

Name: Muhammad Usman

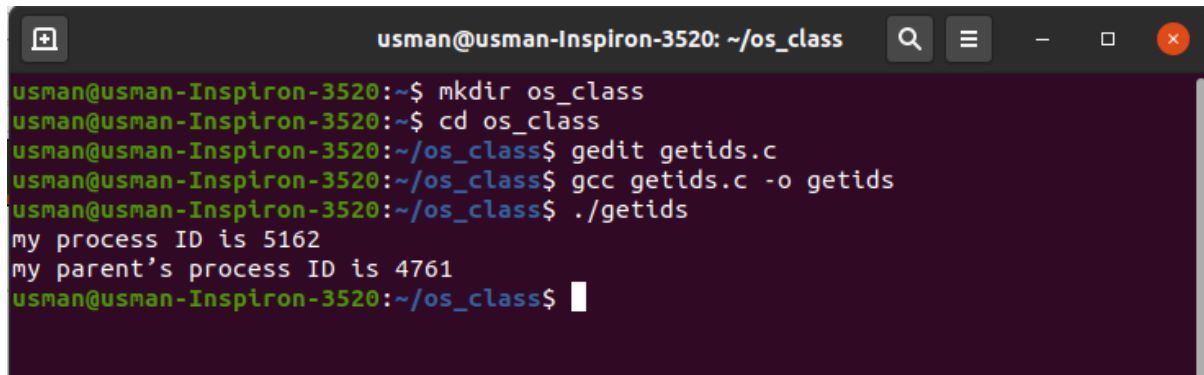
Roll no: 19P-0116

Course: CS220 Operating Systems

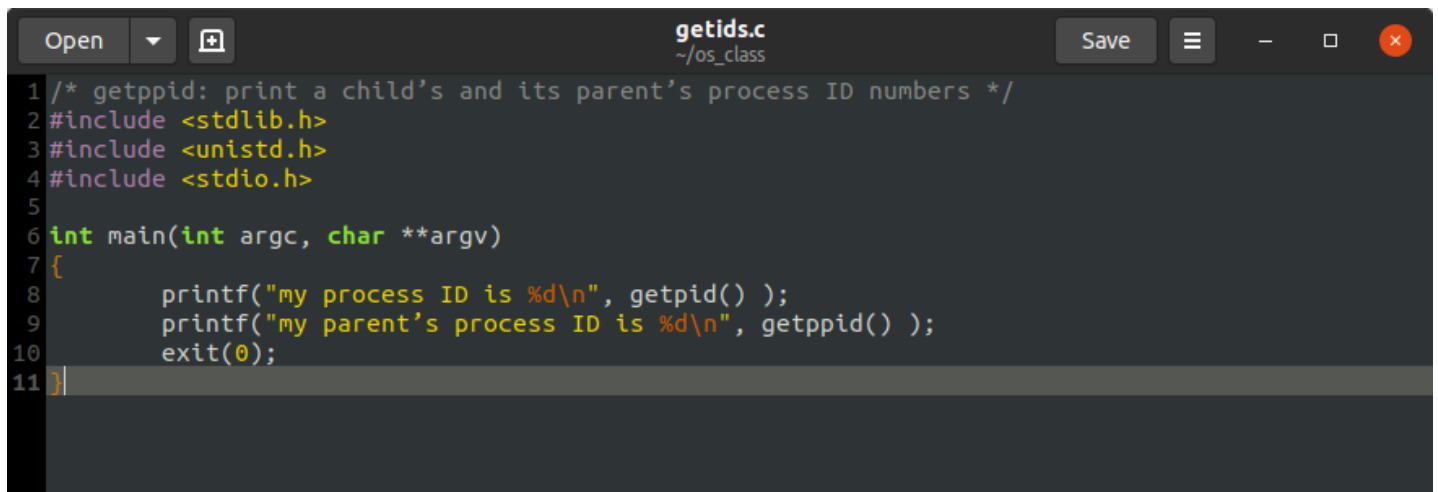
Section: A

Assignment # 2

Question 1:




```
usman@usman-Inspiron-3520: ~/os_class
usman@usman-Inspiron-3520:~$ mkdir os_class
usman@usman-Inspiron-3520:~$ cd os_class
usman@usman-Inspiron-3520:~/os_class$ gedit getids.c
usman@usman-Inspiron-3520:~/os_class$ gcc getids.c -o getids
usman@usman-Inspiron-3520:~/os_class$ ./getids
my process ID is 5162
my parent's process ID is 4761
usman@usman-Inspiron-3520:~/os_class$
```

