NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES ISLAMABAD

OPERATING SYSTEMS Spring 2021

ASSIGNMENT 02

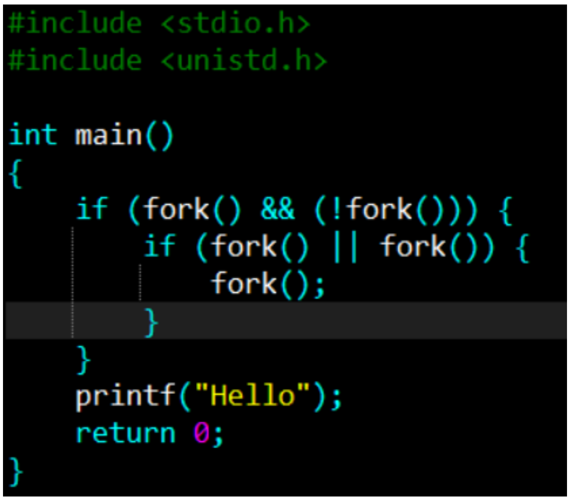
Due Date: 11:59 PM 13th, April 2021.

Instructions

- Zero marks will be awarded to the students involved in plagiarism.
- All the submissions will be done on classroom.
- You have to submit .c/.cpp files. Naming convention has to be followed strictly. Each question will be named as q1.cpp/q1.c. You have to submit 1 zipped file having the questions and pdf files (word files should be names as e.g., q1_pdf).
- No queries will be entertained in last two days.
- Be prepared for viva or anything else after the submission of assignment.

QUESTION NO. 01 [Marks 10: 5(Output) + 5(Tree)]

Write the output of below code and also draw the process tree. Submit pdf file of output and tree.

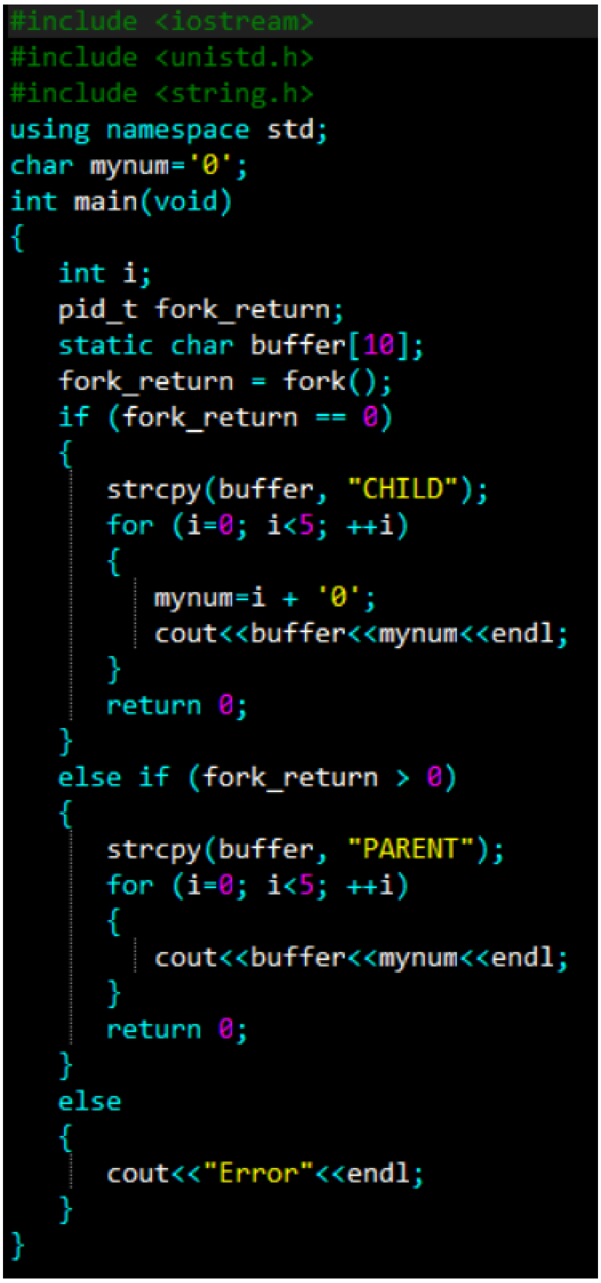


```
#include <stdio.h>
#include <unistd.h>

int main()
{
    if (fork() && (!fork())) {
        if (fork() || fork()) {
            fork();
        }
    }
    printf("Hello");
    return 0;
}
```

QUESTION NO. 02 [Marks 10: 5(Output) + 5(Explanation)]

