Ex. 1 - General Questions

1. **Why would a thread voluntarily release the CPU?**

   Threads in a process cooperate. They are not hostile to one another. If yielding is needed for the good of the application, then a thread will yield. After all, it is usually the same programmer who writes the code for all of them.

2. **What is the biggest advantage/disadvantage of user space threads?**

   The biggest advantage is efficiency. No traps to the kernel are needed to switch threads. The ability of having their own scheduler can also be an important advantage for certain applications.

   The biggest disadvantage is that if one thread blocks, the entire process blocks.

3. **If a multithreaded process forks, a problem occurs if the child gets copies of all the parent’s threads. Suppose that one of the original threads was waiting for keyboard input. Now two threads are waiting for keyboard input, one in each process. Does this problem ever occur in single-threaded processes?**

   No. If a single-threaded process is blocked on the keyboard, it cannot fork.

4. **Many UNIX system calls have no Win32 API equivalents. For each such call, what are the consequences when porting a program from a UNIX system to a Windows system?**

   They are very likely not to run properly.

Ex. 2 - C Programming

See `README.md` and use `Makefile` to compile and test easily.

Ex. 3 - Research on POSIX

The Portable Operating System Interface (POSIX) is an IEEE standard that helps compatibility and portability between operating systems. Theoretically, POSIX compliant source code should be seamlessly portable. In the real world, application transition often runs into system specific issues. But POSIX compliance makes it simpler to port applications which can result in time savings. So developers should get acquainted with the fundamentals of this widely used standard.

POSIX.1-2008 standard deals with four major areas:

1. **Base Definition Volume:** General terms, concepts, and interfaces.
2. **Systems Interfaces Volume:** Definitions of system service functions and subroutines. Also, includes portability, error handling and error recovery.
3. **Shell and Utilities Volume:** Definition of interfaces of any application to command shells and common utility programs.
4. **Rationale Volume:** Contains information and history about added or discarded features and the reasonings of the decisions.