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

Spring 2012 Anthony D. Joseph and Ion Stoica

Midterm Exam

March 7, 2012
CS162 Operating Systems

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

General Information:
This is a **closed book and one 2-sided handwritten note** examination. You have 80 minutes to answer as many questions as possible. The number in parentheses at the beginning of each question indicates the number of points for that question. 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!!

QUESTION	POINTS ASSIGNED	POINTS OBTAINED
1	20	
2	20	
3	15	
4	20	
5	15	
6	10	
TOTAL	100	

1. (20 points total) Short answer questions.
   a. (8 points) True/False and Why? **CIRCLE YOUR ANSWER.**
      i) A lightweight process with one thread is equivalent to a heavyweight process.

      TRUE FALSE
      Why?

      ii) Demand paging requires the programmer to take specific action to force the operating system to load a particular virtual memory page.

      TRUE FALSE
      Why?

   b. (8 points) Two-level Page Tables:
      i) Give a two to three sentence description of a two-level page table.

      ii) Briefly (2 sentences) state one advantage AND one disadvantage of two-level page tables.

   c. (4 points) List the four requirements for deadlock.

2. (20 points total) Consider the following two functions implementing a producer and consumer by using monitors:

```
void send(item) {
  lock.acquire()
  enqueue(item);
  printf("before signal()\n");
  dataready.signal(&lock);
  printf("after signal()\n");
  lock.release();
}

item = get() {
  lock.acquire();
  while (queue.isEmpty()) {
    printf("before wait()\n");
    dataready.wait(&lock);
    printf("after wait()\n");
  }
  item = dequeue();
  lock.release();
}
```

a. (4 points) Use no more than three sentences to contrast Hoare and Mesa monitors.

b. (5 points) Assume two threads T1 and T2, as follows:

```
    T1                                  T2
send(item);                      item = get();
```

What are the possible outputs if the monitor uses the Hoare implementation?

c. (5 points) Repeat question (b) for a Mesa implementation of the monitor.

d. (6 points) Now assume a third thread T3, i.e.,

```
   T1                    T2                    T3
send(item);         item = get();         send(item);
```

What are the possible outputs if the monitor uses the Hoare implementation? Please specify from which thread does an output come by specifying the thread id in front of the output line, e.g., `[T1] before signal` or `[T2] after wait`.

3. (15 points) Design tradeoffs (15 points total):

You've been hired by Orange Computer to help design a new processor and Orange Pro laptop. After choosing the display, case, and other components, you are left with $460 to spend on the following components:

Item	Latency	Minimum Size	Cost
TLB	10 ns	256 entries	$0.10/entry
Main memory	180 ns	2 GB	$10/GB
Magnetic Disk	8 ms (8M ns)	300 GB	$0.10/GB

The page size is fixed at 64 KB. Assume you want to run up to 20 applications simultaneously. Each application has an overall maximum size of 1 GB and a working set size of 256 MB. TLB entries do not have Process Identifiers. Discuss how you would divide the available funds across the various items to optimize performance.

4. (20 points) Concurrency control: Consider the following pseudocode that aims to implement a solution for the Dining Philosopher problem. Note that a philosopher can use any chopstick.

```
// assume chopstick[i].status = FREE, for 1 <= i <= N
get_chopstick(boolean hold_one_chopstick) {
  lock.acquire();
  for (i = 1; i <= N; i++) {
    if (chopstick[i].status == FREE) {
      chopstick[i].status = BUSY;
      return i;
    }
  }
  lock.release();
  return -1;
}

release_chopstick(i) {
  if (i == -1) return;
  chopstick[i].status = FREE;
}


philosopher() {
  plate = FULL;
  while (plate == FULL) {
    chopstick1 = get_chopstick(FALSE);
    if (chopstick1 != -1) {
      chopstick2 = get_chopstick(TRUE);
      plate = EMPTY;
      release_chopstick(chopstick2);
    }
    release_chopstick(chopstick1);
  }
}

main() {
  for (i = 1; i <= N; i++) {
    thread_fork(philosopher());
  }
}
```

a. (2 points) Name an error in how synchronization primitives are used in `get_chopstick()`

b. (10 points) After fixing the error in part (a), does the program work correctly? If it does not, give a simple example to show how the program fails, and provide a fix. If it does, use no more than three sentences to argue why it works.

c. (8 points) Assume `main()` launches N+1 philosopher threads, instead of N. Will the program work correctly given the changes you made for parts (a) and (b)? If it does not, give a simple example to show how the program fails, and provide a fix. If it does, use no more than three sentences to argue why it works.

5. (15 points total) Scheduling.

a. (15 points) Consider the following processes, arrival times, and CPU processing requirements:

Process Name	Arrival Time	Processing Time
1	0	3
2	1	5
3	3	2
4	9	2

For each scheduling algorithm, fill in the table with the process that is running on the CPU (for timeslice-based algorithms, assume a 1 unit timeslice). For RR and SRTF, assume that an arriving thread is run at the beginning of its arrival time, if the scheduling policy allows it. The turnaround time is defined as the time a process takes to complete after it arrives.

Time	**FIFO**	**RR**	**SRTF**
0	**1**	**1**	**1**
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
Average Turnaround Time			

6. (10 points total) Caching: Assume a computer system employing a cache, where the access time to the main memory is 100 ns, and the access time to the cache is 20ns.

   a. (2 points) Assume the cache hit rate is 95%. What is the average access time?

   b. (2 points) Assume the system implements virtual memory using a two-level page table with no TLB, and assume the CPU loads a word X from main memory. Assume the cache hit rate for the page entries as well as for the data in memory is 95%. What is the average time it takes to load X?

   c. (3 points) Assume the same setting as in point (b), but now assume that page translation is cached in the TLB (the TLB hit rate is 98%), and the access time to the TLB is 16 ns. What is the average access time to X?

   d. (3 points) Assume we increase the cache size. Is it possible that this increase to lead to a decrease in the cache hit rate? Use no more than three sentences to explain your answer.

This page intentionally left blank

Do not write answers on this page