University of California, Berkeley
College of Engineering
Computer Science Division – EECS

Spring 2004 Anthony D. Joseph

Midterm Exam

March 18, 2004
CS162 Operating Systems

Your Name:	
SID AND 162 Login:	
TA:	
Discussion Section:	

General Information:
This is a **closed book and notes** examination. You have 120 minutes to answer as many questions as possible. The number in parentheses at the beginning of each question indicates the number of points given to the question; there are 100 points in all. You should read **all** of the questions before starting the exam, as some of the questions are substantially more time consuming.

Write all of your answers directly on this paper. ***Make your answers as concise as possible.*** If there is something in a question that you believe is open to interpretation, then please ask us about it!

Good Luck!!

Problem	Possible	Score
1	17	
2	16	
3	27	
4	22	
5	18	
Total	**100**	

1. (17 points total) Short answer questions:
   a. (12 points) True/False and Why?
      i) (4 points) Because of the overhead of context switching, programs that use threads will always take longer to execute than programs that do not use threads.

      TRUE FALSE
      Why?

      ii) (4 points) Protection and isolation between applications and between applications and the operating system can be provided without hardware support.

      TRUE FALSE
      Why?

      iii) (4 points) After a UNIX fork operation, the parent and new child process are identical in all respects.

      TRUE FALSE
      Why?

   b. (5 points) Consider a system with a mixture of I/O bound processes and CPU bound processes
      i) (3 points) Explain how this mixture of processes maximizes system utilization:

      ii) (2 points) Explain why this combination is more important in batch systems than it is on most computers sitting around in our department:

2. (16 points total) CPU Scheduling.

a. (6 points) 5 identical jobs are run once on a non-preemptive scheduler, and then again on a preemptive scheduler using a round robin scheduling discipline. The jobs are purely computational - they do almost no I/O. On the non-preemptive scheduler it takes 24 hours for all 5 of them to complete; on the preemptive one it takes 24 hours and 2 minutes for all 5 of them. If the time quantum of the preemptive scheduler is 0.05 sec (50 milliseconds), how long does a context switch take? (Show **all** your work; it's OK to leave the answer in symbolic / long form).

b. (4 points) Name 2 things that need to be saved on a context switch.

c. (6 points) Suppose that a scheduling algorithm (at the level of short-term CPU scheduling) favors those processes that have used the least processor time in the recent past. Why will this algorithm favor I/O-bound processes and yet not permanently starve CPU-bound processes?

No Credit – **Problem X** (000000000000 points)

The following is an excerpt from The Washington Post's Style Invitational invitation to readers to: take any word from the dictionary, alter it by adding, subtracting, or changing one letter, and supply a new definition.

Here are this year's winners:

1. Reintarnation: Coming back to life as a hillbilly.

2. Bozone (*n.*): The substance surrounding stupid people that stops bright ideas from penetrating. The bozone layer, unfortunately, shows little sign of breaking down in the near future.

3. Giraffiti: Vandalism spray-painted very, very high.

4. Sarchasm: The gulf between the author of sarcastic wit and the person who doesn't get it.

5. Inoculatte: To take coffee intravenously when you are running late.

6. Hipatitis: Terminal coolness.

7. Osteopornosis: A degenerate disease. (This one got extra credit.)

8. Karmageddon: It's like, when everybody is sending off all these really bad vibes, right? And then, like, the Earth explodes and it's like, wow, a serious bummer.

9. Decafalon (*n.*): The grueling event of getting through the day consuming only things that are good for you.

10. Glibido: All talk and no action.

11. Dopeler effect: The tendency of stupid ideas to seem smarter when they come at you rapidly.

12. Arachnoleptic fit (*n.*): The frantic dance performed just after you've accidentally walked through a spider web.

13. Beelzebug (*n.*): Satan in the form of a mosquito that gets into your bedroom at three in the morning and cannot be cast out.

And the pick of the literature:
14. Caterpallor (*n.*): The color you turn after finding half a grub in the fruit you're eating.

3. (27 points total) Concurrency problem: Implementing Semaphores.

You are programming on a multiprocessor system using threads. The system includes monitors and condition variables, with the following classes and methods:

```
public class Monitor {
        public Monitor() {
                /* Creates a new monitor */
                …
        }

        public void Enter() {
                /* Enters the monitor */
                …
        }

        public void Exit() {
                /* Exits the monitor */
                …
                }
}
```

```
public class ConditionVariable {
        public ConditionVariable(Monitor mon) {
                /* Creates a condition variable
                   associated with monitor mon */
                …
        }
        public void Wait() {
                /* Blocks on the condition variable */
                …
        }
        public void Notify() {
                /* Wakes up one thread waiting on
                   cv, if there is such a thread */
                …
        }
        public void Broadcast() {
                /* Wakes up all threads waiting on cv,
                   if there are such threads */
                …
        }
}
```

Provide an implementation of general semaphores using this system. In other words, write the file Semaphore.java, implementing the following methods:

```
public class  Semaphore {
        public Semaphore(int initialValue) {
                /* Create and return a semaphore with initial value: initialValue*/
                …
                }
        public P() {
                /* Call P() on the semaphore */
                …
        }
        public V() {
                /* Call V() on the semaphore */
                …
        }
}
```

Write your solution on the following page.

3. (continued) Concurrency problem: Implementing Semaphores.

Write your solution here:

```
public class Semaphore {
```

4. (22 points) Deadlock:

A restaurant would like to serve four dinner parties, P1 through P4. The restaurant has a total of 8 plates and 12 bowls. Assume that each group of diners will stop eating and wait for the waiter to bring a requested item (plate or bowl) to the table when it is required. Assume that the diners don't mind waiting. The maximum request and current allocation tables are shown as follows:

Maximum Request	**Plates**	**Bowls**
P1	7	7
P2	6	10
P3	1	2
P4	2	4

Current Allocation	**Plates**	**Bowls**
P1	2	3
P2	3	5
P3	0	1
P4	1	2

a. (4 points) Determine the Need Matrix for plates and bowls.

Need	**Plates**	**Bowls**
P1		
P2		
P3		
P4		

b. (7 points) Will the restaurant be able to feed all four parties successfully? *Clearly explain your answer – specifically, why no or why/how there is a safe serving order*.

4. (continued) Deadlock

c. (11 points) Assume a new dinner party, P5, comes to the restaurant at this time. Their maximum needs are 5 plates and 3 bowls. Initially, the waiter brings 2 plates to them. In order to be able to feed all five parties successfully, the restaurant needs more plates.

i. (2 points) Determine the new Need Matrix for plates and bowls.

Need	Plates	Bowls
P1		
P2		
P3		
P4		
P5		

ii. (6 points) At least how many plates would the restaurant need to add?

iii. (3 points) Show a safe serving sequence.

5. (18 points) Paging:

Suppose you have a system with 32-bit pointers and 4 megabytes of physical memory that is partitioned into 8192-byte pages. The system uses an Inverted Page Table (IPT). Assume that there is no page sharing between processes.

a. (8 points) Describe what page table entries should look like. Specifically, how many bits should be in each page table entry, and what are they for? Also, how many page table entries should there be in the page table?

b. (5 points) Describe how an IPT is used to translate a virtual address into a physical address.

c. (3 points) How can you make an IPT more efficient? *Explain your solution and how it works in detail*.

d. (2 points) What effect, if any, does your solution in part (c) have on what happens on a context switch?

This page intentionally left blank as scratch space.

Do not write answers on this page

Do not write answers on this page