Addis Ababa Institute of Technology (AaiT)

ITSC - System Programming

Laboratory 8: Process Management

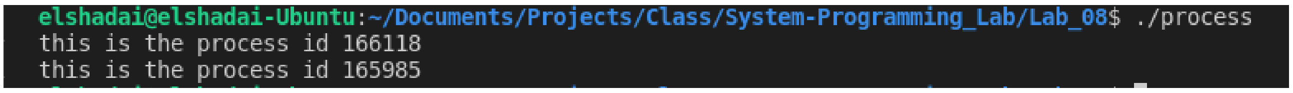
Name: Elshadai Kassu Tegegn
ID No: ATR/1831/11
Section 02

Submission Date: June 13, 2021

Part 1: Process Creation and Process ID

Practical 1: The following program demonstrates how to determine process id and parent process id. Create the C program called process.c which is shown below and answer the questions listed below.

1.



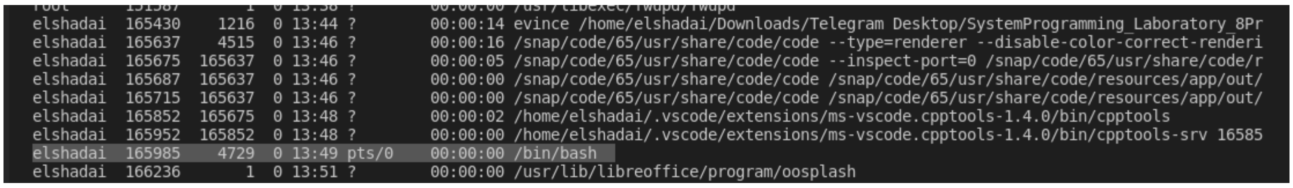
```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./process
this is the process id 166118
this is the process id 165985
```

The process id of the child & parent processes

2. 166118

3. 165985

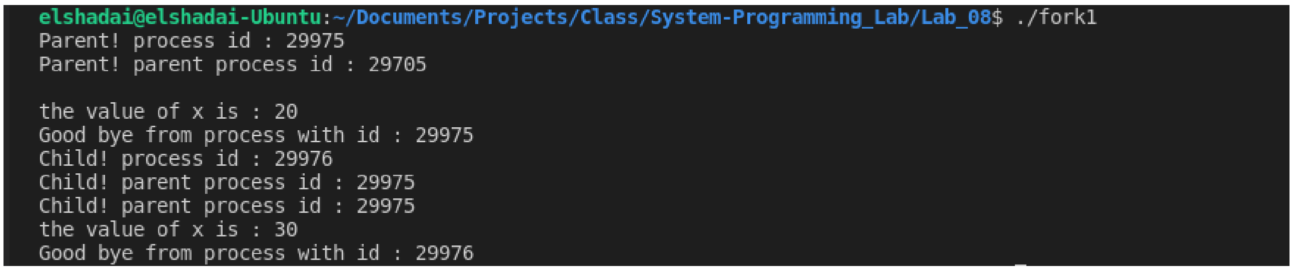
4.



```
root      151587       1  0 13:38 ?        00:00:00 /usr/libexec/fwupd/fwupd
elshadai  165430    1216  0 13:44 ?        00:00:14 evince /home/elshadai/Downloads/Telegram Desktop/SystemProgramming_Laboratory_8Pr
elshadai  165637    4515  0 13:46 ?        00:00:16 /snap/code/65/usr/share/code/code --type=renderer --disable-color-correct-renderi
elshadai  165675  165637  0 13:46 ?        00:00:05 /snap/code/65/usr/share/code/code --inspect-port=0 /snap/code/65/usr/share/code/r
elshadai  165687  165637  0 13:46 ?        00:00:00 /snap/code/65/usr/share/code/code /snap/code/65/usr/share/code/resources/app/out/
elshadai  165715  165637  0 13:46 ?        00:00:00 /snap/code/65/usr/share/code/code /snap/code/65/usr/share/code/resources/app/out/
elshadai  165852  165675  0 13:48 ?        00:00:02 /home/elshadai/.vscode/extensions/ms-vscode.cpptools-1.4.0/bin/cpptools
elshadai  165952  165852  0 13:48 ?        00:00:00 /home/elshadai/.vscode/extensions/ms-vscode.cpptools-1.4.0/bin/cpptools-srv 16585
elshadai  165985    4729  0 13:49 pts/0    00:00:00 /bin/bash
elshadai  166236       1  0 13:51 ?        00:00:00 /usr/lib/libreoffice/program/oosplash
```

