

## LAB 3

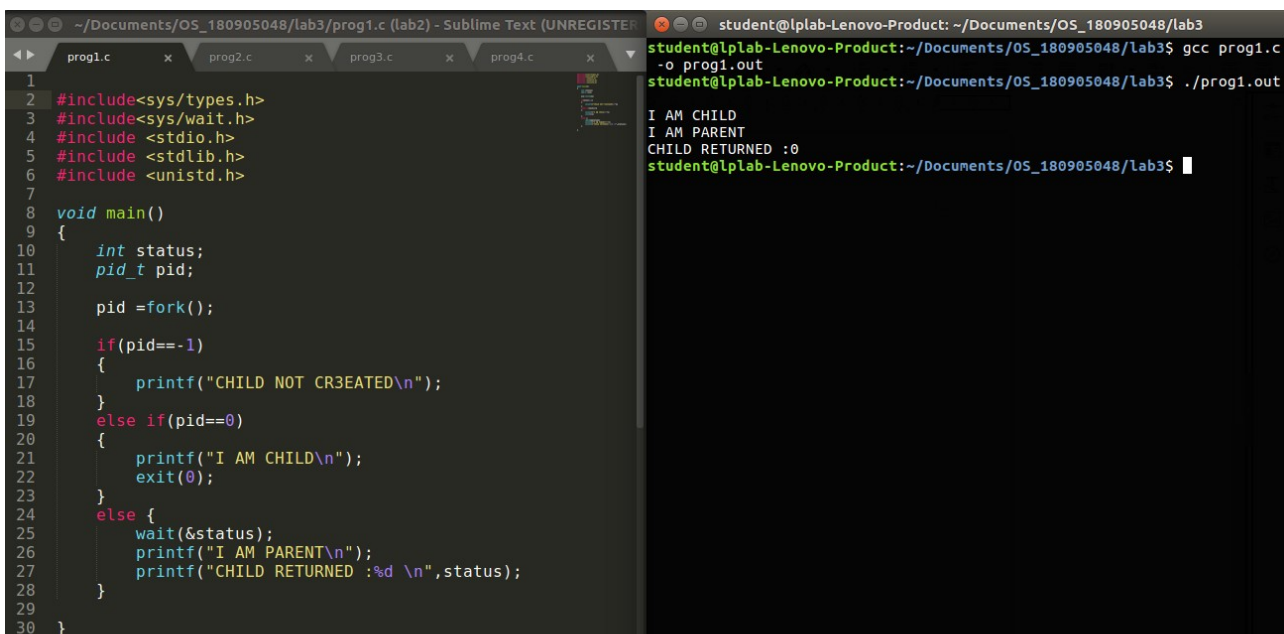
Sahil Saini Salaria  
Reg no. 180905048  
Roll No. 11C  
OS lab

Q1

```
#include<sys/types.h>
#include<sys/wait.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

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

    if(pid==-1)
    {
        printf("CHILD NOT CR3EATED\n");
    }
    else if(pid==0)
    {
        printf("I AM CHILD\n");
        exit(0);
    }
    else {
        wait(&status);
        printf("I AM PARENT\n");
        printf("CHILD RETURNED :%d \n",status);
    }
}
```



The screenshot shows a Sublime Text editor window on the left with the C program code, and a terminal window on the right showing the output of the program. The code in the editor is identical to the one provided in the previous blocks. The terminal output shows the program being compiled with 'gcc prog1.c' and then executed with './prog1.out'. The output of the execution is:

