CS570 Summer 2019 Assignment 3

This page last modified 27 June, 2019

Design and implement an event management program (emp).

You shall create a program that uses signals to coordinate work among sibling threads. You will build **one thread to be the signal catcher/handler** and **worker threads to perform the following** (note, you will also **use semaphores as needed**).

- Your program shall read the arguments provided by the user. If arguments are not provided use default values as stated below.
- Your program shall create worker threads from the main, initial thread which busy-waits until all worker threads have completed, then, perform a clean exit.
- One of the **worker thread**s shall manage/monitor the **countdown timer**, upon reaching the specified number of seconds (after the number of seconds has elapsed) it shall generate a signal for the program to terminate (all threads, then the process). Use a signal catcher (running in it's own thread) to handle all signals, upon receipt of the terminate signal the signal handler will signal all threads to terminate then it should terminate itself and the process should then terminate.
- Another **worker thread** shall implement a "**wall clock**" which prints the time of day in hour, minute, and second format using the 24 hour clock system. This is printed out to the same terminal screen that all threads are using.
- Another **worker thread** shall manage/monitor the **alarm**. When the user specified (or default value) is reached, the thread shall generate a signal to have an Alarm message printed to the terminal (same terminal all threads are using). This alarm only occurs once, at the specified/default time (seconds that have elapsed).
- Once all child threads have terminated, the parent thread (and the process) shall print a friendly message to the user (using stdout), then perform a clean exit.

User provided inputs/defaults (e.g. % emp: 75 1 45) – note, 75 is the 1st parameter, 1 is the 2nd parameter, and 45 is the 3rd parameter. If not parameters are provided by the user, then use the default values provided below:

- The **first parameter** will be time (in seconds) the program will run before it exits, the value for the **countdown timer**.
- The **second parameter** will be 1 or 60, it it's 1 then print out time every second, if 60 then print out time every minute, all other values will be an error and your program shall gracefully exit. This is the **clock print interval**.
- The **third parameter** will be used as an **alarm**; the value is the time (in seconds from when the program starts running) when the alarm worker thread shall schedule a signal to occur and prints an "alarm" message (you may design the alarm message/statement)
- If just emp is entered (no parameters), use the following **default values: 32, 1, 17**

Your project shall include a README file using the same conventions/requirements specified in the course README instructions file.

Your program will be tested by compiling and executing on edoras. Your program shall be written such that it compiles and executes cleanly when using gcc/g++. Note - you must use a Makefile. You shall create a sub-directory named "a3" in your home directory. In it, you shall place all of your project files, including your Makefile. Your source files shall contain sufficient comments for making the source easy to read. Points will be taken off for poorly (or non) commented source. Name the executable "emp". Also, create an archive file (tarball, zip) and upload to Blackboard (one student per project).

- Create ~/a3 by hand.
- Create all necessary project files. Put them into ~/a3.
- The Makefile shall create an executable named "emp" in this same directory (~/a3).
- The system call "system()" will NOT be allowed
- Threads will use signals and a signal catcher (in it's own thread) for communication/synchronization between threads as specified above (no shortcuts)
- You must work individually or in pairs (individually or a team of 2 students)
- You may use gcc, or g++ compiler on this assignment

The assignment is due by 1800 on Sunday, 7 Jul 2019. This supercedes the specified date/time posted in the course syllabus

TURNING IN YOUR WORK:
Follow the turn-in procedures on class Blackboard.