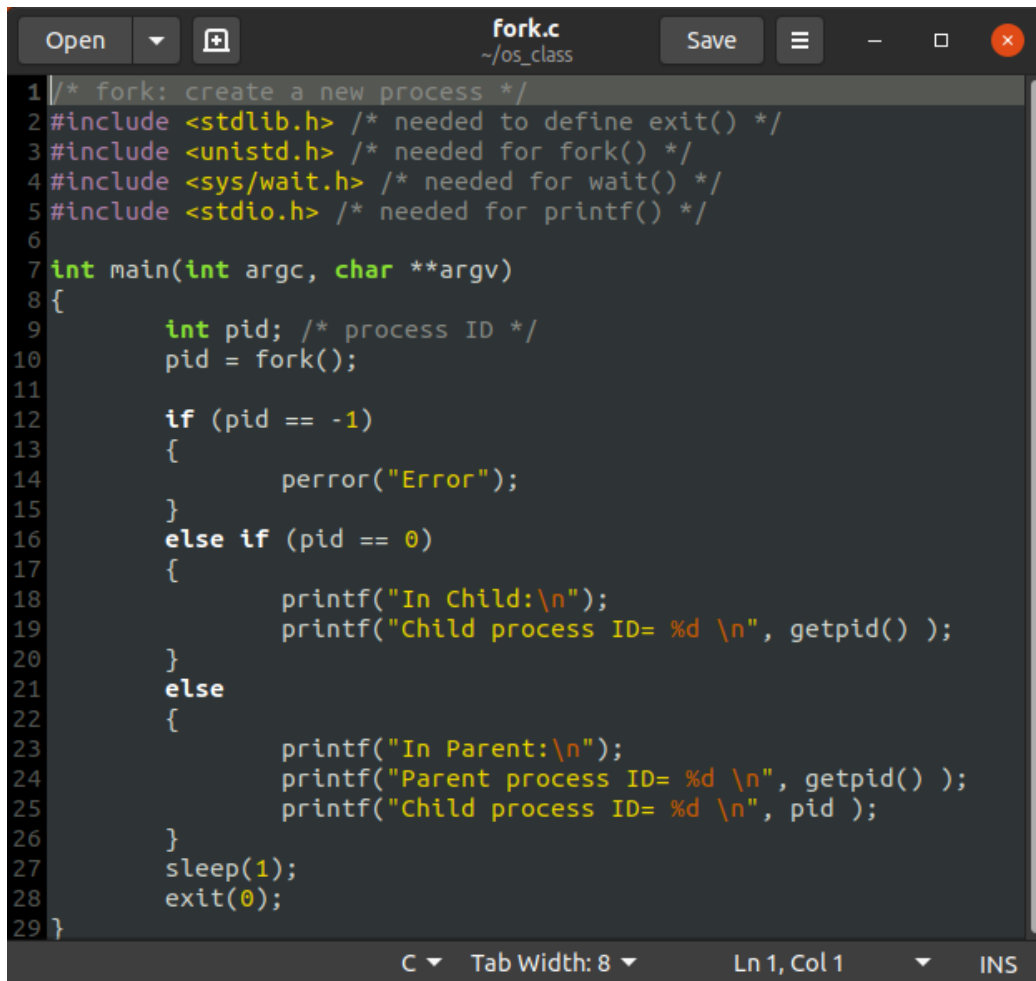


```
Open getids.c ~/os_class Save
1 /* getpid: print a child's and its parent's process ID numbers */
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <stdio.h>
5
6 int main(int argc, char **argv)
7 {
8     printf("my process ID is %d\n", getpid() );
9     printf("my parent's process ID is %d\n", getppid() );
10    exit(0);
11 }
```

Question 2:



