

Name, SURNAME and ID ⇒

Middle East Technical University
Department of Computer Engineering



CENG 334

Section 1 and 2
Spring '2011-2012

Final

- **Duration:** 120 minutes.
- **Exam:**
 - This is a **closed book, closed notes** exam. No attempts of cheating will be tolerated. In case such attempts are observed, the students who took part in the act will be prosecuted. The legal code states that students who are found guilty of cheating shall be expelled from the university for a **minimum of one semester!**
- **About the exam questions:**
 - The points assigned for each question are shown in parenthesis next to the question.
 - Wherever available, use the boxes to write down your answers.
- **This booklet consists of 8 pages including this page. Check that you have them all!**
- **GOOD LUCK !**

Question 1

Question 2

Question 3

Question 4

Question 5

Question 6

Total ⇒

1 (10 pts)



Answer the following questions as (T) rue or (F)alse in the table below. Each correct answer will be awarded with 1 points and each wrong answer will be punished with -1 points

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

- (1) The code that changes the system clock runs in user mode.
- (2) A thread can be blocked on multiple condition variables simultaneously.
- (3) Round-robin scheduling policy is non-preemptive.
- (4) Without kernel-level support, when a user-level thread blocks, it will block all the other threads in the same process.
- (5) The use of a translation look-aside buffer (TLB) for a paging memory system eliminates the need for keeping a page table in memory.
- (6) Consider the storing of the free space on a disk using free blocks. The number of *free blocks* required increases with the amount of data put on the disk.
- (7) The Least Recently Used (LRU) algorithm used for paging out pages out of memory is slower than the Second Chance (Clock) algorithm.
- (8) A cyclic dependency in the resource allocation graph **always** leads to deadlock.
- (9) A smaller page size leads to smaller page tables.
- (10) In a multi-processor system running a symmetric multi-processing (SMP) OS, each CPU runs its own copy of the OS and do not share kernel data structures.

2 (30 pts)



This problem is about the Ring bus service in the university. Riders come to a bus stop and wait for a bus. When the bus arrives, all the waiting riders invoke `boardBus`, but anyone who arrives while the bus is boarding has to wait for the next bus. The capacity of the bus is 50 people; if there are more than 50 people waiting, some will have to wait for the next bus. When all the waiting riders have boarded, the bus can invoke `depart`. If the bus arrives when there are no riders, it should depart immediately.

Write synchronization code using semaphores that enforces all of these constraints.

You can declare and initialize semaphores such as `sem = Semaphore(1);` and use them by calling two methods as `sem.up();` or `sem.down();`.

```
// feel free to make more declarations
int riders = 0; // keeps track of how many riders are waiting
mutex = Semaphore(1); // protects riders
multiplex = Semaphore(50); //makes sure there are no more
                        //than 50 riders in the boarding area
bus = Semaphore(0); // Riders wait on the bus which gets signaled
                    //when the bus arrives
allAboard = Semaphore(0); // The bus waits on allAboard, which gets
                        //signaled by the last student to board.
```

```
void bus(){
```

```
    }
    void riders(){
```

```
    }
```

3 (20 pts)



Answer the following questions about a computer system that has virtual memory implemented using paging.

- (a) (4 pts) Assume that the computer system has a virtual address space of size 2^V bytes and running 2^M bytes of physical memory with a page size of 2^p bytes. If the paging is implemented in single-level (as opposed to multi-level), what would be the minimum size of the page table of a single process in bytes. Your answer should be in terms of V , M and p , and you should consider/comment on what is stored in Page Table Entries.

- (b) (16 pts) Consider the following address space of processes, and comment on where pages from different sections are paged in and paged out from.

Section name and description	Paged in from:	Paged out to:
Text (also known as code section)	Executable on disk.	None.
Initialized variables (such as int c=5;) Unmodified.		
Initialized variables (such as int c=5;) Modified. (e.g. after the execution of c=7; in code)		
Uninitialized variables (such as int d;) Unmodified.		
Uninitialized variables (such as int d;) Modified. (e.g. after the execution of d=3; in code).		
Heap. Initially unmodified.		
Heap. Modified.		
Stack Initially unmodified.		
Stack Modified.		

4 (10 pts)



Here is a table of processes and their associated arrival and running times.

Process ID	Arrival time	CPU running time
Process 1	0	5
Process 2	4	5
Process 3	2	1
Process 4	7	2
Process 5	8	1

- Show the scheduling order for these processes under 3 policies: First Come First Serve (FCFS), Shortest-Remaining-Time-First (SRTF), Round-Robin (RR) with timeslice quantum = 1. Assume that context switch overhead is 0 and that new RR processes are added to the head of the queue and new FCFS processes are added to the tail of the queue.

Time slot	FCFS	SRTF	RR
0	1	1	1
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

- For each process in each schedule above, indicate the queue wait time. Note that wait time is the total time spend waiting in queue (all the time in which the task is not running).

