Ceng 328 Operating Systems
Final
June 1, 2010 15.00–17.00
Good Luck!

You are allowed to use CALCULATOR.

No any other electronic equipment is allowed.

Answer all the questions.

1. (15 pts)

   i What is time sharing?

   ii Describe the concept of the multiprogramming level.

   iii What is meant by the term *context switch*? What might cause a context switch to occur?

2. (40 pts) Choose only **four** questions.

   i Explain why an operating system can be viewed as a resource allocator.

   ii Describe the difference between the $fork()$ and $clone()$ Linux system calls.

   iii 1 What is a race condition?

   2 Suppose that we have a message-passing system using mailboxes. When sending to a full mailbox or trying to receive from an empty one, a process does not block. Instead, it gets an error code back. The process responds to the error code by just trying again, over and over, until it succeeds. Does this scheme lead to race conditions?

   iv Describe the dining-philosophers problem and how it relates to operating systems.

   v For deadlock to occur, four conditions must hold: mutual exclusion, hold and wait, no preemption, and circular wait. If any one condition does not hold, no deadlock can occur. Assume we want to allow preemption , and thus get out of deadlocks; in other words, if a deadlock is detected, we will forcibly take a lock away from a thread; by repeatedly doing this, we will eventually undo the deadlock. What new problems are introduced by this preemptive approach?

   vi What causes a page fault? What actions may be taken by the operating system when servicing a page fault?

   vii Describe internal and external fragmentation and explain the difference between them. Which one(s) occurs in paging systems? Which one(s) occurs in systems using **pure** segmentation?

3. (15 pts) Explain the UNIX index node (inode) structure in detail.

4. (20 pts) Consider the following sets of processes, with the length of the CPU-burst time given in milliseconds. Arrival time is the time at which the process is added to the ready queue in the order P1, P2, P3, P4, P5.

Process	**Burst Time**	**Priority**	**Arrival Time**
P1	1	1	0
P2	2	3	0
P3	8	3	0
P4	1	4	0
P5	5	2	0

A smaller priority number implies a higher priority. Scheduling algorithms:

- Round Robin (RR); assume that the system is multiprogrammed, and that each job gets its fair share of the CPU.
- Priority Scheduling (PS); assume that only one job at a time runs, until it finishes.
- First Come First Served (FCFS); assume that only one job at a time runs, until it finishes.

i Draw appropriate charts illustrating the execution of these processes using RR, PS, FCFS. Ignore process switching overhead.

ii Calculate the wait (W) and turnaround (T) times of each process for each strategy. Also calculate the average wait and turnaround times under each strategy. Fill the table below

Process	**RR** W Time	**RR** T Time	**PS** W Time	**PS** T Time	**FCFS** W Time	**FCFS** T Time
P1						
P2						
P3						
P4						
P5						
P6						
Average						

iii Discuss which strategy is the best according to the table.

5. (20 pts) Estimated total average access time in disks is given as the following formula

$$T_a = T_s + \frac{1}{2r} + \frac{b}{rN}$$

The following values are given;
Average Seek Time: 2 ms.
6000 rpm (rpm: revolution per minute).
A millisecond (ms) is a thousandth (1/1000) of a second.
Sectors per Track: 64.
Logical Block Size: 512 bytes.
File Size: 20972032 bytes.

i Describe each term in the formula.

ii Fill the table for the file stored compactly (adjacent tracks).

Average seek	
Rot. Delay	
Read ____ sectors	
Total	
All Sectors	

iii Fill the table for the file sectors distributed randomly over the disk.

Average seek	
Rot. Delay	
Read ____ sectors	
Total	
All Sectors	

6. (10 pts) What is "Mutual Exclusion"? 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.