Practical 2: The following program demonstrates how to create processes using fork. Create the C program called fork1.c which is shown below and answer the questions listed below.

1.



```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./fork1
Parent! process id : 29975
Parent! parent process id : 29705

the value of x is : 20
Good bye from process with id : 29975
Child! process id : 29976
Child! parent process id : 29975
Child! parent process id : 29975
the value of x is : 30
Good bye from process with id : 29976
```

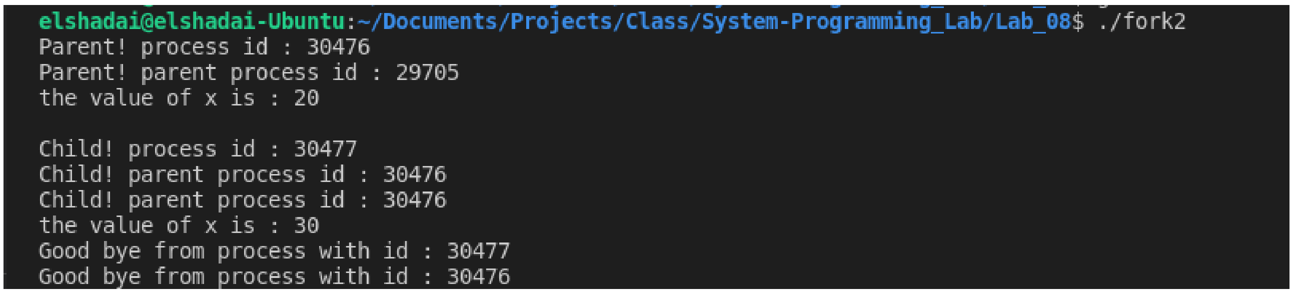
2. The Parent Process is executed first.

3. Parent Process => 29975 & Child Process => 29976

4. `printf("Good bye from process with id : %d\n",getpid());`

Practical 3: Modify the above program in practical 2 to make the parent process sleep for 10 seconds. Save the modified program as fork2.c and answer the questions listed below. You need to modify the parent process only (the code written inside the else statement).

1.



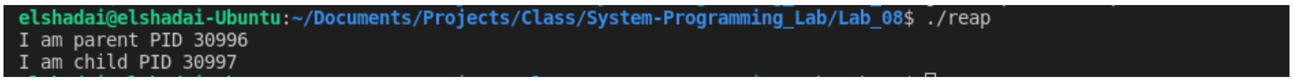
```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./fork2
Parent! process id : 30476
Parent! parent process id : 29705
the value of x is : 20

Child! process id : 30477
Child! parent process id : 30476
Child! parent process id : 30476
the value of x is : 30
Good bye from process with id : 30477
Good bye from process with id : 30476
```