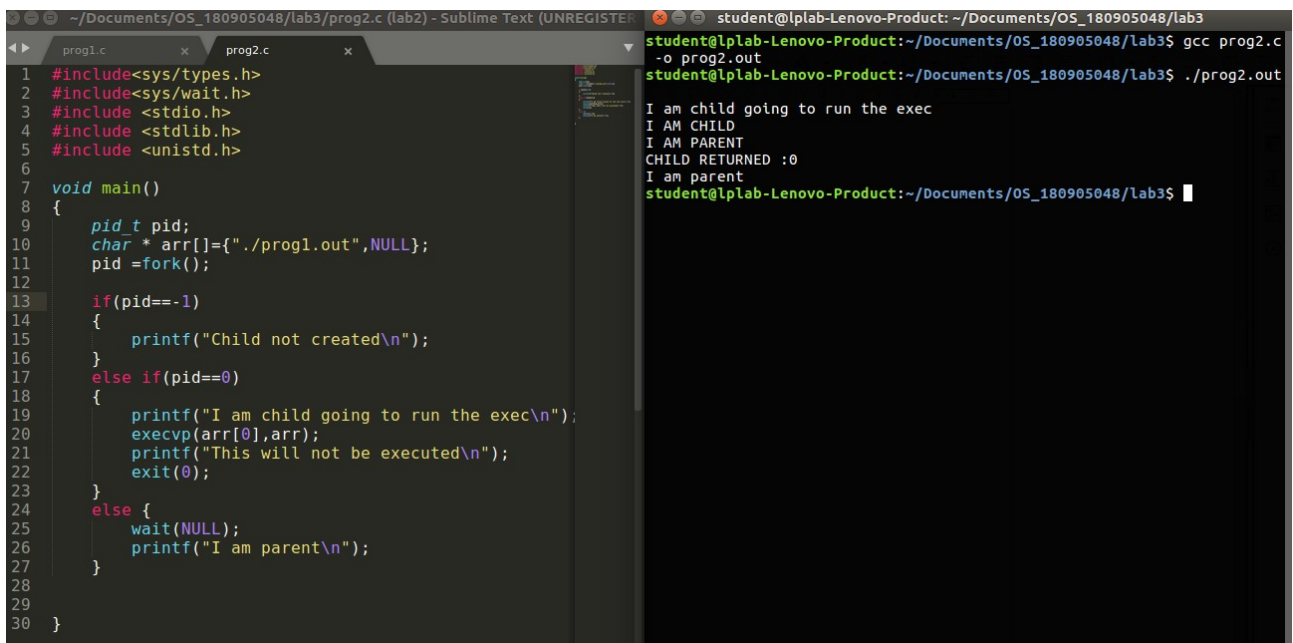
```
I AM CHILD
I AM PARENT
CHILD RETURNED :0
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$
```

Q2

```
#include<sys/types.h>
#include<sys/wait.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

```
void main()
{
    pid_t pid;
    char * arr[]={"/prog1.out",NULL};
    pid =fork();

    if(pid== -1)
    {
        printf("Child not created\n");
    }
    else if(pid==0)
    {
        printf("I am child going to run the exec\n");
        execvp(arr[0],arr);
        printf("This will not be executed\n");
        exit(0);
    }
    else {
        wait(NULL);
        printf("I am parent\n");
    }
}
```



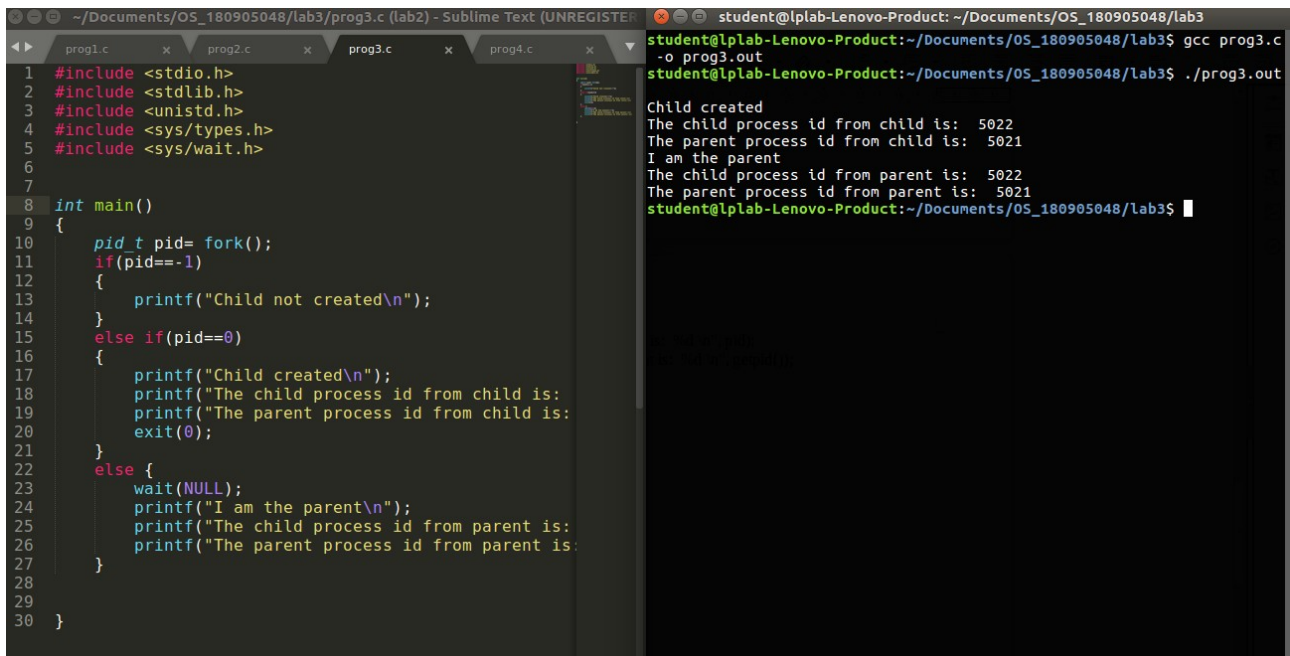
```
~/Documents/OS_180905048/lab3/prog2.c (lab2) - Sublime Text (UNREGISTERED)
1 #include<sys/types.h>
2 #include<sys/wait.h>
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <unistd.h>
6
7 void main()
8 {
9     pid_t pid;
10    char * arr[]={"/prog1.out",NULL};
11    pid =fork();
12
13    if(pid== -1)
14    {
15        printf("Child not created\n");
16    }
17    else if(pid==0)
18    {
19        printf("I am child going to run the exec\n");
20        execvp(arr[0],arr);
21        printf("This will not be executed\n");
22        exit(0);
23    }
24    else {
25        wait(NULL);
26        printf("I am parent\n");
27    }
28
29
30 }
```