Scheduler	Process 1	Process 2	Process 3	Process 4	Process 5
FCFS wait					
SRTF wait					
RR wait					

- Assume that we could have an oracle perform the best possible scheduling to reduce average wait time. What would be the optimal average wait time, and which of the above three schedulers would come closest to optimal?

5 (25 pts)



Consider a UNIX-like server with a file system where the disk block size is 1kB, and an inode takes 128 bytes. Disk addresses take 32 bits, and the inode contains space for 8 direct addresses, one indirect, one double-indirect and one triple- indirect (the rest of the space in the inode is taken up with other information such as ownership and protection).

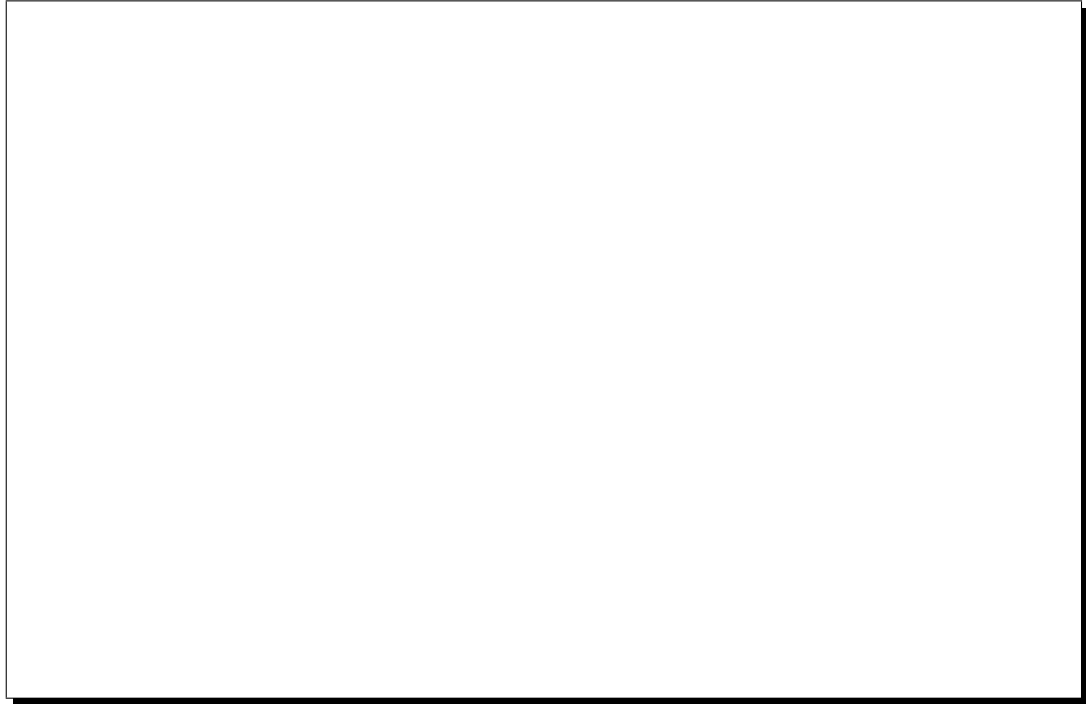
- (a) (10 pts) Saint Lightning, the administrator of the server, makes a last-moment attempt to delete a file `/fb/2011/matchfixing.txt` at the moment the police enters his office. The police turns down the power, before the deletion is fully executed on the server, leaving the filesystem in an inconsistent state. List the full list of disk accesses needed to complete the deletion operation. Assuming that only one disk access (write down which one you've assumed) was successful before the power was down, comment on whether you can recover the file data or not, and if so, how the recovery can be made.

A large, empty rectangular box with a thin black border, intended for the student to write their answer to the question. It occupies the lower half of the page.

- (b) (5 pts) After the server is returned back, Alex, the user, decides to clean up his account on the server by deleting the files `championship-2011.mpg` with size 1025 bytes long, and `championship-2012.mpg` with size 1048576 (1MB) bytes. How much space (including overhead) will be freed as a result of these deletions. Hint: it may help if you draw a picture of how inodes are used to locate the blocks making up a file.



- (c) (10 pts) Assuming that ONLY the inode and data block of the root directory are in memory, write down the sequence of all disk accesses to read the LAST byte of `/alex/fb/championship-2011.mpg`. Hint: For simplicity, assume that all directory tables fit into a single disk block.



6 (10 pts)



- What is deadlock? What is starvation? How do they differ from each other?

- What is a device driver? What is a device controller?

- Write down three reasons why we may need buffering in dealing with I/O devices.

- What's VFS? What changes happen in VFS structures, when a new filesystem is mounted?

- What happens when we open a file? Explain in terms of OS data structures involved.