CSE 320 Spring 2018 HW #3 Part 2

March 21, 2018

Deadline: April 1, 23:59 Stony Brook time (Eastern Time Zone).

1 Introduction

Goal of this homework is to exercise more writing C code, learn how to debug it, and start working with fork system call. This homework consists of three parts. In the first part you are required to read provided material that will help you to understand better the material and will prepare you for the rest two parts of the homework. Moreover, it will prepare you to other homeworks. We cannot emphasize more how important it is to go through readings as they will reduce amount of time you need to spend on the homework. Again, if you are spending too much time then most likely you are doing something wrong. In case you still have problems please talk to TAs and/or Professor.

In the second part you will need to write a simple application that will perform various actions based on the input from a user. During this part of the homework you will need to use `fork()` and `execve()` system calls. You can find more details in the corresponding doc.

In the third part part you will be given relatively big code base and you will need to fix it using `gdb`, `valgrind`, and you knowledge. You can find more details in the corresponding doc.

And we highly recommend to spend some time thinking about your code. You may want to design your code first, think about possible problems, and probably even write down it on paper. It may significantly reduce amount of time that you will spend on the writing code itself.

2 Part II

In this part you will need to write four applications. Three of them will be doing some simple tasks and their description you can find in the following subsections. The fourth one will be the main application, which will run one of those application. More details can be found in the corresponding subsection.

2.1 GitHub Link

For this part we will not provide any code and therefore no GitHub link will be given. Please refer to the "Submission" subsection for the details how to submit this part of the homework.

2.2 Application 1

The purpose of this application is to find a root of the equation $f(x) = 0$ with a certain tolerance, where $f(x)$ is a polynomial and it is guaranteed that there is exactly one root. This application will take a file as an input that contains degree of the polynomial, tolerance value ϵ, left and right bounds of the interval that contains root, and all coefficients of the polynomial starting from one that corresponds to the highest degree. More specifically, if $f(x) = \sum_{i=0}^{n} c_i x^i$ and there is a root of an equation $f(x) = 0$ on the interval $[A, B]$ then file has following format

$$n \quad \epsilon \quad A \quad B \quad c_n \quad c_{n-1} \quad \cdots \quad c_1 \quad c_0 \tag{1}$$

Application should print root into file with name "part2_solver.dat".

2.3 Application 2

This application will take a file as an input. Application will read two matrices from file, multiply them, and find trace of the resulting matrix. First line of a file contains number of rows and number of columns in the matrix A. Then each row of the matrix will be on a different line in the that file. Right after first matrix will go second matrix B in the same format. For example, following file

```
3 2
1 5
2 3
1 1
2 4
1 4 6 2
9 3 1 3
```

represents following matrices

$$\begin{pmatrix} 1 & 5 \\ 2 & 3 \\ 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 4 & 6 & 2 \\ 9 & 3 & 1 & 3 \end{pmatrix}$$

This application should find a trace of matrix C where $C = A \cdot B$.
Result should be printed file with name "part2_trace.dat".

2.4 Application 3

This application will calculate Nth Fibonacci number. Application will receive N as an argument. Important to note, N can be big so you need to being able to calculate that Fibonacci number relatively fast (matter of seconds).

Result should be printed file with name "part2_fib.dat".

2.5 Main application

This is the main application. It should take file name as an input and then prompt user which of other three applications to run. After that main application should `fork` a child and substitute child process with an application that user asked to run by calling `execve()` function. Once that application finishes its job the main application should ask user again which application to run. There also should be an option to quit main application and change input file name.

Name for the options should be following:

- **solver** for the application 1
- **trace** for the application 2
- **fib** for the application 3
- **quit** to exit application
- **change** to change input file name

An easy way to start testing your application is to come up with a simple input, solve it by hand, and compare your results with the results of your program. You also should account for different errors and/or edge cases such as zombies, non-existent files, etc. However, you may assume that content of a file will always be correct.

2.6 Submission

You will be provided GitHub link to create a repository in the Part III. You will need to put your code for Part II into that repository once it is created. We should be able to compile your code by simple typing "make part2". Name of the executable for the main application should be "part2".

2.7 Useful reading

First, reading text book and man pages for the `fork()` and `execve()` is important. You may also find following readings useful
https://en.wikipedia.org/wiki/Bisection_method
https://en.wikipedia.org/wiki/Trace_(linear_algebra)
https://en.wikipedia.org/wiki/Fibonacci_number
https://www.mathsisfun.com/numbers/fibonacci-sequence.html