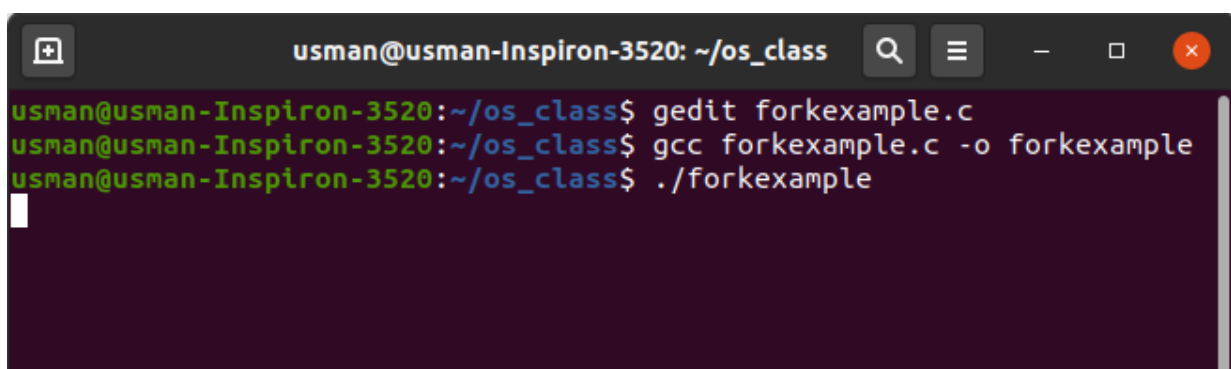
```
usman@usman-Inspiron-3520: ~/os_class
usman@usman-Inspiron-3520:~/os_class$ gedit fork.c
usman@usman-Inspiron-3520:~/os_class$ gcc fork.c -o fork
usman@usman-Inspiron-3520:~/os_class$ ./fork
In Parent:
Parent process ID= 5395
Child process ID= 5396
In Child:
Child process ID= 5396
usman@usman-Inspiron-3520:~/os_class$
```



The screenshot shows a code editor window titled 'fork.c' with the file path '~/.os_class'. The code is a C program that demonstrates the use of the 'fork()' system call to create a new process. It includes standard headers for `<stdlib.h>`, `<unistd.h>`, `<sys/wait.h>`, and `<stdio.h>`. The `main` function takes `argc` and `argv` as arguments. It declares a `pid` variable and calls `fork()`. If `fork()` returns `-1`, it indicates an error, and `perror("Error")` is called. If it returns `0`, the child process prints its own PID using `getpid()`. If it returns a positive value, the parent process prints both its own PID and the child's PID. The parent process then sleeps for 1 second before exiting with `exit(0)`. The status bar at the bottom indicates 'C', 'Tab Width: 8', 'Ln 1, Col 1', and 'INS'.

```
1 /* fork: create a new process */
2 #include <stdlib.h> /* needed to define exit() */
3 #include <unistd.h> /* needed for fork() */
4 #include <sys/wait.h> /* needed for wait() */
5 #include <stdio.h> /* needed for printf() */
6
7 int main(int argc, char **argv)
8 {
9     int pid; /* process ID */
10    pid = fork();
11
12    if (pid == -1)
13    {
14        perror("Error");
15    }
16    else if (pid == 0)
17    {
18        printf("In Child:\n");
19        printf("Child process ID= %d \n", getpid() );
20    }
21    else
22    {
23        printf("In Parent:\n");
24        printf("Parent process ID= %d \n", getpid() );
25        printf("Child process ID= %d \n", pid );
26    }
27    sleep(1);
28    exit(0);
29 }
```

Question 3:



The screenshot shows a terminal window with the prompt `usman@usman-Inspiron-3520: ~/.os_class`. The user has executed three commands: `gedit forkexample.c` to open the file in a text editor, `gcc forkexample.c -o forkexample` to compile the C program into an executable named `forkexample`, and `./forkexample` to run the program. The terminal shows the output of the program, which prints the PIDs of the parent and child processes.

```
usman@usman-Inspiron-3520:~/.os_class$ gedit forkexample.c
usman@usman-Inspiron-3520:~/.os_class$ gcc forkexample.c -o forkexample
usman@usman-Inspiron-3520:~/.os_class$ ./forkexample
```

```
Open  forkexample.c  Save  -  □  ×
~/os_class

1 /* fork: create a new process */
2 #include <stdlib.h> /* needed to define exit() */
3 #include <unistd.h> /* needed for fork() */
4 #include <stdio.h> /* needed for printf() */
5
6 int main(int argc, char **argv)
7 {
8     fork();
9     fork();
10    fork();
11    sleep(10000);
12    exit(0);
13 }
```

C Tab Width: 8 Ln 13, Col 2 INS

```
usman@usman-Inspiron-3520: ~
usman@usman-Inspiron-3520:~$ ps aux | grep forkexample
usman      5721  0.0  0.0  2356   520 pts/0    S+   16:31   0:00 ./forkexample
usman      5722  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5723  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5724  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5725  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5726  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5727  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5728  0.0  0.0  2356    80 pts/0    S+   16:31   0:00 ./forkexample
usman      5750  0.0  0.0  9372   732 pts/1    S+   16:31   0:00 grep --color=auto forkexample
usman@usman-Inspiron-3520:~$
```