Name: Gustavo Hammerschmidt.

CS433 Written Homework 2

(60 points) All question numbers refer to exercises on the textbook (10th edition). Make sure you use the right textbook and answer right questions. Students must finish written questions individually. Type your answers and necessary steps clearly.

1. **(5 points) 3.11** Including the initial parent process, how many processes are created by the program shown in Figure 3.32? Explain your answer.

   In total, 16 processes are created, including the parent process. The program executes with only one process until it reaches the for loop and fork itself three times. By forking itself, at every time it executes another fork() system call, the program creates another process for every existing process, therefore, the parent process will also be forked during the execution of the for loop, that will lead to a number of processes that is 2 raised to the times the processes were forked.

2. **(5 points) 3.12** Explain the circumstances under which the line of code marked `printf("LINE J")` in following Figure E3.33 will be reached.

   The line of code will be reached only when the execlp() system call fails to complete and the fork() system call was completed with success.

3. **(5 points) 3.13** Using the program in Figure 3.34, identify the values of pid at lines A, B, C, and D. (Assume that the actual pids of the parent and child are 2600 and 2603, respectively.)

   Line A pid = 0; output is: child: pid = 0.
   Line B pid = 2603; output is: child: pid = 2603.
   Line C pid = 2603; output is: parent: pid = 2603.
   Line D pid = 2600; output is: parent: pid = 2600.

4. **(5 points) 3.16** Using the program shown in Figure 3.35, explain what the output will be at lines X and Y.

In the example given, two processes are created, therefore, each of them has its own global variables, so the output in the X line will be: CHILD: 0 \nCHILD: -1 \nCHILD: -4 \nCHILD: -9 \nCHILD: -16; and the values outputed on the Y line will be: PARENT: 0 \nPARENT: 1 \nPARENT: 2 \nPARENT: 3 \nPARENT: 4. Both processes will have a copy of the incremental variable, which was defined as 0; they do not share it(the same incrementation variable) because each of them both has its own global variables: a copy of the process with its variables was created, and it has its own memory space. They will also have its own array of values and the values outputed in line X were multiplied by minus its index in the vector.

5. **(10 points) 3.17** What are the benefits and the disadvantages of each of the following? Consider both the system level and the programmer level.

   1. Synchronous and asynchronous communication

      A synchronous communication ables the application to have a rendez-vous communication between its processes or threads. But in a rendez-vous, there may be some excesive unrequired waiting time, for those cases, it is better to use an asynchronous communication.

   2. Automatic and explicit buffering

      In automatic buffering, the memory is automatically managed, giving the impression of infinite memory. If the buffer has a large memory size, some quantity of the memory may not be used. In explicit buffering, the buffer memory size is defined. If the memory is all occupied, then the sender may await available space.

   3. Send by copy and send by reference

      Send by reference enables the user to alter the value directly, it is better to pass large values or objects. Send by copy will not give the user the option to change the original value of the copy, only the copy itself.

   4. Fixed-sized and variable-sized messages

      Fixed-size messages allow the program to use a explicit buffer to handle them; variable-size messages need, if the application uses a explicit buffer, to use the buffer many times to pass a message. But it is hard to define a fixed-size for all messages used in a program sometimes, which makes it more convenient to use a variable-size.

6. **(5 points) 4.10** Which of the following components of program state are shared across threads in a multithreaded process?

   a. Register values

   b. Heap memory

   c. Global variables

   d. Stack memory

Only the heap memory and the global variables of those mentioned above is shared across threads.

7. **(10 points) 4.16** A system with two dual-core processors has four processors available for scheduling. A CPU-intensive application is running on this system. All input is performed at program start-up, when a single file must be opened. Similarly, all output is performed just before the program terminates, when the program results must be written to a single file. Between start-up and termination, the program is entirely CPU-bound. Your task is to improve the performance of this application by multithreading it. The application runs on a system that uses the one-to-one threading model (each user thread maps to a kernel thread).

   - How many threads will you create to perform the input and output? Explain.

     For the input and output, I would create only one thread because both of them need to be synchronized(they are normally sequential) and can't have multiple thread performing on them. So I would have a single thread to read the input and another thread to write the output, creating then 2 threads at total.

   - How many threads will you create for the CPU-intensive portion of the application? Explain.

     I would create a number of thread that is equivalent to the number of processors available, so, in this way, the application can use most of the computer resources with a better performance rate(in this case, it would be four threads), according to amdahl's law. It's worth mention that all these threads would be created on the main execution thread: in the overall, there would be four execution threads to perform the calculations, the input and output threads and the main execution thread responsible for creating all of these threads.

8. **(5 points) 4.17** Consider the following code segment:

   1. How many unique processes are created?

      Six unique processes are created.

   2. How many unique threads are created (not counting the main thread of each process)?

      Two threads are created.

9. **(5 points) 4.19** The program shown in Figure E4.23 uses the Pthreads API. What would be the output from the program at LINE C and LINE P?

The output in Line C would be: CHILD: value = 5.

The output in Line P would be: PARENT: value = 0.

10. **(5 points) 4.20** Consider a multicore system and a multithreaded program written using the many-to-many threading model. Let the number of user-level threads in the program be greater than the number of processing cores in the system. Discuss the performance implications of the following scenarios.

1. The number of kernel threads allocated to the program is less than the number of processing cores.

   In this scenario, the performance of the application would not be the best it could be, because not all of the cores are being applied to a user thread.

2. The number of kernel threads allocated to the program is equal to the number of processing cores.

   In this scenario, all the processing cores are being used by kernel-threads, that are executing user-level threads; this is the best scenario because all processors are being used.

3. The number of kernel threads allocated to the program is greater than the number of processing cores but less than the number of user-level threads.

   In this scenario, all the processing cores are being used by the kernel-threads, that are executing user-level threads; however, if a kernel-level gets compromised by its execution and is blocked, another kernel-level can replace the previously one and evict the processing core to be "freezed" for a while.