6.1. Questions

1. What the output will be at LINE A?

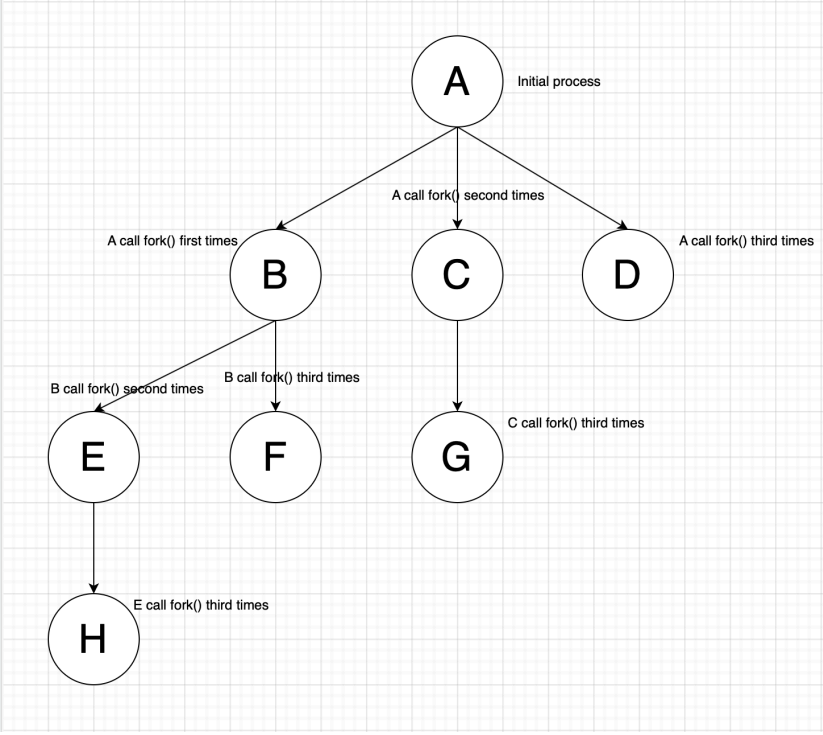
- After running the program, LINE A will print "PARENT: value = 5".In child process, value plus 15 but then exit. The line A is cointained in a part code of the parent process. Each of process have thier own variables soo changing the value of a variable in child process doesn't affect to variables in parent process.That is the reason why the value is 5.

```
angs-MacBook-Pro:Exercise 6.1.1 quan0402$ ./main
PARENT: value = 5
```

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

int value = 5;

int main(int argc , char **argv){
    pid_t pid;
    pid = fork();

    if(pid == 0){
        value += 15;
        return 0;
    }
    else if(pid >0){
        wait(0);
        printf("PARENT: value = %d\n",value);
        return 0;
    }

}
```

2. How many processes are created by the program shown below, including the initial parent process? How many process are created when n fork() called?

- There are 8 process are created by that program (including the initial parent process). We can use a diagram to show the tree which shows the relationship between these process.



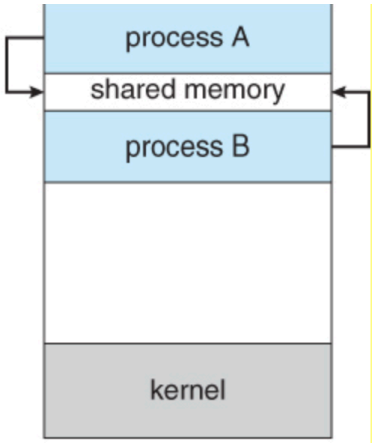
- There are 2^n-1 process which are created when the program call fork() n times. When a fork is called, there are 2 process will be exist: the initial process (parent process) and process which are created (child process). Sooo there will have 2^n process, including 1 initial process and 2^n-1 child process. For we can use this fomula to calculate the numbers of process in your program.

3. When a process creates a new process using the fork() operation, which of the following states is shared between the parent process and the child process? Why?

A. Stack

B. Heap

C. Shared memory segments



The answer : the Shared memory segments.

The reason: The parent process and child process only use and share data in the shared memory segments. Copies of the stack and the heap are made for the newly created process.

4. What process id (PID) and process group id are used for?

To optimize any programs, using fork() to create new process and divide work for every single process is necessary. We also have to how to manage and control all the processes to ensure that process is doing the work that it suppose to do. A process ID (PID) is a ID number to identify an active process. We can use PID as parameter in various function calls, conditional statement that allowing processes to be executed. Process groups are identified by a positive integer, the process group ID, which is the process identifier of the process that is (or was) the process group leader. It have the same function just like the PID: to manage and control the processes in program and make sure that they are doing the right work.