Name, SURNAME and ID ⇒

Middle East Technical University
Department of Computer Engineering

CENG 334

Section 2 and 3
Spring '2010-2011

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.
  - Whereever 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 (30 pts)

Answer the following questions as (T)rue or (F)alse in the table below. Each correct answer will be awarded with 1.5 points and each wrong answer will be punished with -1.5 points

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

(1) Context switch between the threads of a process can be handled at user-level.

(2) In modern UNIX systems, during the context switch, the OS saves the whole address space of a process into the swap disk.

(3) The Translation Lookaside Buffer caches the mapping from virtual addresses to physical addresses.

(4) In a single-level paging virtual memory system, the total number of Page Table Entries are doubled if we double the amount of the physical memory in the computer.

(5) In a single-level paging virtual memory system, the size of a Page Table Entry remains the same even if the amount of the virtual memory is changed.

(6) Round-robin scheduling policy is non-preemptive.

(7) Race conditions happen only when multiple threads (or processes) tries to read a shared variable.

(8) In a single-level paging virtual memory system, internal fragmentation cannot occur.

(9) Threads within the same process share the same stack.

(10) In a Mesa-style monitor, the thread that signal()'s to wake up other sleeping threads is immediately blocked.

(11) Priority inversion can happen with user-level threads.

(12) In a FAT-like filesystem the largest file that is supported is half of the partition.

(13) 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.

(14) The Least Recently Used (LRU) algorithm used for paging out pages out of memory is slower than the Second Chance (Clock) algorithm.

(15) A cyclic dependency in the resource allocation graph always leads to deadlock.

(16) The "code section" of a process executable is never paged-out to a swap area.

(17) In Unix-derived file systems, the name and attributes of a file are stored in the directory structure.

(18) Part of a device driver responds to interrupts from the device

(19) The clock algorithm provides an approximation to a least-recently used (LRU) page replacement mechanism.

(20) 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 (20 pts)

Three kinds of threads share access to a singly-linked list: searchers, inserters and deleters. Searchers merely examine the list; hence they can execute concurrently with each other. Inserters add new items to the end of the list; insertions must be mutually exclusive to preclude two inserters from inserting new items at about the same time. However, one insert can proceed in parallel with any number of searches. Finally, deleters remove items from anywhere in the list. At most one deleter process can access the list at a time, and deletion must also be mutually exclusive with searches and insertions.

Write code for searchers, inserters and deleters that enforces this kind of three-way categorical mutual exclusion.

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
insertMutex = semaphore(1); // ensures that one inserter active at a time
noSearcher = semaphore(1); // indicates that there are no active searchers.
noInserter = semaphore(1); // indicates that there are no active inserters.



void searcher(){                                    void deleter{









}
void inserter(){









}                                                   }
```

3 (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	3

- 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?

4 (15 pts)

Consider a computer system which has a virtual memory architecture with the following parameters:

- Virtual addresses are 48 bits.
- The page size is 32K byte.
- The architecture allows a maximum 64 Terabytes (TB) of real memory (RAM).
- The first- and second-level page tables are stored in real memory.
- All page tables can start only on a page boundary.
- Each second-level page table fits exactly in a single page frame.
- There are only valid bits and no other extra permission, or dirty bits.

Draw and label a figure showing how a virtual address gets mapped into a real address. You should list how the various fields of each address are interpreted, including the size in bits of each field, the maximum possible number of entries each table holds, and the maximum possible size in bytes for each table (in bytes). Also, your answer should indicate where checks are made for faults (e.g., invalid addresses).

5 (10 pts)

In the recent UNIX file systems, the disk block size is 1kB, and an inode takes 128 bytes. Disk addresses take 32 bits, and the inode contains space for 64 bytes of data (the first 64 bytes of the file content), 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). How much space (including overhead) do files that are: a) one (1) byte long, b) 1025 bytes long, c) 65536 (64KB) bytes long, and d) 1048576 (1MB) bytes long require? Hint: it may help if you draw a picture of how inodes are used to locate the blocks making up a file.

6 (15 pts)

- Compare the pros and cons of static and dynamic linking.

- Describe what copy-on-write mechanism is, and give an example where it minimizes the memory usage.

- 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 does the setuid bit of a file mean in UNIX like filesystems? Give a example where the use of this bit is essential.