2. The child process finished executing first. (Which has a process id of 30477)

PART II: PROCESS TERMINATION AND WAITING FOR CHILDREN

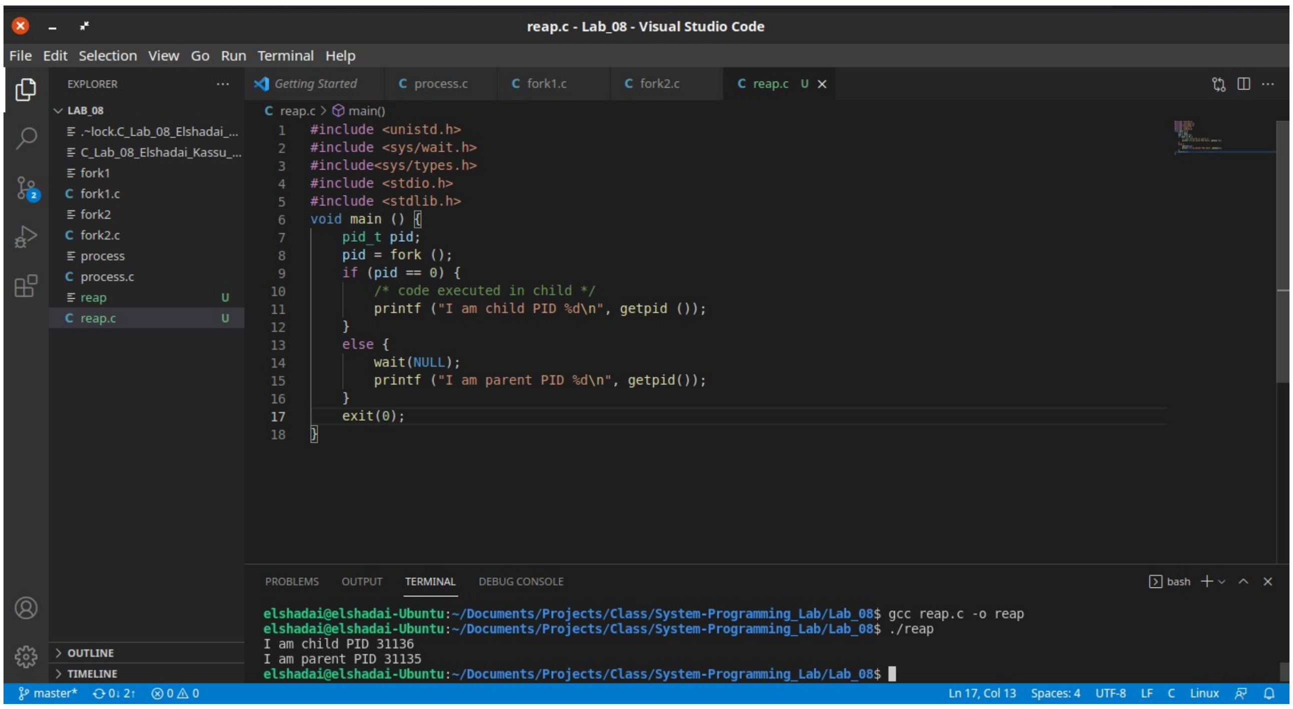
Practical 4: Write the program reap.c shown below and answer the questions listed below.

1.



```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./reap
I am parent PID 30996
I am child PID 30997
```

2.



reap.c - Lab_08 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB_08
.~lock.C_Lab_08_Elshadai_...
C_Lab_08_Elshadai_Kassu_...
fork1
fork1.c
fork2
fork2.c
process
process.c
reap
reap.c
Getting Started
process.c
fork1.c
fork2.c
reap.c
OUTLINE
TIMELINE
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
bash
master* 0↓ 2↑ 0 0
Ln 17, Col 13 Spaces: 4 UTF-8 LF C Linux

