Name:

CSCI 3753: Operating Systems
Spring 2013
Midterm Exam
03/20/2013

Answer all questions in the space provided

Multiple Choice Questions: Choose one option that answers the question best.
[30 Points]

1. Resource manager view of operating systems is

   - A. OS provides mechanisms to deal with the complexity of hardware
   - B. OS provides support for developing applications for a computing system
   - ✓ C. OS allows sharing and effective utilization of computing system resources
   - D. OS provides equivalent of a virtual machine that is easier to use
   - E. OS provides support for security and protection

2. Which of the following statement is FALSE about batch processing and multiprogramming

   - A. Batch jobs are very non-interactive
   - ✓ B. In multiprogramming systems, OS time-multiplexes between executable programs
   - C. In batch processing systems, different programs are executed in a serial order
   - D. In multiprogramming systems, more than one program may reside in memory
   - E. In batch processing systems, OS may schedule another program to run when the running program blocks for I/O

3. Which of the following is FALSE about a bootstrap program

   - A. It initializes registers, main memory, devices, etc
   - B. It runs diagnostics to determine the state of the machine
   - C. It locates the kernel, loads it into main memory, and starts its execution
   - D. It is started when the system is turned on
   - ✓ E. It updates the kernel when a kernel update is received

4. A trap table or a jump table

   - A. stores pointers to system call handler routines
   - B. stores addresses of all interrupt handling routines
   - C. stores process control blocks
   - D. resides in the user space
   - ✓ E. Both A and B

5. Two threads with in a process

    - A. share their stacks
    - B. have their own separate heaps
    - ✓ C. have their own separate program counters
    - D. have their own separate code segment
    - E. cannot have race conditions

6. Which of the following is NOT possible in implementing mutual exclusion using disabling/enabling interrupts?

    - A. A process may get preempted while executing in the critical section
    - B. Two processes may run in their critical sections at the same time if there are multiple processors in the system
    - C. A programmer may forget to enable interrupts after exiting from the critical section
    - D. A process may run indefinitely in its critical section and thus prevent any other process from running
    - E. A process in the critical section may affect the execution of another process outside the critical section

7. Which of the following statement is FALSE about Completely Fair Scheduler (CFS)?

    - A. Decay factor is used to determine the virtual runtime of a process
    - B. Self-balancing B-Tree is used to implement the scheduling policy
    - ✓ C. Scheduling time complexity is constant (O(1))
    - D. As processes use more CPU, they move to the right of the B-Tree
    - E. Priority assignment is simplified compared to the O(1) scheduler

8. Shortest Job First scheduling policy

    - ✓ A. minimizes average wait time of the processes
    - B. minimizes average turnaround time of the processes
    - C. minimizes average response time of the processes
    - D. ensures that all process deadlines are met
    - E. ensures fairness

9. Signals can be used to

    - A. inform processes of unexpected external events
    - B. allow one process to interrupt another process and send it a coded signal
    - C. implement shared-memory IPC
    - D. implement message-based IPC
    - ✓ E. Both A and B, but not C or D

10. Which of the following is FALSE about monitors?

- A. At most one process/thread may be active in a monitor at any time
- B. Monitors can be used to implement mutual exclusion as well as interprocess synchronization
- C. A monitor includes its own data structures, variables, procedures and initialization code
- ✓ D. Any program code implemented using semaphores can also be implemented using monitors
- E. Monitors are low-level synchronization primitives

Short Answer Questions:

1. **[6 Points]** What is a context switch? What is its impact on OS performance?

A context switch is when the CPU switches from one running program to another. Impact on OS performance is larger overhead (copying of registers slows things done) and no useful progress occurs for any application

2. **[6 Points]** Provide a step-by-step description for adding a new device in Linux operating system without requiring recompiling the kernel.

1. Write the device driver, compile it to create an LKM (.ko file)
This LKM contains an initialization routine init_module( ) function that registers
various device functions contained in the LKM with the kernel using appropriate kernel
functions such as register_chrdev, register_blkdev, etc.
This registration fills the entry table in the kernel (dev_func_i[j]) with appropriate
function pointers
2. Call insmod command that gets a unique device number for the new device and
invokes the init_module systems call to load the LKM. This system call invokes the
LKM's initialization routine.

