AUB American University of Beirut
الجامعة الأميركية في بيروت

Faculty of Arts & Sciences
Department of Computer Science
CMPS 272—Operating Systems Concepts
Spring 2004–2005

Final Exam

Date: Tuesday, June 7th, 2005 at 11:30 am.
Duration: 120 minutes.
Instructor: Jihad Boulos

ID #: ----------

Section: ----------

You are NOT allowed to have your book and notes with you during this exam.

Your exam should have 9 pages, and there are 9 problems totaling 100 points. Your answers should be concise, and when possible should be a list of important points rather than prose. Solve as many problems as you can. I recommend that you start with problems you think the easiest. I also advise you to spend time on understanding what is being asked by each problem and **budget your time** wisely to be able to solve **all** problems.

Beware, wordy and/or irrelevant answers might reduce your score for that problem. Your answers should be the summary of work done on scratch paper that you do not hand in. Also, do not expect that I will spend much time trying to decipher your hand writing. I will give a ZERO to **illegible** answers. The space allocated for answers should be sufficient. If not, use the back sides of the pages. **Please turn-off your cell-phones.**

Wish you good work.

	Prob. 1	Prob. 2	Prob. 3	Prob. 4	Prob. 5	Prob. 6	Prob. 7	Prob. 8	Prob. 9
Max Grade	8	8	10	12	6	12	12	12	20
Your Grade									

Exercise 1 (8 points): Scheduling

a. Four processes arrive with the following CPU burst values.

	Arrival time	*CPU Burst*
P_1	7	2
P_2	2	2
P_3	3	5
P_4	0	8

What is the average waiting time with preemptive SJF. Ignore context switching overhead.

b. For process P_4, we would like to predict the value of the next CPU burst. Remember that t_n is defined as the actual length of the n^{th} CPU burst, and τ_{n+1} is defined as the predicted value for the next CPU burst. α, $0 \le \alpha \le 1$, is the weight given to the actual CPU burst versus the predicted one.
Remember also that:

$$\tau_{n+1} = \alpha\, t_n + (1 - \alpha)\, \tau_n$$

The first predicted CPU burst for P_4 was 12 time units. For $\alpha = 0.7$, what is the next predicted burst?

Exercise 2 (8 points): Address Translation

Compute the required TLB (Translation Look-aside Buffer) hit ratio to achieve the average memory access time of 125 ns given the following system specification.

i. TLB access time = 20 ns;
ii. Memory access time = 100 ns.

Exercise 3 (10 points): EAT

Consider a demand-paging system with a paging disk that has an average access and transfer time of 20 milliseconds. Addresses are translated through a page table in main memory, with an access time of 1 microsecond per memory access. Thus, each memory reference through the page table takes two accesses. To improve this time we have added an associative memory that reduces access time to one memory reference if the page table entry is in the associative memory. Assume that 80 percent of the accesses are in the associative memory and that of the remaining, 10 percent (or 2 percent of the total) cause page faults. What is the effective memory access time (EAT)?

Exercise 4 (12 points): Virtual Memory

Consider a virtual memory system with 34-bit addresses. The first 23 bits are used as a page number, and the rest is the offset.

(a) How many Kilobytes (KB) is a single page frame?

(b) Assuming the single-level paging and each page-table entry to be 4 bytes, how many Megabytes of memory is needed to store the page table?

(c) Suppose the system uses two-level paging, with first n bits ($n > 11$) of the address used as an index into the first-level page table. Assume that a typical program uses 8MB memory. Write an expression in terms of n that gives the number of second-level page-tables used by the typical program.

Exercise 5 (6 points): File System

"tail –n 10 file1" prints the last 10 lines of file1 on the screen. This command often has a better performance than just open the file with an editor and jump to the end if the file is very large. Explain why this is the case. Is there any system in which such a command wouldn't have any performance advantages?

Exercise 6 (12 points): File System

Suppose that you have a Unix file system where the disk block size is 1kB, and an i-node takes 128 bytes. Disk addresses take 32 bits, and the i-node contains space for 64 bytes of data, 8 direct addresses, one indirect, one double-indirect and one triple-indirect (the rest of the space in the i-node is taken up by other information, such as ownership and protection). An index block is the same size as a disk block.
How much space (including overhead) is needed by files that are:

1. one (1) byte long,
2. 1025 bytes long,
3. 65536 (64KB) bytes long, and
4. 1048576 (1MB) bytes long require?

Hint: it may help if you draw a picture of how i-nodes are used to locate the blocks making up a file.

Exercise 7 (12 points): Resource Management

N processes share *M* resource units that can be reserved and released only one at a time. The maximum need of each process does not exceed *M* and the sum of all maximum needs is less than $N + M$. Show that a deadlock cannot occur.

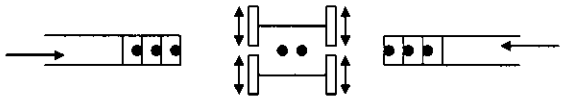
Exercise 8 (12 points): Disk Scheduling

A disk has 200 cylinders. The head is currently over cylinder 20 heading outwards (towards higher numbered cylinders). A time t = 0 we have a queue of requests for blocks on different cylinders: 25, 36, 28, 124, 56, 18, 178, 23. At times t = 129, a new request arrives for cylinder 77; at time t = 251, another request arrives for cylinder 198; finally, at time t = 311, a request arrives for cylinder 12. List the order in which the cylinders will be visited using:

1. SCAN
2. C-Look
3. If we assume moving X cylinders required X units of time, then how long will it take to visit all cylinders using SSTF?

Exercise 9 (20 points): Synchronization

In this problem, you are supposed to synchronize the sliding door to a certain building. The door is pictured below. It is made of two parts (2 sliding doors) where people coming from one direction go through the first door that opens automatically, and when they are inside the hall, the door on their back closes, and the one in front of them opens. Every door closes after one person passes through it. Using semaphores and their atomic Wait() and Signal() operators, your job is to keep people flowing in both directions through the doors. Here are the constraints:



1. At most 2 people can be inside the doorway between the two automatic doors at any time.
2. People wait in line on either side of the door until an empty space inside the doorway is available. A person must exit the doorway before the next person can enter it.
3. If people are waiting on both sides of the doorway, then one person from each direction should move through the door at each turn.
4. You have available to you (already written) procedures called **EnterDoorWay**() and **LeaveDoorWay**(), which moves a person calling this routine into and out of the doorway. You have no idea how long these procedures take to execute.
5. When your program starts, assume that the doorway is completely empty and that both sliding doors are closed. Also, assume that no one is waiting in line on both sides.

You are to write the code for **EnterBuilding**() and **LeaveBuilding**() procedures. These procedures are called by a person (process) when (s)he (it) wants to enter or leave the building. These procedures will return when the person (process) has safely entered or exited. You also have to write the code for any procedure that **EnterBuilding**() and **LeaveBuilding**() may need to call. When you use a semaphore, specify whether it is a binary or a counting semaphore, and its initial value.