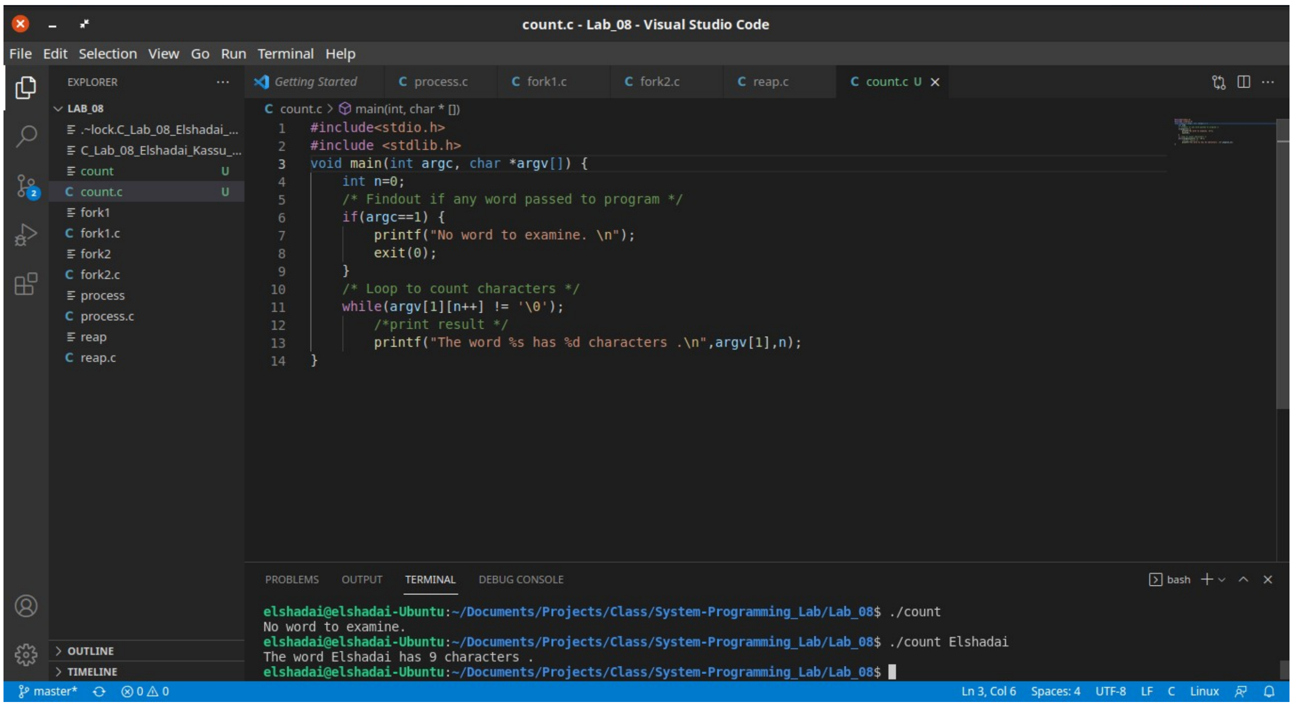
```c
#include <unistd.h>
#include <sys/wait.h>
#include<sys/types.h>
#include <stdio.h>
#include <stdlib.h>
void main () {
    pid_t pid;
    pid = fork ();
    if (pid == 0) {
        /* code executed in child */
        printf ("I am child PID %d\n", getpid ());
    }
    else {
        wait(NULL);
        printf ("I am parent PID %d\n", getpid());
    }
    exit(0);
}
```

```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ gcc reap.c -o reap
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./reap
I am child PID 31136
I am parent PID 31135
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$
```

3. The statement “wait(NULL);” is added before the parent’s print statement so that the parent’s execution waits until the child’s process termination.
And the statement “exit(0);” is added on the last line so that the program exits with a normal status (passing the status code 0).

4. “I am child” with process id 31136 prints first because the parent has been suspended with the statement wait, until it’s child terminates.

