# BITS, Pilani – Dubai
## Dubai International Academic City, Dubai

**III Year (Computer Science)**
**First Semester, 2009-2010**

## Comprehensive Examination

**Course No: CS C372** | **Course Title: Operating Systems**
**Date: 28th Dec 2009** | **Weightage: 40%**
**Duration: 3 Hours** | **Max. Marks. 80**

**(Answer Parts A and B on separate answer books.)**
**(Answer the questions in the sequential order.)**
**(Answer all the parts of a question together.)**

## PART – A

1. How is single buffering and double buffering advantageous in I/O operation by a process compared to that with no buffering. **[ 4 Marks]**

2. A disk scheduler receives the following sequence of request for tracks. 40, 18, 30, 20, 11. Find out the total seek time for both FIFO and SCAN scheduling policies, assuming a seek time of 1ms for every track. (Initially request queue is empty, and head is on track no 0, which is inner most track and track 77 is the outer most track). **[ 4 Marks]**

3. What are the advantages of maintaining a thread pool? **[ 4 Marks]**

4. a) With respect to the following resource allocations for the processes P1, P2, P3 and P4, Resources R1, R2, and R3 given below, using banker's algorithm show how resource allocation can be completed without any deadlock. **[ 6 Marks]**

| | R1 | R2 | R3 |
|---|---|---|---|
| P1 | 3 | 3 | 4 |
| P2 | 6 | 1 | 3 |
| P3 | 3 | 1 | 4 |
| P4 | 4 | 3 | 2 |

**Claim Matrix**

| R1 | R2 | R3 |
|---|---|---|
| 1 | 0 | 0 |
| 6 | 1 | 2 |
| 2 | 1 | 1 |
| 0 | 0 | 2 |

**Allocation Matrix**

| R1 | R2 | R3 |
|---|---|---|
| 9 | 3 | 6 |

**Resource Vector**

| | | |
|---|---|---|
| 0 | 1 | 1 |

**Available Vector**

b) What are the conditions by which deadlock can be avoided? **[ 6 Marks]**

**[P.T.O.]**

5. a) Outline with an example fragmentation problem in Dynamic Memory Allocation. **[4 Marks]**

b) How is the problem of fragmentation overcome using Virtual Memory? **[8 Marks]**

---

## PART – B

6. For the Bounded buffer problem shown, explain what will happen if the wait(empty) and wait(mutex) are interchanged in the producer code? **[4 Marks]**

```
semaphore mutex = 1;
semaphore empty = N; // count of empty buffers
semaphore full = 0;
```

```
producer {
    while(1) {
        Produce new resource;
        wait(empty);
        wait(mutex);
        Add resource to an empty buffer;
        signal(mutex);
        signal(full);
    }
}
```

```
consumer {
    while(1) {
        wait(full);
        wait(mutex);
        Remove resource from an empty buffer;
        signal(mutex);
        signal(empty);
        Consume resource;
    }
}
```

7. What are the advantages of semaphores over locks? **[4 Marks]**

8. a) What are the synchronization problems in bounded buffer? **[4 Marks]**
b) How are they overcome using monitors? (Using pseudo code, clearly state what is done). **[8 Marks]**

9. a) Explain the difference between mode switch and process switch, in operating systems? **[ 6 Marks]**
b) Consider a uniprocessor based OS where process P1 and P2 are running. Process P1 is having kernel level threads Thr1 and Thr2, and process P2 is having kernel level threads Thr3, Thr4, Thr5. Now justify what type of switching (mode/process) takes place due to thread quantum interval interrupt and process quantum interval interrupt. **[ 6 Marks ]**

10. a) Outline with block diagram the structure of microkernel based operating system, in single machine? **[8 Marks]**
b) How it can be extended to build a distributed operating system? **[4 Marks]**

