Ceng 328 Operating Systems
Midterm
April 6, 2011 10.40–12.30
Good Luck!

Answer all the questions.

1. (40 pts) Choose only **five** questions. Each questions 8 pts.

   i Discuss and compare

      a Single Programming

      b Pure Multiprogramming

      c Multiprogramming

   ii You are supposed to design a highly reliable operating system. What is meant by "reliable"? Which criteria and measures should be taken care?

   iii What are the two main purposes of an operating system?

   iv An operating system runs in privileged mode, a hardware state where it has full access to machine resources. Why is such a mode needed, and why can't normal user processes and threads enter privileged mode?

   v What are the advantages of using a higher-level language to implement an operating system?

   vi What are the differences between a trap and an interrupt? What is the use of each function?

   vii What is the purpose of system calls?

   viii Explain the difference between an I/O-bound process and a CPU-bound process.

   ix What is an "atomic" operation? Give an example.

2. (10 pts) What advantages do threads have over multiple processes? Suggest one application that would benefit from the use of threads?

3. (10 pts) Why is the separation of mechanism and policy a desirable principle?

4. (10 pts) Three jobs (A, B, and C) arrive to the job scheduler at time 0. Job A needs 15 seconds of CPU time, Job B needs 25 seconds, and Job C needs 35 seconds.

   i What is the average turnaround time for the jobs, assuming a shortest-job-first (SJF) scheduling policy?

   ii What is the average turnaround time assuming a longest-job-first (LJF) policy?

   iii Which finishes first, Job C in SJF or Job A in LJF?

5. (15 pts) Describe the round-robin scheduling algorithm. Discuss what issues need to be consider to achieve good performance from this algorithm.

6. (10 pts) What is a "Critical Region (Section)"? List and explain the conditions that need to be satisfied to solve the critical region problem?

7. (15 pts) Describe the "Strict Alternation" as a solution for mutual exclusion (see the figure)?

```
while (TRUE) {
    while (turn != 0)        /* loop */ ;
    critical_region( );
    turn = 1;
    noncritical_region( );
}
```

(a)

```
while (TRUE) {
    while (turn != 1)        /* loop */ ;
    critical_region( );
    turn = 0;
    noncritical_region( );
}
```

(b)

Figure 1: Strict alternation for achieving mutual exclusion, (a) process0 (b) process1.