King Saud University

College of Computer and Information Sciences
Computer Science Department

Course Code:	CSC 227
Course Title:	Operating Systems
Semester:	Spring 2015
Exercises Cover Sheet:	**Mid 2 Exam**

Duration: 90 min

Student Name:	
Student ID:	
Student Section No.	

Tick the Relevant	Computer Science B.Sc. Program ABET Student Outcomes	Question No. Relevant Is Hyperlinked	Covering %
	knowledge of computing and mathematics appropriate to the discipline;	1	???
	ze a problem, and identify and define the computing requirements appropriate to its solution		
	n, implement and evaluate a computer-based system, process, component, or program to meet desired needs;		
	ion effectively on teams to accomplish a common goal;		
	rstanding of professional, ethical, legal, security, and social issues and responsibilities;		
	unicate effectively with a range of audiences;		
	ze the local and global impact of computing on individuals, organizations and society;		
	gnition of the need for, and an ability to engage in, continuing professional development;		
	rrent techniques, skills, and tools necessary for computing practices.		
	mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;		
	design and development principles in the construction of software systems of varying complexity;		

King Saud University
College of Computer and Information Sciences
CSC 227: Operating Systems

Total Marks: 25
Spring 2015-16
Midterm Exam II
Date: 19-April-2016

Time: 7:00pm – 8:30pm (90 minutes)
Name: ……………………………………………….……………………
ID#: …………………………………………………………………….
Section#:…………………… **or Teacher Name**: ………………………

Instructions:

This exam has 6 pages including the title page.

Do not use pencil.

Write clearly and neatly.

Question 1. [10 marks] Select ONLY ONE ANSWER (the best answer).

Copy your answer for question 1-1 to 1-15 in the table on page2. ONLY THAT TABLE WILL BE GRADED.

1.	The fork system call returns …………….. for the child process:
a.	The PID of the parent
b.	-1
c.	**0**
d.	1

2.	Which of the following component is NOT shared among different threads of the same process:
a.	code
b.	**registers**
c.	data
d.	files

3.	If a thread invokes the exec() system call:
a.	**It will replace the entire process including all threads**
b.	It creates a new process with new threads
d.	It copies existing process
f.	It will rename the existing process

4.	Which of the following state is shared between the parent process and the child process?
a.	Stack
c.	Heap
e.	Shared memory segments
g.	**All the above**

5.	In a deferred cancellation mode what will happen to the thread?
a.	It will be cancelled immediately
b.	**It will continue running until it arrives to cancellation point**
c.	It can ignore the cancellation request and continue running to its end
d.	It will be put in the waiting queue until it receives a new signal

6.	Two types of semaphores are:
a.	adding Semaphores and Binary Semaphores
b.	analog Semaphores and Octal Semaphores
c.	**counting Semaphores and Binary Semaphores**
d.	critical Semaphores and System Semaphores

7.	Parent may terminate the execution of children processes because of:
a.	the child has exceeded allocated resources
b.	the task assigned to child is no longer required
c.	the parent is exiting and the OS does not allow a child to continue if its parent terminates
d.	**any of the above reasons.**

8.	Semaphore can be used for solving:
a.	Wait & signal
b.	Deadlock
c.	Priority
d.	**Synchronization**

9.	Which of the following is not used for synchronization?
a.	Peterson’s Solution
b.	**Interrupts**
c.	Mutex Locks
d.	All the above

10.	The context-switch time is:
a.	dependent on the underlying hardware support
b.	overhead
c.	shorter in complex OS and PCB
d.	**all the above**

11.	Which of the following statements is true?
a.	**Shared memory is typically faster than message passing.**
b.	Message passing is typically faster than shared memory.
c.	Message passing is most useful for exchanging large amounts of data.
d.	Shared

12.	The state of a process is defined by the:
a.	final activity of the process
b.	activity just executed by the process
c.	activity to next be executed by the process
d.	**current activity of the process**

13.	Mutual exclusion can be enforced with a general semaphore whose initial value is:
a.	Greater than 1
b.	Less than 0
c.	**Equal 1.**
d.	Equal 0

14.	A race condition is when:
a.	several threads try to access the same data concurrently
b.	**when several threads try to access and modify the same data concurrently**
c.	the outcome of execution does not depend on the order in which instructions are executed
d.	None of the above

15.	____________ occurs when a higher-priority process needs a resource that is currently being accessed by a lower-priority process.
a.	**Priority inversion**
b.	Deadlock
c.	A race condition
d.	A critical section

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.

11.	12.	13.	14.	15.

16.Answer following statement using True (T) or False (F):

Statement	Answer
a. A binary semaphore takes on numerical values 0 and 1 only	**True**
b. An atomic operation is one that must be executed to completion without interruption	**True**
c. While a process is blocked on a semaphore's queue, it is in busy waiting.	**False**
d. User-level threads have kernel support.	**False**
e. `test and set()` instruction can be used for synchronization.	**True**

Question 2. [5 marks]

2-a) [1 mark] Explain the main differences between a short-term and long-term scheduler.