---

**9.** a) Explain the difference between mode switch and process switch, in operating systems? **[ 6 Marks]**

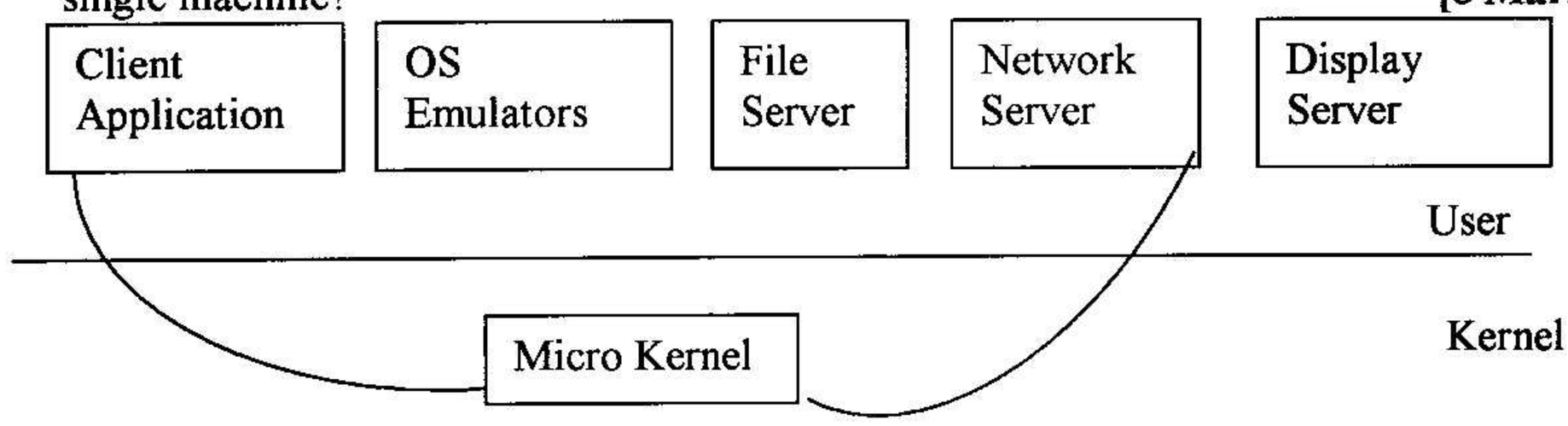
Mode switch may occur without changing the state of process currently in running state
Mode switch involves only context saving and subsequent restoral-little overhead
In the case of process switching followed by mode switch, then the state of the current process will move to Ready or blocked
Now the OS needs to do substantial changes in its environment, more time consuming.

Save context of processor including program counter and other registers
Update the process control block of the process that is currently running
Move process control block to appropriate queue - ready, blocked
Select another process for execution

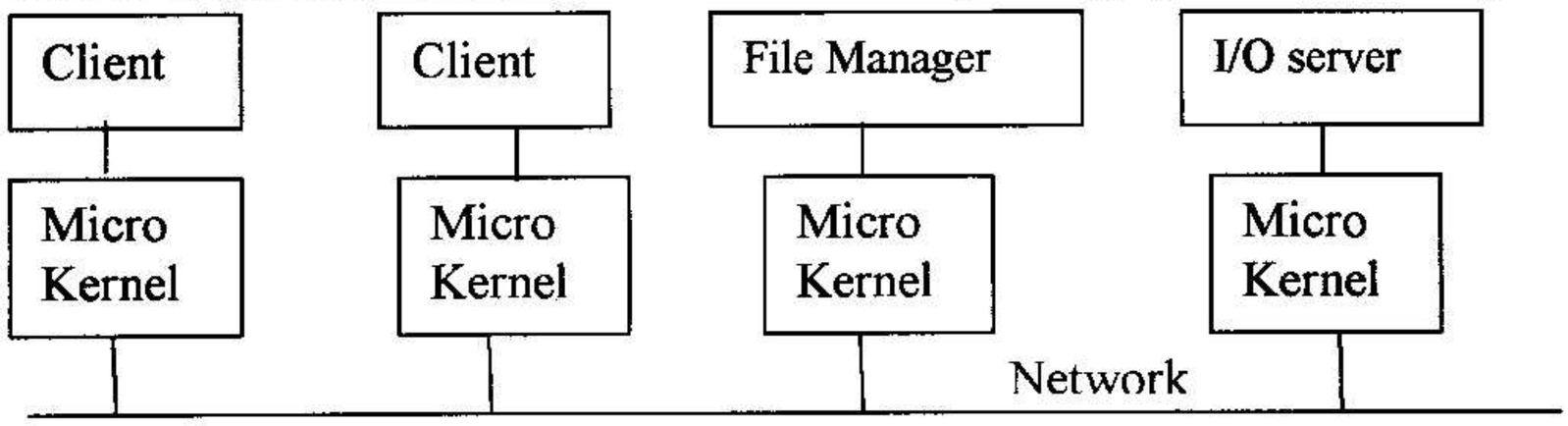
b) Consider a uniprocessor based OS where process P1 and P2 are running. Process P1 is having kernel level threads Thr1 and Thr2, and process P2 is having kernel level threads Thr3, Thr4, Thr5. Now justify what type of switching (mode/process) takes place due to thread quantum interval interrupt and process quantum interval interrupt. **[ 6 Marks ]**