PART III: Command Line arguments

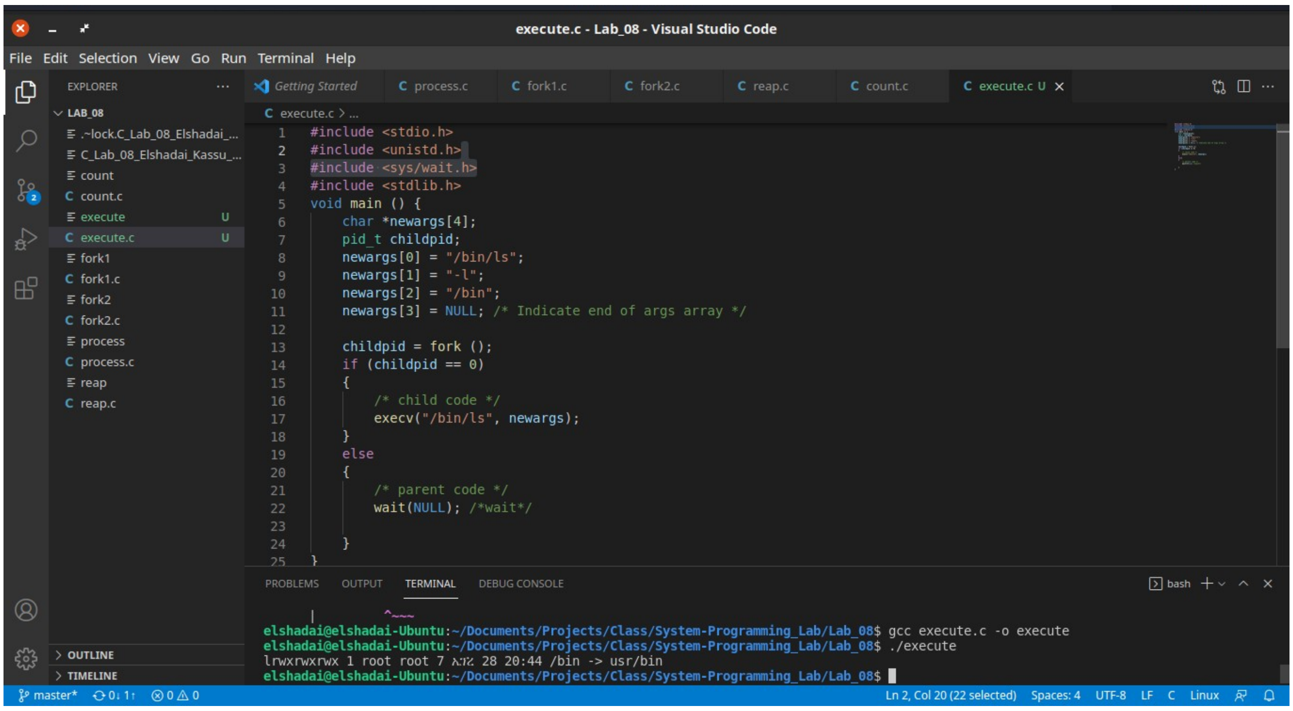

count.c - Lab_08 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB_08
.~lock.C_Lab_08_Elshadai_...
C_Lab_08_Elshadai_Kassu_...
count U
count.c U
fork1
fork1.c
fork2
fork2.c
process
process.c
reap
reap.c
OUTLINE
TIMELINE
Getting Started
process.c
fork1.c
fork2.c
reap.c
count.c U
count.c > main(int, char * [])
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
bash
master* 0 0
Ln 3, Col 6 Spaces: 4 UTF-8 LF C Linux

```c
#include<stdio.h>
#include <stdlib.h>
void main(int argc, char *argv[]) {
    int n=0;
    /* Findout if any word passed to program */
    if(argc==1) {
        printf("No word to examine. \n");
        exit(0);
    }
    /* Loop to count characters */
    while(argv[1][n++] != '\0');
        /*print result */
        printf("The word %s has %d characters .\n",argv[1],n);
}
```

```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./count
No word to examine.
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./count Elshadai
The word Elshadai has 9 characters .
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$
```