The primary distinction between the two schedulers lies in the frequency of execution. The short-term scheduler is designed to frequently select a new process for the CPU, at least once every 100 milliseconds. Because of the short time between executions, the short-term scheduler must be fast. The long-term scheduler executes much less frequently; minutes may separate the creation of one new process and the next. The long-term scheduler controls the degree of multiprogramming. Because of the longer interval between executions, the long-term scheduler can afford to take more time to decide which process should be selected for execution. [0.5+0.5 mark] Give any TWO reasons for why we need process cooperation.

2-b) Give three reasons for why we need process cooperation.

There are several reasons for providing an environment that allows process cooperation:

• Information sharing. Since several users may be interested in the same piece of information (for instance, a shared file), we must provide an environment to allow concurrent access to such information.

• Computation speedup. If we want a particular task to run faster, we must break it into subtasks, each of which will be executing in parallel with the others. Notice that such a speedup can be achieved only if the computer has multiple processing cores.

• Modularity. We may want to construct the system in a modular fashion, dividing the system functions into separate processes or threads, as we discussed in Chapter 2.

• Convenience. Even an individual user may work on many tasks at the same time. For instance, a user may be editing, listening to music, and compiling in parallel.

2-c) [1 mark] Describe in detail THREE different events that would cause an executing process to stop using CPU and be switched to any other state.

• The process may issue an I/O request and then be placed in an I/O queue.

• The process may create a new subprocess and wait for the subprocess's termination.

• The process may be removed forcibly from the CPU because of an interrupt

• The process may be terminate execution normally or abnormally

2-d) [0.5 + 0.5 mark] How many times does each of the programs below print “Hello”?

```
int main (){
  fork( );
  fork( );
  fork( );
  printf(“Hello ”);
}
```

```
int main (){
  fork( );
  printf(“Hello ”);
  fork( );
  printf(“Hello ”);
  fork( );
  printf(“Hello ”);
}
```

Ans: 8

Ans: 14

2-e) [1 mark] Inter-process communication (IPC) can be done either by Shared Memory or Message Passing. What is the main problem with Shared Memory method of IPC?

Protection, synchronization

Question 3. [5 marks]

3-a) [1 mark] Creating an additional thread is considered to have less cost than creating a new process. Why?

As the threads share the heap and the global variables, the system does not need to duplicate these components when creating a new thread.

3-b) [0.5+0.5 mark] Multithreading can be implemented using different techniques: Many to One, One to One, Many to Many.

i) What is the disadvantage of the Many-to-One solution?

If one process invokes a blocking kernel calls, the kernel thread will be blocked and all the other user threads will also be blocked, reducing the benefit of multithreading.

3-c) [1 mark] What is the disadvantage of the One-to-One solution?

Each time a new user thread is created, a kernel thread must also be created which takes additional time and reduces the performance of the running application.

3-d) [1 mark] Provide two programming examples in which multithreading provides better performance than a single-threaded solution.

1. A web browser with separate threads playing sound, downloading a file, collecting user input, etc.
2. A word processor with a thread to save, at regular intervals, the file that is currently being executed, and another thread a spell-checker, etc
3. An application such as matrix multiplication where each row of the matrix product is evaluated, in parallel, by a different thread.

3-e) [1 mark] The thread cancellation can be performed using one of the two methods: Asynchronous cancellation or Deferred cancellation. Explain the difference between these two methods.

In asynchronous cancellation, the thread is immediately cancelled in response to a cancellation request. There is no insurance that it did not quit in the middle of a data update or other potentially dangerous situation. In deferred cancellation, the thread polls whether or not it should terminate. This way, the thread can be made to cancel at a convenient time .

3-f) [1 mark] Multicore or multiprocessor has many processor on a single chip. What are the problems in programming for multicore systems??

Dividing activities, Balance, Data splitting, Data dependency, Testing and debugging

Question 4. [5 marks]

4-a) [1 mark] Briefly describe the characteristics of a complete solution to the critical section problem.

- **Mutual Exclusion - If process Pi is executing in its critical section, then no other processes can be executing in their critical sections.**
- **Progress - If no process is executing in its critical section and there exist some processes that wish to enter their critical section, then the selection of the processes that will enter the critical section next cannot be postponed indefinitely.**
- **Bounded Waiting - A bound must exist on the number of times that other processes are allowed to enter their critical sections after a process has made a request to enter its critical section and before that request is granted.**
- **Assume that each process executes at a nonzero speed.**
- **No assumption concerning relative speed of the N processes.**

4-b) [1 mark] Explain how deadlock state occurs between two processes.

It occurs when a process is waiting for an event that can be caused only be another process.

4-c) [0.5 + 0.5 marks] One of the conditions in critical section solutions is that the operation(s) should be executed atomically? What does it mean? What problem occurs if this condition is violated?

Operation must finish completely without interruption. Mutual exclusion will be violated.

4-d) [1 mark] The critical-section problem could be solved simply in a single-processor by preventing interrupts from occurring while a shared variable was being modified. What problem this technique causes when used in multiprocessor environment?

Disabling interrupts on a multiprocessor can be time consuming, since the message is passed to all the processors. This message passing delays entry into each critical section, and system efficiency decreases.

4-e) [1 mark] Busy waiting means that a process is waiting for a condition to be satisfied in a tight loop without relinquishing the processor. What problem it causes?

Wastes CPU cycles.