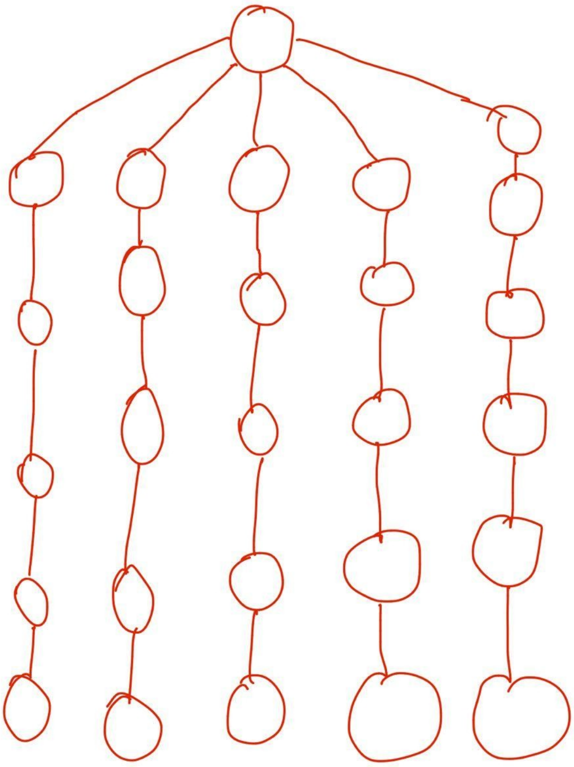
Yunus Emre Yiğit
150190107

BLG312E - Operating Systems
Assignment 1

1) This code creates processes as the function of n, where n is the depth, f(n) = n^2+1. Thus for n = 5, code generates 26 processes. As there is one process at the start, 25 of them generated later, 25 processes can be itentified as children.
2)



Picture 1: Process Tree

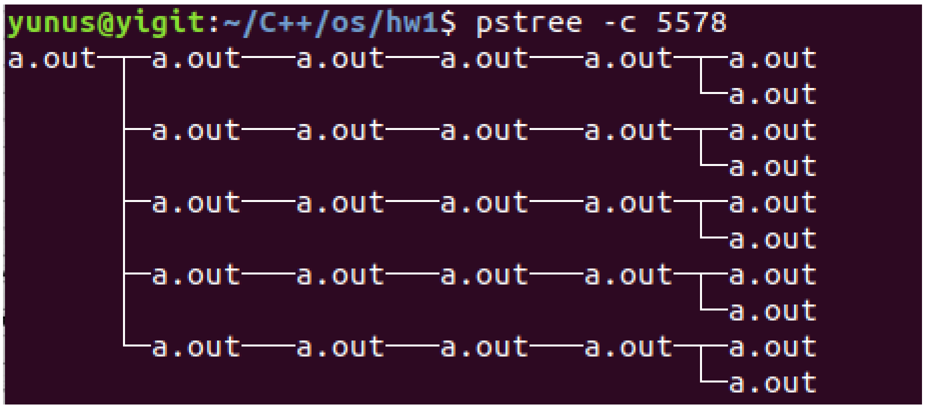
3) As stated above total number of process is f(n) = n^2+1, so depth must be 10.

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

int main(){
    int i, res, depth = 10; // modified part
    for (i = 0; i < depth; i++){
        res = fork();
        if (res == 0){
        depth = depth - 1;
            while(depth > 0){
                depth = depth - 1;
                res = fork();
                if(res != 0){
                    wait(NULL);
                    exit(0);
                }
            }
            exit(0);
        }
    }
    wait(NULL);
    exit(0);
}
```

4)

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

int main(){
    int i, res, depth = 5;
    for (i = 0; i < depth; i++){
        res = fork();
        if (res == 0){
        depth = depth - 1;
            while(depth > 0){
                depth = depth - 1;
                res = fork();

                if(depth == 0 && res != 0) // modified part
                    res = fork();

                if(res != 0){
                    wait(NULL);
                    exit(0);
                }
            }

            exit(0);
        }

    }
    wait(NULL);
    exit(0);
}
```

Proof that it works:



```
yunus@yigit:~/C++/os/hw1$ pstree -c 5578
a.out─┬─a.out───a.out───a.out───a.out─┬─a.out
      │                               └─a.out
      ├─a.out───a.out───a.out───a.out─┬─a.out
      │                               └─a.out
      ├─a.out───a.out───a.out───a.out─┬─a.out
      │                               └─a.out
      ├─a.out───a.out───a.out───a.out─┬─a.out
      │                               └─a.out
      └─a.out───a.out───a.out───a.out─┬─a.out
                                      └─a.out
```