**Homework Assignment #2**

***Due Date: 2/28, 11:59 p.m., please submit a soft copy via Blackboard. Late submissions are accepted till 3/2, 11:59 p.m., with 10% penalty each day.***

***Please name your submission file starting with “LastName\_FirstName\_HW2”.***

**Q1.** What is a process in an operating system? What is a thread in an operating system?

The process is an instance of program execution which associated with resources. On the other hand, the thread is the minimal unit that the OS allocates, which means one process can have many threads, and each thread can fetch and execute instructions independently.

The different process has different resources and different executions, where different thread in the same process shares the same resources and different executions.

**Q2. Modeling multiprogramming:** (a) Assume the I/O fraction time of all processes is 20%, and assume processes are independent from each other, what’s the CPU utilization, if the number of processes, *n* = 1, 2, 4, and 8, respectively? (b) If the I/O fraction time is 50% for all processes, what’s the CPU utilization again, if the number of processes, *n* = 1, 2, 4, and 8, respectively?

**Q3.** Consider a system that has two CPUs and each CPU has two hardware threads (hyperthreading). Suppose three processes, P0, Pl, and P2, are started with run times of 5, 10 and 20 msec, respectively. How long will it take to complete the execution of these processes? **Please discuss all possibilities** depending on different processes scheduled to run on different CPUs/threads, and **what is the minimum execution time**? Assume that all three processes are 100% CPU bound, do not block during execution, and do not change CPUs once assigned.

**Q4 (Problem 12).** In Fig. 2-8, a multithreaded Web server is shown. If the only way to read from a file is the normal blocking read system call, do you think user-level threads or kernel-level threads are being used for the Web server? Why?

**Q5 (Problem 27).** In a system with threads, is there one stack per thread or one stack per process when user-level threads are used? What about when kernel-level threads are used? Please explain.

**Q6** Please briefly discuss the advantages and disadvantages of implementing threads in user space and kernel space, respectively.

**Q7 (Problem 24).** Does Peterson's solution to the mutual exclusion problem shown in Fig. 2-24 work when process scheduling is preemptive? How about when it is non-preemptive?

**Q8 (Problem 43).** Measurements of a certain system have shown that the average process runs for a time T before blocking on I/O. A process switch requires a time S, which is effectively wasted (overhead). For round-robin scheduling with quantum Q, give a formula for the CPU efficiency, defined as the useful CPU time divided by the total CPU time including the overhead, for each of the following:

1. Q = ∞
2. Q > T
3. S< Q< T
4. Q = S< T
5. Q nearly 0

**Q9 (Modified Problem 45).** Five batch jobs A through E, arrive at a computer center at almost the same time. They have estimated running times of 10, 4, 2, 6, and 8 minutes. Their (externally determined) priorities are 3, 5, 4, 2, and 1, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the average process turnaround time. Ignore process switching overhead.

1. Round robin.
2. Priority scheduling.
3. First-come, first-served (run in order 10, 4, 2, 6, 8).
4. Shortest job first.

For (a), assume that the system is multiprogrammed, and that each job gets its fair share of the CPU. For (b) through (d) assume that only one job at a time runs, until it finishes. All jobs are completely CPU bound.

**Q10 (Problem 55).** Consider the procedure *put\_forks* in Fig. 2-47. Suppose that the variable *state*[*i*] was set to *THINKING* *after* the two calls to *test*, rather than *before*. How would this change affect the solution?

THE END.