.c. What is displayed (printed) by the program?



1. Both processes executed at the same time but the final statement from the parent process is suspended for 10 seconds. So, it is executed after the child’s final statement is printed out.

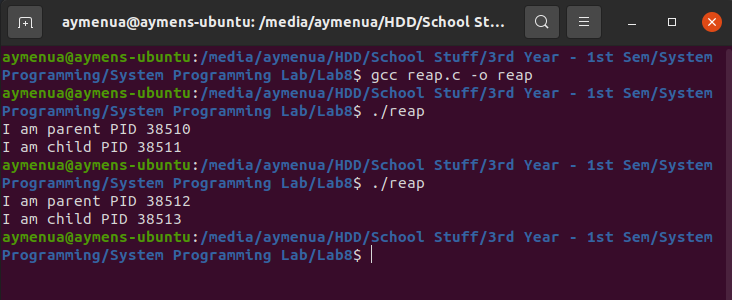
# **Part II: Process Termination and Waiting for Children**

## *Practical 4*

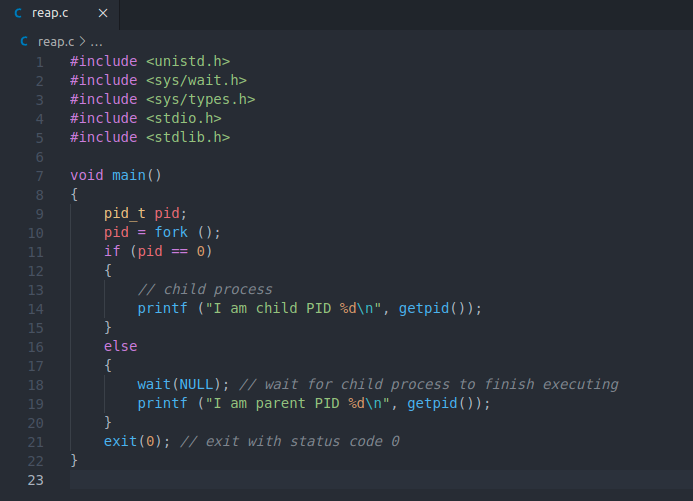


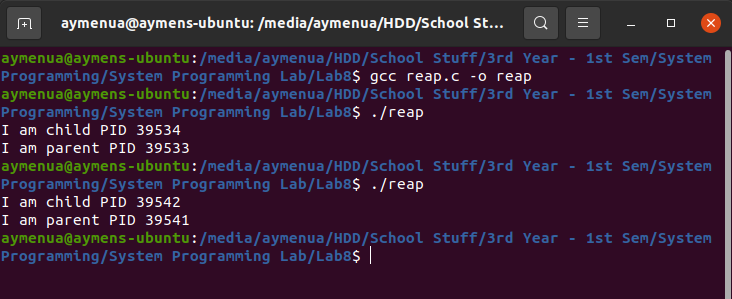
## *Questions*

1. Write the above program reap.c and execute it



1. Modify the above program (reap.c) by adding the correct form of wait() and exit(0), so that the child terminates successfully and the parent waits for the child to finish execution displaying “I am child …”

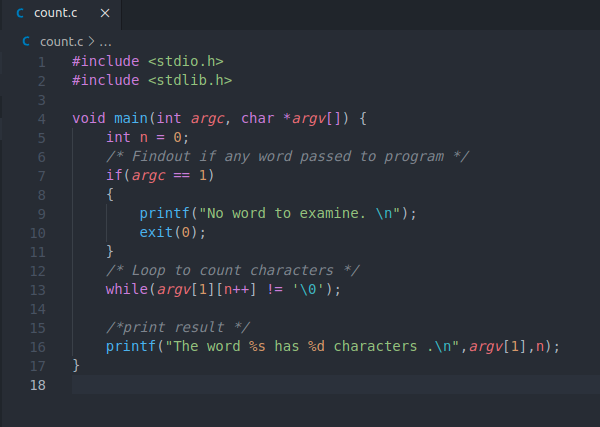




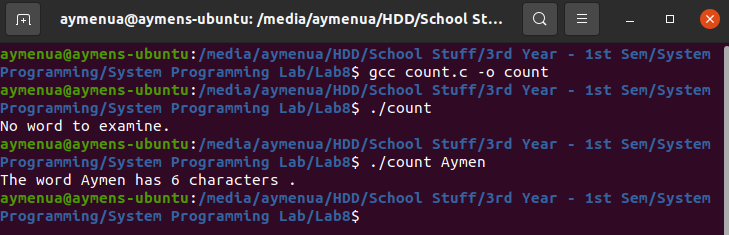
1. I added wait(NULL) in line number 18 (right after the else statement) to suspend it from executing until the child process was done. And exit(0) was added in line number 21 (after the else block) for both processes to execute.
2. The child process prints “I am...” first. This is because we have added the wait(NULL) command thus the parent will wait for the child to finish executing.

