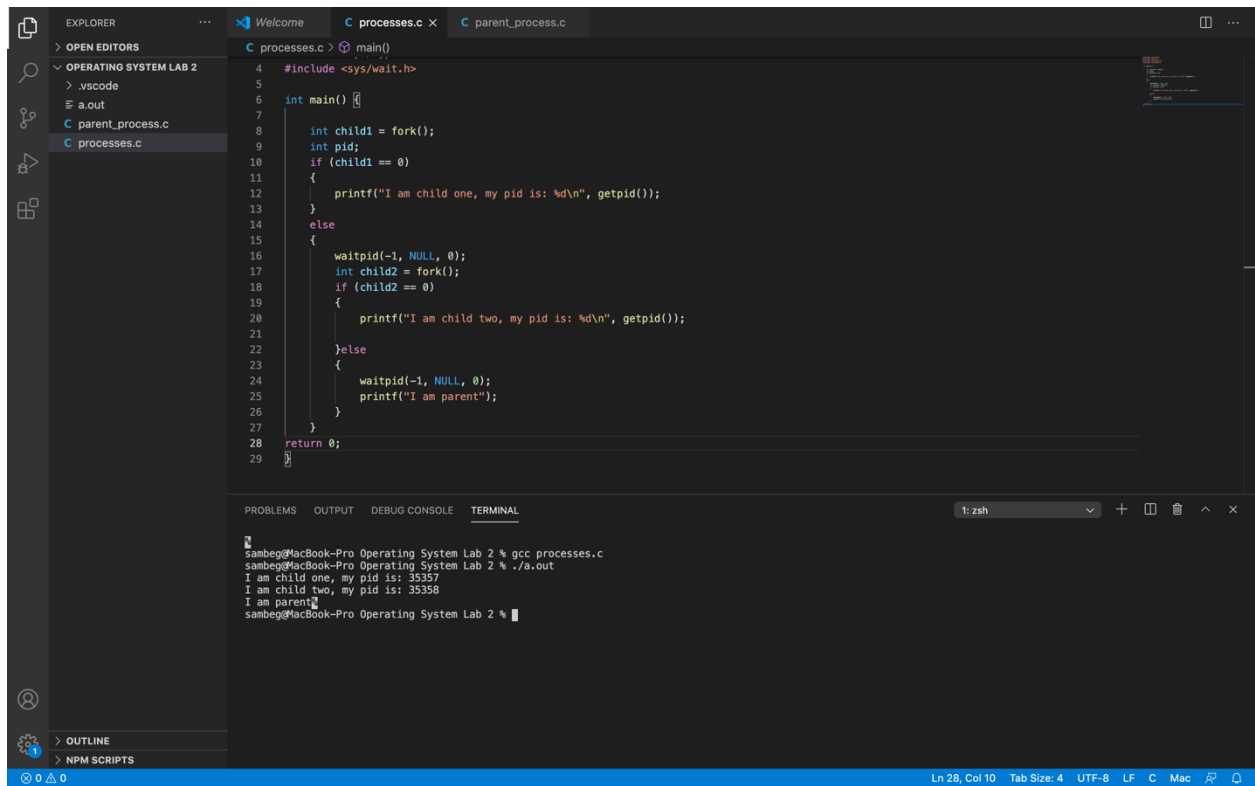


Sambeg Raj Subedi

Date: 03/11/2021

Lab Report 2

Part 1:



The screenshot shows a Visual Studio Code editor with two open files: `processes.c` and `parent_process.c`. The `processes.c` file is active and contains the following C code:

```
1 #include <sys/wait.h>
2
3 int main()
4 {
5     int child1 = fork();
6     int pid;
7     if (child1 == 0)
8     {
9         printf("I am child one, my pid is: %d\n", getpid());
10    }
11    else
12    {
13        waitpid(-1, NULL, 0);
14        int child2 = fork();
15        if (child2 == 0)
16        {
17            printf("I am child two, my pid is: %d\n", getpid());
18        }
19        else
20        {
21            waitpid(-1, NULL, 0);
22            printf("I am parent");
23        }
24    }
25    return 0;
26 }
```

The terminal output at the bottom shows the execution of the program:

```
sambeg@MacBook-Pro Operating System Lab 2 % gcc processes.c
sambeg@MacBook-Pro Operating System Lab 2 % ./a.out
I am child one, my pid is: 35357
I am child two, my pid is: 35358
I am parent
sambeg@MacBook-Pro Operating System Lab 2 %
```

Part 2:

First run:

```
42 }
43
44 }else
45 {
46     b = a + b - 5;
47     printf ("Parent Process P\n");
48     printf ("a value : %d\n", a);
49     printf ("b value : %d\n", b);
50     printf ("The value of PID: %d\n", getpid());
51     printf ("Parent's PID: %d\n", getppid());
52     printf ("\n");
53 }
54 return 0;
55 }
56 }
57
```

```
sam@MacBook-Pro Operating System Lab 2 % gcc parent_process.c
sam@MacBook-Pro Operating System Lab 2 % ./a.out
Parent Process P
a value : 10
b value : 30
The value of PID: 35435
Parent's PID: 35424

Process Q
a value: 35
b value: 25
The value of PID: 35436
Parent's PID: 1

parent Process P
a value: 35
b value: 40
The value of PID: 35436
Parent's PID: 1

Process R
a value: 895
b value: 25
The value of PID: 35437
Parent's PID: 1

sam@MacBook-Pro Operating System Lab 2 %
```

Second run :

```
42 }
43
44 }else
45 {
46     b = a + b - 5;
47     printf ("Parent Process P\n");
48     printf ("a value : %d\n", a);
49     printf ("b value : %d\n", b);
50     printf ("The value of PID: %d\n", getpid());
51     printf ("Parent's PID: %d\n", getppid());
52     printf ("\n");
53 }
54 return 0;
55 }
56 }
57
```

```
sam@MacBook-Pro Operating System Lab 2 % gcc parent_process.c
sam@MacBook-Pro Operating System Lab 2 % ./a.out
Parent Process P
a value : 10
b value : 30
The value of PID: 35456
Parent's PID: 35447

Process Q
a value: 35
b value: 25
The value of PID: 35457
Parent's PID: 1

sam@MacBook-Pro Operating System Lab 2 %
parent Process P
a value: 35
b value: 40
The value of PID: 35457
Parent's PID: 1

Process R
a value: 895
b value: 25
The value of PID: 35458
Parent's PID: 35457
```

Here, As we know the parent process is P, and Q and R are the new processes often called as Child process. fork () system call is used twice to create a new child process. In the above output screenshot, we can see that the value of PID changes when we rerun the program. In this system, different conditional statements based on the value of fq and fr are used. We were provided the following framework. The modification of this code is also attached.

```
int a=10, b=25, fq=0, fr=0
fq=fork() // fork a child - call it Process Q
if(fq==0){ // Child successfully forked
a=a+b print values of a, b, and process_id
fr=fork() // fork another child - call it Process R
if(fr!=0){
b=b+15
//print values of a, b, and process_id
}else{
a=(a*b)+20
//print values of a, b, and process_id
}
}else{
b=a+b-5;
print values of a, b, and process_id
}
```