3. **[6 Points]** Draw a process state transition diagram showing the states of a process and the possible transitions from one state to another. Label all edges.

Text See Lecture Set 9

Text

4. **[6 Points]** Consider the following solution for mutual exclusion problem. Explain through an example that this solution may result in two processes accessing the critical section at the same time.

```
boolean lock = FLASE; //shared variable

while (lock);
lock = TRUE;

  // critical section

lock = FLASE;
```

Both processes may enter their critical section if there is a context switch just before the <lock = TRUE> statement
Example:

e

5. **[6 Points]** Consider the following synchronization primitives:

   sleep *value*: block the executing process on *value*
   wakeup *value*: awaken all processes blocked on *value*

   where *value* is an integer.

   Process P1 is to execute statements S1 and S2; process P2 is to execute statements S3 and S4. Statement S2 must be executed after S3. A colleague gives you the following program:

```
P1:                    P2:
  S1;                    S3;
  sleep(1);              wakeup(1);
  S2;                    S4;
```

   Is the solution correct? Explain your answer.

6. **[5 Points]** Give an example to show that some of the goals of a scheduling algorithm may be contradictory.

A scheduler should minimize the respone time for interactive user yet should also minimize the time batch users must wait for an output (turnaround)

Throughtput: A scheduler shoud maximize the number of jobs processed per unit time

Problems

1. **[10 Points]** Draw and label a figure to show the sequence of steps in a *write()* operation to disk, from the application first calling a *write()* through the OS processing the *write()* to the final return from the *write()* call upon completion of the disk operation. Assume interrupt-driven I/O. Your answer should include components such as the device controller, interrupt handler, device handler, device driver and any other OS components you deem appropriate to add.

e

e

Text

Iex

t

2. **[10 Points]** Three non-realtime processes, P1, P2 and P3 are running on a Linux system that is using O(1) scheduler. Assume that the time slice is 5 ms long, context switch time is 1 ms, and the priorities are updated by one place based on CPU or I/O bound nature of the process. Show the Gantt chart for the execution of these three processes and calculate the average turnaround time and average response time.

Process	Initial priority	CPU execution time (ms)	Additional description
P1	8	9	P1 doesn't perform any I/O
P2	13	5	P2 performs I/O once for 3 ms after 1 ms of its execution on CPU
P3	12	8	P3 doesn't perform any I/O

3. **[15 Points]** You have just been hired by Greenpeace to help the environment. Because unscrupulous commercial interests have dangerously lowered the whale population, whales are having synchronization problems in finding a mate. The trick is that in order to have children, *three* whales are needed, one male, one female, and one to play matchmaker --- literally, to push the other two whales together (I am not making this up!). Your job is to write three functions: *Male* ( ), *Female* ( ), and *Matchmaker* ( ). A male whale calls *Male* ( ), which waits until there is a waiting female and a matchmaker. A female whale calls *Female* ( ), which must wait until there is a waiting male and a matchmaker. Similarly, a matchmaker calls *Matchmaker* ( ), which must wait until there is a waiting male and female. Once all three types of whales are present, all three return with one of them printing an appropriate message. Use semaphores to implement the required synchronization.

```
Recall that the semaphore blocks/waits when 0, and falls through to
next line of code when non-zero (note: value of semaphore will always
be positive if not zero). Also, recall that signal()/sem_post()
increments counter by 1 whereas wait()/sem_wait() decrements by 1 with
every call
sem_t Male, Fem, Matcher;
//Initialize semaphores to 0
sem_init(Male,0);
sem_init(Fem,0);
sem_init(Matcher, 0);
male()
{
    signal( Male );
    signal( Male);
    wait(Fem);
    wait(Matcher);
}
female()
{
    signal( Fem);
    signal( Fem);
    wait( Male);
    wait(Matcher)
}
Matcher()
{
    signal( Matcher );
    signal (Matcher);
    wait (Male);
    wait( Fem);
    printf("Match made!")
}
```