Part IV: Running other programs using execv system call

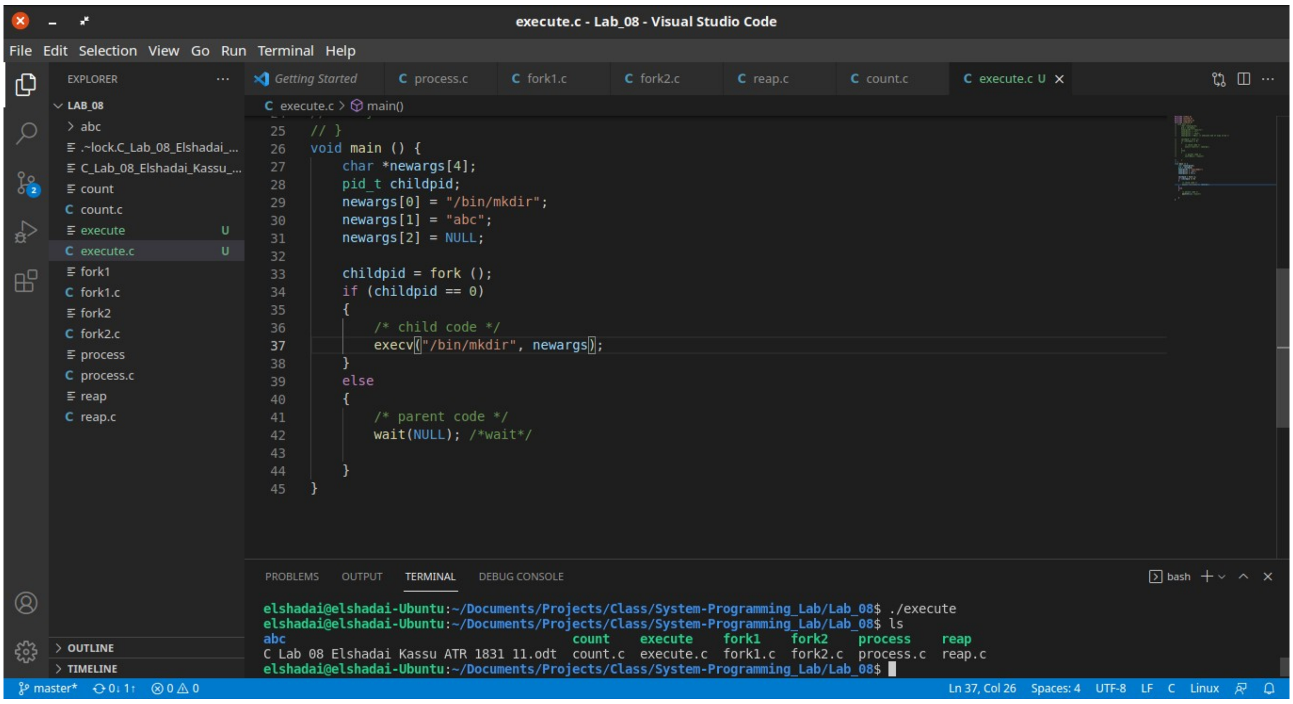
1.



```c
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdlib.h>
void main () {
    char *newargs[4];
    pid_t childpid;
    newargs[0] = "/bin/ls";
    newargs[1] = "-l";
    newargs[2] = "/bin";
    newargs[3] = NULL; /* Indicate end of args array */

    childpid = fork ();
    if (childpid == 0)
    {
        /* child code */
        execv("/bin/ls", newargs);
    }
    else
    {
        /* parent code */
        wait(NULL); /*wait*/

    }
}
```

```
      |          ^~~~
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ gcc execute.c -o execute
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./execute
lrwxrwxrwx 1 root root 7 ኦገስ 28 20:44 /bin -> usr/bin
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ 
```

