March 24th, 2022

CSE344 – System Programming - Homework #2 - v1
Process creation, waiting and signals

Objective
The objective is to have a process read an input file, interpret its contents as coordinates, and forward them for calculations to children processes. Then it will collect their outputs and make a final calculation.

Process P
Process P will read an ASCII file (denoted by inputfilepath) provided as a commandline argument. It will interpret every character in it as un unsigned byte. Consecutively read 3 bytes are interpreted as integer coordinates in a 3-dimensional space. Every i-th group of consecutive 10 coordinates it reads, are forwarded to a child process R_i. The child process R_i will be created by P using the fork+exec paradigm. The 10 coordinates will be passed to the child process as environment variables. Once P reaches the end of the file (or there are less than 10 coordinates left), P will collect the outputs of the calculations of its children processes. The outputs will take place in an output file (denoted by outputfilepath) and will be in the form of one matrix from each R_i. Then P will calculate the two matrices that are closest to each other with respect to the Frobenius norm, and print them on stdout and terminate.

- The process P must not start collecting the outputs before all the children processes have finished their calculations.
- The children processes must not access the output file simultaneously for writing.

```
./processP -i inputFilePath -o outputFilePath
```

Process R_i
Each process R_i will receive 10 3d integer coordinates as environment variables. The names of the variables are up to you. It will then calculate their covariance matrix and write it to the output file provided to P (the format is up to you). This means you need to pass the outputfilepath information to the child process from P, I recommend passing it as a commandline argument while creating R_i. Then R_i will terminate.

- If P receives SIGINT, it will forward the signal to all of its children, free all of its resources, close open files, and remove the output file, clean-up after its children and terminate. When the children receive the forwarded signal, they will terminate after freeing their resources as well. Be careful with the signal handler, remember what I told you during the lectures. Use the polling based paradigm I showed you for graceful exiting.

Output
```
Process P reading hw2/inputFile.dat
Created R_1 with (66,77,81),(110,91,34),...,(63,90,91)
...
Created R_34 with (61,17,11),(210,51,54),...,(33,60,75)
Reached EOF, collecting outputs from hw2/outputFile.dat
The closest 2 matrices are ... and ..., and their distance is 0.5.
```

Evaluation rules
1) Compilation error: grade set to 1; if the error is resolved during the demo, then evaluation continues.
2) Compilation warning (with respect to the **-Wall** flag); -1 for every warning until 10. -20 points if there are more than 10 warnings; no chance of correction at demo.
3) No makefile: -30

4) No pdf report submitted (or submitted but insufficient, e.g. 3 lines of text with no design explanation, etc): -20. Other file formats are not admissible.
5) If the required command line arguments are missing/invalid, your program must print usage information and exit. Otherwise: -10
6) The program crashes or doesn't produce expected output with normal input: -100
7) The program doesn't satisfy a requirement: -100
8) Presence of memory leak (regardless of amount – checked with valgrind) -30
9) Late submissions will not be accepted
10) In case of an arbitrary error, exit by printing to stderr a nicely formatted informative message. Otherwise: -10
11) Cleanup after the children processes, no zombie processes, -50 otherwise.
12) All file I/O must be done using system calls. Take precautions against signals interrupting your operations. -30 otherwise.

Is my homework submission valid?
If P reads the input, creates the children processes and passes them the proper information, then yes.

Submission rules:

- Your source files, your makefile and a report; place them all in a directory with your student number as its name, and zip the directory.
- Your report must contain: how you solved this problem, your design decisions, which requirements you achieved and which you have failed.
- The report must be in English.
- Your makefile must only compile the program, not run it!
- Do not submit any binary executable files. The TAs will compile them on their own boxes.
- Your code will be compared against online sources; you are free to be inspired, but you are also expected to write your own code.
- Homeworks are invidual tasks. Proven cases of plagiarism will be punished to the full extent.
- Calculations will be accurate down to 3 decimals. You can use the standard math library.

Hints:

- Plan/design carefully.
- Check for memory leaks with valgrind before submitting.
- Compile and run your program on more than one POSIX systems to ensure portability, if you can.
- Control whether you satisfy each and every one of the requirements prior to submission.

Good luck.