# **Part III: Command Line Arguments**

The count.c code is shown below…



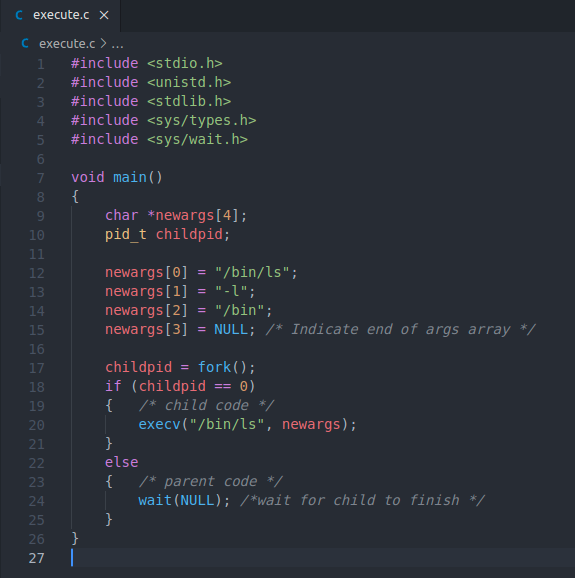
The above code has the following output...



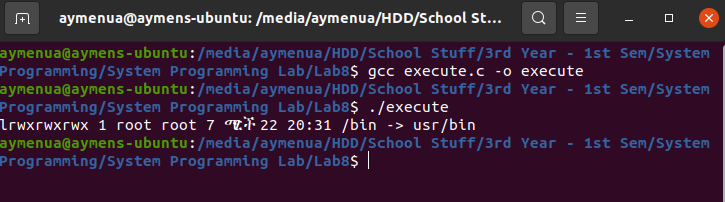
# **Part IV: Running other programs using execv system call**

## *Questions*

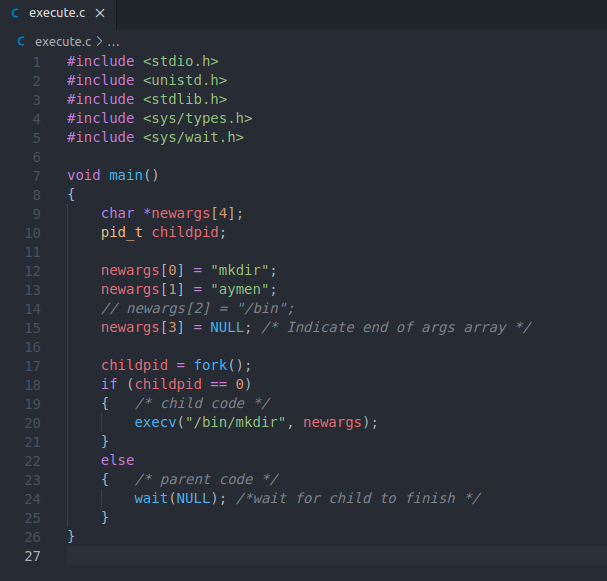
1. Write execute.c whose code is shown. Compile and run the program.



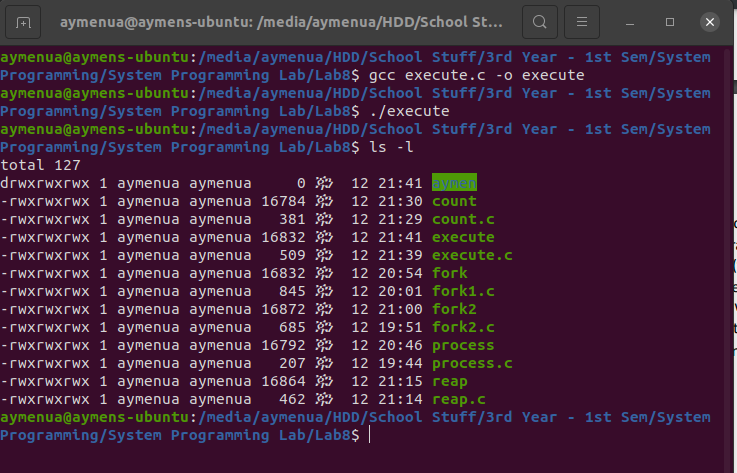
Output of the above code is shown below...



1. Modify above program (execute.c) so that it runs the command mkdir abc. mkdir is a program that creates a directory (folder) and the argument as a name for the directory.



1. Compile and run the program (the modified execute.c).



1. As we can see, the directory named “aymen” was successfully created.