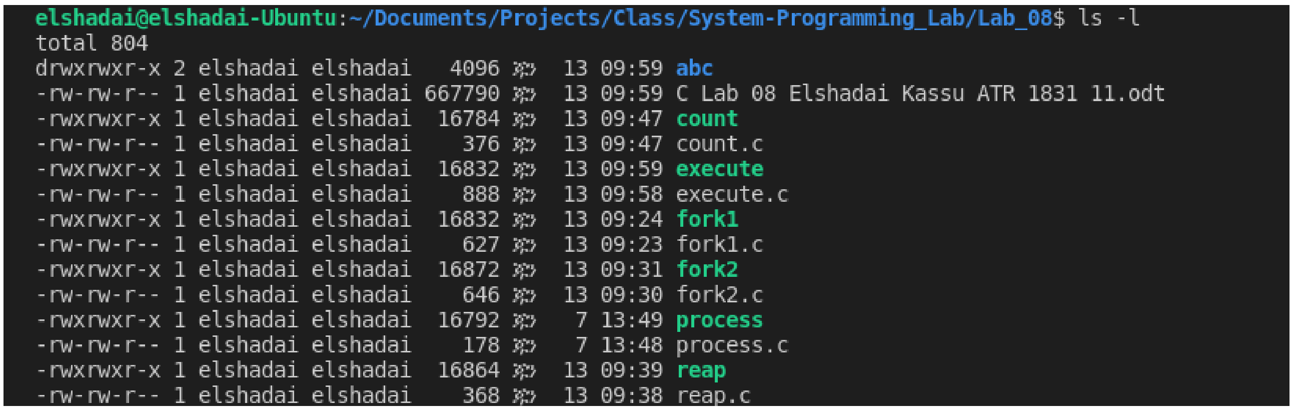
2.



```c
// }
void main () {
    char *newargs[4];
    pid_t childpid;
    newargs[0] = "/bin/mkdir";
    newargs[1] = "abc";
    newargs[2] = NULL;

    childpid = fork ();
    if (childpid == 0)
    {
        /* child code */
        execv("/bin/mkdir", newargs);
    }
    else
    {
        /* parent code */
        wait(NULL); /*wait*/

    }
}
```

```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ./execute
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ls
abc                                   count    execute    fork1    fork2    process    reap
C Lab 08 Elshadai Kassu ATR 1831 11.odt  count.c  execute.c  fork1.c  fork2.c  process.c  reap.c
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ 
```

3.



```
elshadai@elshadai-Ubuntu:~/Documents/Projects/Class/System-Programming_Lab/Lab_08$ ls -l
total 804
drwxrwxr-x 2 elshadai elshadai   4096 [illegible]  13 09:59 abc
-rw-rw-r-- 1 elshadai elshadai 667790 [illegible]  13 09:59 C Lab 08 Elshadai Kassu ATR 1831 11.odt
-rwxrwxr-x 1 elshadai elshadai  16784 [illegible]  13 09:47 count
-rw-rw-r-- 1 elshadai elshadai    376 [illegible]  13 09:47 count.c
-rwxrwxr-x 1 elshadai elshadai  16832 [illegible]  13 09:59 execute
-rw-rw-r-- 1 elshadai elshadai    888 [illegible]  13 09:58 execute.c
-rwxrwxr-x 1 elshadai elshadai  16832 [illegible]  13 09:24 fork1
-rw-rw-r-- 1 elshadai elshadai    627 [illegible]  13 09:23 fork1.c
-rwxrwxr-x 1 elshadai elshadai  16872 [illegible]  13 09:31 fork2
-rw-rw-r-- 1 elshadai elshadai    646 [illegible]  13 09:30 fork2.c
-rwxrwxr-x 1 elshadai elshadai  16792 [illegible]   7 13:49 process
-rw-rw-r-- 1 elshadai elshadai    178 [illegible]   7 13:48 process.c
-rwxrwxr-x 1 elshadai elshadai  16864 [illegible]  13 09:39 reap
-rw-rw-r-- 1 elshadai elshadai    368 [illegible]  13 09:38 reap.c
```

4. Yes, it does.