



NATIONAL TEXTILE
UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

Subject

Operating System

SUBMITTED BY:

Fatima Waseem:

23-NTU-CS1155

SECTION SE: 5th (A)

SUBMITTED TO:

Sir Nasir

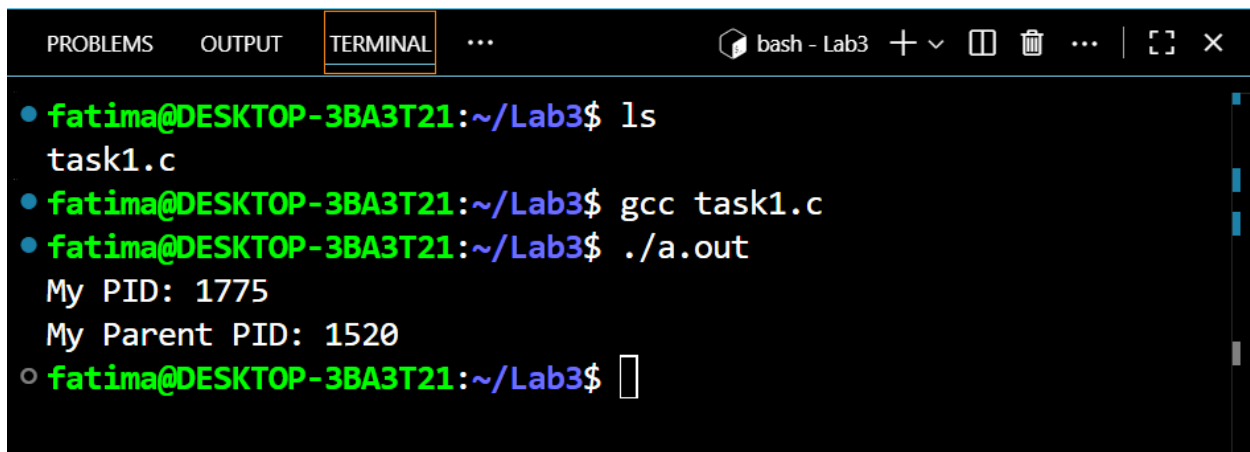
Lab 3

Task1: Print PID and PPID

Code:

```
#include <stdio.h>
#include <unistd.h>
int main() {
    printf("My PID: %d\n", getpid());
    printf("My Parent PID: %d\n", getppid());
    return 0;
}
```

Output:

A screenshot of a terminal window with a dark background. The window has a title bar with 'bash - Lab3' and several icons. The terminal shows a series of commands and their outputs. The prompt is 'fatima@DESKTOP-3BA3T21:~/Lab3\$'. The commands are 'ls', 'gcc task1.c', and './a.out'. The outputs are 'task1.c', 'My PID: 1775', and 'My Parent PID: 1520'. The terminal also shows a list of tabs at the top: 'PROBLEMS', 'OUTPUT', 'TERMINAL', and '...'.

```
fatima@DESKTOP-3BA3T21:~/Lab3$ ls
task1.c
fatima@DESKTOP-3BA3T21:~/Lab3$ gcc task1.c
fatima@DESKTOP-3BA3T21:~/Lab3$ ./a.out
My PID: 1775
My Parent PID: 1520
fatima@DESKTOP-3BA3T21:~/Lab3$
```

Task2: Fork – Creating Child Process

Code:

```
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid = fork();
```

```

    if (pid == 0) {
        // This block runs in the child process
        printf("Child: PID=%d, Parent=%d\n", getpid(),
getppid());
    } else {
        // This block runs in the parent process
        printf("Parent: PID=%d, Child=%d\n", getpid(), pid);
    }

    return 0;
}

```

output:

```

• fatima@DESKTOP-3BA3T21:~/Lab3$ gcc task2.c
• fatima@DESKTOP-3BA3T21:~/Lab3$ ./a.out
Parent: PID=5966, Child=5967
Child: PID=5967, Parent=5966
○ fatima@DESKTOP-3BA3T21:~/Lab3$ █

```

Task3: Excel – Replacing a Process

Code:

```

#include <stdio.h>
#include <unistd.h>
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        execlp("ls", "ls", "-l", NULL);
        printf("This will not print if exec succeeds.\n");
    } else {
        printf("Parent still running...\n");
    }
    return 0;
}

```

Out put:

```
fatima@DESKTOP-3BA3T21:~/Lab3$ ./a.out
Parent still running...
fatima@DESKTOP-3BA3T21:~/Lab3$ total 28
-rwxr-xr-x 1 fatima fatima 16048 Oct  3 15:43 a.out
-rw-r--r-- 1 fatima fatima   139 Oct  3 15:18 task1.c
-rw-r--r-- 1 fatima fatima   359 Oct  3 15:40 task2.c
-rw-r--r-- 1 fatima fatima   230 Oct  3 15:43 task3.c
```

Task4: Wait – Synchronization

Code:

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        execlp("ls", "ls", "-l", NULL);
        printf("This will not print if exec succeeds.\n");
    } else {
        waitpid(pid, NULL, 0); // Wait for the child process to finish
        printf("Parent still running...\n");
    }
    return 0;
}
```

OutPut:

```
fatima@DESKTOP-3BA3T21:~/Lab3$ gcc task4.c
fatima@DESKTOP-3BA3T21:~/Lab3$ ./a.out
total 32
-rwxr-xr-x 1 fatima fatima 16088 Oct  3 15:47 a.out
-rw-r--r-- 1 fatima fatima   139 Oct  3 15:18 task1.c
-rw-r--r-- 1 fatima fatima   359 Oct  3 15:40 task2.c
-rw-r--r-- 1 fatima fatima   230 Oct  3 15:43 task3.c
-rw-r--r-- 1 fatima fatima   315 Oct  3 15:46 task4.c
Parent still running...
```