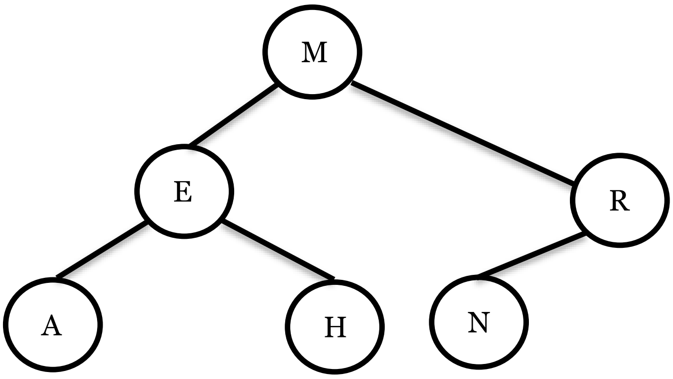
Write the output of below code and also explain the output. Submit pdf file of output and tree.



```
#include <iostream>
#include <unistd.h>
#include <string.h>
using namespace std;
char mynum='0';
int main(void)
{
    int i;
    pid_t fork_return;
    static char buffer[10];
    fork_return = fork();
    if (fork_return == 0)
    {
        strcpy(buffer, "CHILD");
        for (i=0; i<5; ++i)
        {
            mynum=i + '0';
            cout<<buffer<<mynum<<endl;
        }
        return 0;
    }
    else if (fork_return > 0)
    {
        strcpy(buffer, "PARENT");
        for (i=0; i<5; ++i)
        {
            cout<<buffer<<mynum<<endl;
        }
        return 0;
    }
    else
    {
        cout<<"Error"<<endl;
    }
}
```

QUESTION NO. 03 [Marks 20: P1 – 5 | P2 – 15]

Part 1:



Considering name e.g., **MEHRAN**, draw a binary tree on paper according to the given implications.

Tree will be of your full name, using upper case alphabet. Less character will be moved to **left**, **greater** and **equal** will be drawn on right side. If your name is composed of **three** names first, middle and last just take your middle and last name as full name. Also no need to take space between name.

Part 2:

Write a C/C++ program in Linux platform of your full name as refers to the generated tree in PART 1.
Each node has to display its parent id, child id, letter and ASCII value.

e.g. Node H: "PPID: 1223, PID: 1234 and ASCII: 072"

Each child will return its **character** with value as **ASCII** to the parent. All the parent node will add the child letter in its display statement.

e.g. Node E: "PPID: 2223, PID: 2234 and ASCII: 083 My child is H"

At the end parent will display the final output as sum of all characters. E.g. Parent **M** will get both the **E** and **R** sub trees **ASCII** values and will sum their return values. It should be a positive integer value.
You have to use **wait ()** system call for properly displaying the statements and tree hierarchy from bottom up view.

You can refer the below link for ASCII values:
http://sticksandstones.kstrom.com/appen.html

Important Note: Submit pdf file as well as code for this question. If you write generic code of part 2 will be awarded zero.

QUESTION NO. 04 [Marks 5 + 5]

Using **exec** system call list all the files and directories of home then store the result in “**output1.txt**”**.** Run a second command “**grep**” and find the word “**output1.txt**” in all the text files of the home also place home path in “**PATH**” environment variable while doing exec for grep. This all will be done in one process.

QUESTION NO. 05 [Marks 5 + 5]

Write a program that receives one command line argument: the name of a text file. Your program will work as follows:

- Start out by creating a child process.
- The child process calls exec to run **cat** command with command line argument that the parent received.
- The parent process calls wait so that it blocks until the child terminates and passes back its termination status.
- If the child process terminates without error, the parent spawns another child and, again, calls wait so that it can block until the child terminates.
- The new child calls exec again, but this time it runs **wc** command on the same argument that the parent received from the command line.
- Once the parent learns that the child has terminated, it goes on to terminate also. If the parent gets to this point, it’s because all has gone well.

QUESTION NO. 06 [Marks 10]

Write your shell program which take input commands/exe’s from user as your shell takes and execute them. Your shell will terminate when user enter “**quit**”. Your shell program should be able to handle arguments of commands/exe’s

QUESTION NO. 07 [Marks 10: 2 Marks for each proper task with exec]

Write a program which take string from user and you have to do following tasks with the string.

1. Reverse
2. Find Length
3. Add 2 in the ASCII of each character
4. Sort
5. Capitalize

You have to create child of child's as given below process tree for each task and each child exec with the image of program of particular task. Print the string after each operation.