```
student@lplab-Lenovo-Product: ~/Documents/OS_180905048/lab3
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$ gcc prog2.c
-o prog2.out
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$ ./prog2.out
I am child going to run the exec
I AM CHILD
I AM PARENT
CHILD RETURNED : 0
I am parent
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$
```

Q3

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
```

```
int main()
{
    pid_t pid= fork();
    if(pid==-1)
    {
        printf("Child not created\n");
    }
    else if(pid==0)
    {
        printf("Child created\n");
        printf("The child process id from child is: %d \n", getpid());
        printf("The parent process id from child is: %d \n", getppid());
        exit(0);
    }
    else {
        wait(NULL);
        printf("I am the parent\n");
        printf("The child process id from parent is: %d \n", pid);
        printf("The parent process id from parent is: %d \n", getpid());
    }
}
```



```
~/Documents/OS_180905048/lab3/prog3.c (lab2) - Sublime Text (UNREGISTERED)
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 #include <sys/wait.h>
6
7
8 int main()
9 {
10     pid_t pid= fork();
11     if(pid==-1)
12     {
13         printf("Child not created\n");
14     }
15     else if(pid==0)
16     {
17         printf("Child created\n");
18         printf("The child process id from child is:
19         printf("The parent process id from child is:
20         exit(0);
21     }
22     else {
23         wait(NULL);
24         printf("I am the parent\n");
25         printf("The child process id from parent is:
26         printf("The parent process id from parent is:
27     }
28
29
30 }
```

```
student@lplab-Lenovo-Product: ~/Documents/OS_180905048/lab3$ gcc prog3.c
-o prog3.out
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$ ./prog3.out
Child created
The child process id from child is: 5022
The parent process id from child is: 5021
I am the parent
The child process id from parent is: 5022
The parent process id from parent is: 5021
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$
```

Q4

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
int main()
{
    pid_t pid;
    pid=fork();
    if(pid==-1){
        printf("Fork failed\n");
        exit(0);
    }
    else if(pid==0)
    {
        printf("In Child:\n");
        for(int i=0;i<10;i++)
        {
            printf("Child process - %d\n",i+1);
        }
        printf("Parent PID : %d\nChild PID : %d\n",getppid(),getpid());
        exit(0);
    }
    else
    {
        printf("In Parent:\n");
        for(int i=0;i<2;i++)
            printf("Parent process - %d\n",i+1);
        sleep(10);
        int p=fork();
        if(p==0){
            printf("In child\n");
            sleep(1);
            printf("Parent pid : %d\n",getppid());
        }
        else{
            printf("Exiting parent\n");
            exit(0);
        }
    }
    return 0;
}
```

```
~/Documents/OS_180905048/lab3/prog4.c (lab2) - Sublime Text (UNREGISTERED)
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <sys/types.h>
4 #include <sys/wait.h>
5 #include <unistd.h>
6 int main()
7 {
8     pid_t pid;
9     pid=fork();
10    if(pid==-1){
11        printf("Fork failed\n");
12        exit(0);
13    }
14    else if(pid==0)
15    {
16        printf("In Child:\n");
17        for(int i=0;i<10;i++)
18        {
19            printf("Child process - %d\n",i+1);
20        }
21        printf("Parent PID : %d\nChild PID : %d\n",getppid(),getpid());
22        exit(0);
23    }
24    else
25    {
26        printf("In Parent:\n");
27        for(int i=0;i<2;i++)
28            printf("Parent process - %d\n",i+1);
29        sleep(10);
30        int p=fork();
31        if(p==0){
32            printf("In child\n");
33            sleep(1);
34            printf("Parent pid : %d\n",getppid());
35        }
36    }
37 }
```

```
student@lplab-Lenovo-Product: ~/Documents/OS_180905048/lab3
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$ ./prog4.out
In Parent:
Parent process - 1
Parent process - 2
In Child:
Child process - 1
Child process - 2
Child process - 3
Child process - 4
Child process - 5
Child process - 6
Child process - 7
Child process - 8
Child process - 9
Child process - 10
Parent PID : 6952
Child PID : 6953
Exiting parent
In child
student@lplab-Lenovo-Product:~/Documents/OS_180905048/lab3$ Parent pid
: 1533
```

//////////////////////////////////////END//////////////////////////////////////