BLG312E – Operating Systems, Spring 2021

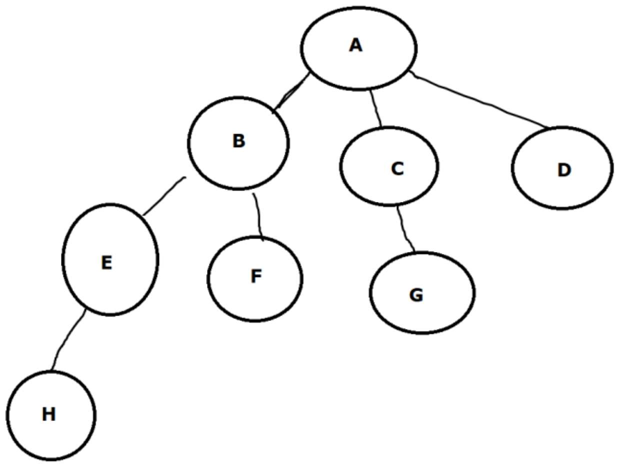
Assignment 1

Abdulkadir Pazar – 150180028

a)

8 processes will be created by the code. 1 process is parent and there are 7 children processes.

b)

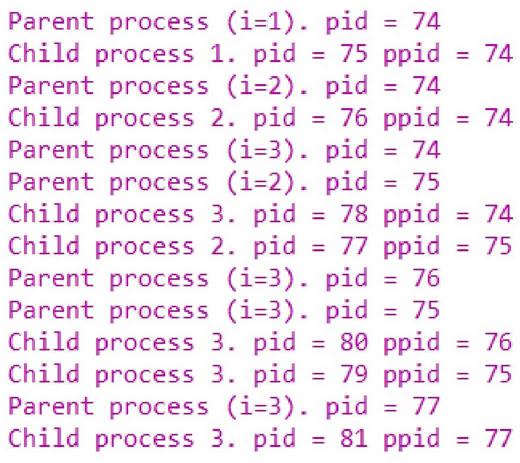


Tree created by the program

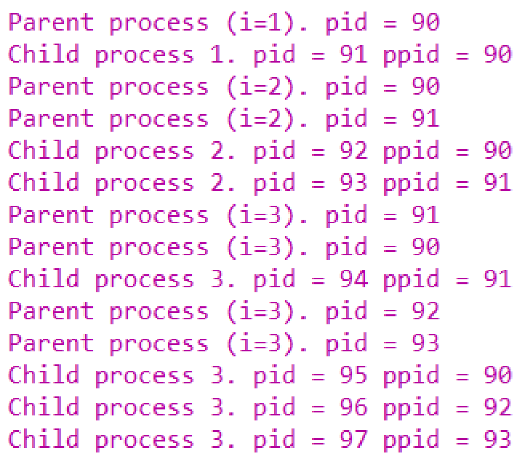
c)

fork() system call will be invoked 7 times as there are 7 child processes. printf() function will be invoked 14 times, twice per process. In general fork will be invoked 2^n-1 times and printf will be invoked $2^{n+1} – 2$ where n is the number in the loop condition i <= n.

d)



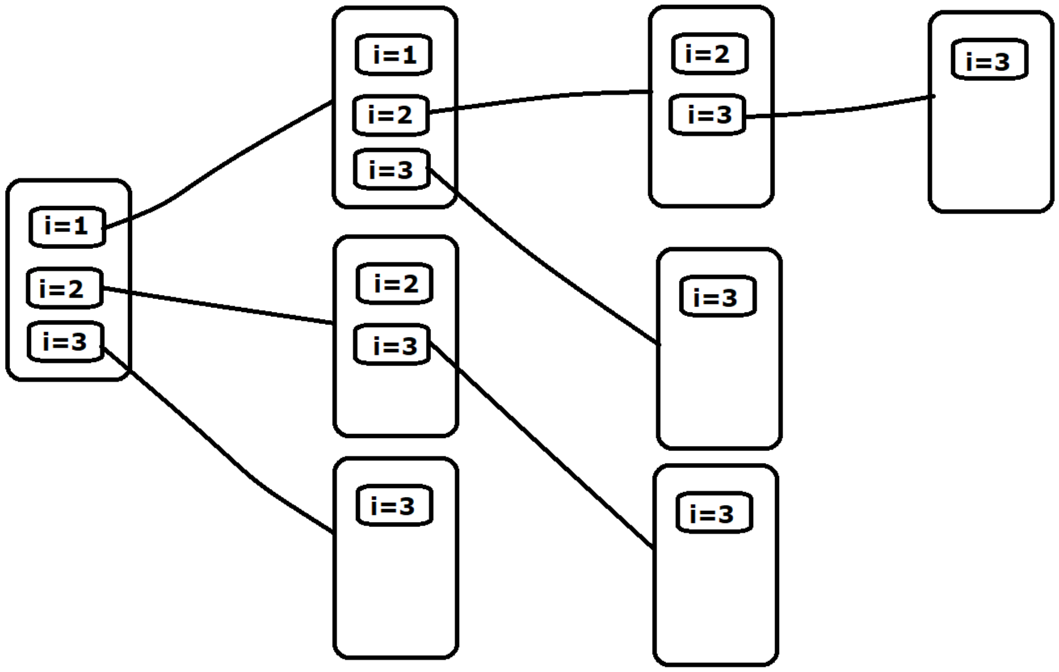
```
Parent process (i=1). pid = 74
Child process 1. pid = 75 ppid = 74
Parent process (i=2). pid = 74
Child process 2. pid = 76 ppid = 74
Parent process (i=3). pid = 74
Parent process (i=2). pid = 75
Child process 3. pid = 78 ppid = 74
Child process 2. pid = 77 ppid = 75
Parent process (i=3). pid = 76
Parent process (i=3). pid = 75
Child process 3. pid = 80 ppid = 76
Child process 3. pid = 79 ppid = 75
Parent process (i=3). pid = 77
Child process 3. pid = 81 ppid = 77
```



```
Parent process (i=1). pid = 90
Child process 1. pid = 91 ppid = 90
Parent process (i=2). pid = 90
Parent process (i=2). pid = 91
Child process 2. pid = 92 ppid = 90
Child process 2. pid = 93 ppid = 91
Parent process (i=3). pid = 91
Parent process (i=3). pid = 90
Child process 3. pid = 94 ppid = 91
Parent process (i=3). pid = 92
Parent process (i=3). pid = 93
Child process 3. pid = 95 ppid = 90
Child process 3. pid = 96 ppid = 92
Child process 3. pid = 97 ppid = 93
```

Two different outputs of the program

The program outputs cannot be predicted as the time a process will have the processor cannot be predicted since the processor will run a different process every time there is a context switch. The diagram below is helpful to visualize why the output cannot be predicted as the processes are represented as rounded rectangles and if the order of the boxes changes the output the program prints to the console changes.



e)

```
#include <stdio.h>

int main()
{
    int i = i;
    int result = 0;
    for (i = 1; i <= 3; i++)
    {
        result = fork();
        if (result == 0)
        {
            printf("Child process pid = %d ppid = %d. \n",getpid(), getppid());
            break;
        }
        else
        {
            printf("Parent process. pid= %d\n",getpid());
        }
    }
    return 0;
}
```

The code above will create 3 child processes which have the same parent process.