Thread 1 and Thread 2 are time multiplexed during process 1 quantum interval, using thread quantum interval timer. Similarly Thread 3,4,5 are time multiplexed during process p2 quantum interval.
Hence mode switch happens during thread quantum interval, where as process quantum interval interrupt schedules a new process running. Hence process switch takes place.

**10.** a) Outline with block diagram the structure of microkernel based operating system, in single machine? **[8 Marks]**




b) How it can be extended to build a distributed operating system? **[4 Marks]**




END

# BITS, Pilani – Dubai

Dubai International Academic City, Dubai

**III Year - CS**
**First Semester 2009-2010**

COURSE NO. : CS C372

COURSE TITLE : Operating Systems

**Test 2 (Open Book)**

**Date : 13-12-2009.**

**Total marks=40, Weightage=20%**

**Class notes and Text book alone are permitted.**

## Answer all the questions

Q1. Shown below is the implementation of reader-writer problem. Three reader threads (R1, R2, R3) are created at t=0ms, t=2ms, t=4ms, and each thread takes 10ms to complete the read() method. At t=6ms a writer thread starts.

```
int read_count = 0;
semaphore mutex = 1;
semaphore w_or_r = 1;

writer {
    wait ( w_or_r );
    write();
    signal ( w_or_r );
}
```

```
reader() {
    wait (mutex);
    readcount += 1;
    If (readcount == 1)  wait (w_or_s);
    signal (mutex);
    read();
    wait(mutex);
    readcount -= 1;
    If ( readcount == 0 ) signal (w_or_s);
    signal (mutex);
}
```

Now specify, the count values of the various semaphores, and the value of readcount, with respect to time, as each of the the reader threads enter one after another, perform read() and leave one after another. Ignore the other delays. ----- 10 M

**Ans:**

| | Initial | R1 Entry | R2 Entry | R3 Entry | W Entry | R1 Exit | R2 Exit | R3 Exit |
|---|---|---|---|---|---|---|---|---|
| Mutex | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| w_or_r | 1 | 0 | 0 | 0 | -1 | -1 | -1 | 0 |
| readcount | 0 | 1 | 2 | 3 | 3 | 2 | 1 | 0 |



(PTO)

Q2. Shown below is a function called by two threads t1 and t2 of a process to print their respective thread ids, in a uni-cpu environment. mytid() gives a unique thread id for t1(=100) and t2(=200).

```
static int a;
void printtid() {
    a= mytid();
                    ---→ Context switch for thread 1
    cout << a;
                    --→ Context switch for thread 2.
}
```

a) What will be output for above problem? ------ 5 M

**Ans:**

**Output 200 200**

b) Give a solution so that expected output will be printed? ------ 5 M

**Ans:**

```
semaphore mutex = 1;
 static int a;

 void printtid() {
 {
        wait(mutex);
        a = mytid();
        cout << a;
        signal (mutex);
 }
```

Q3. a) Explain how monitor construct guarantees mutual exclusion? ----- 5 M

**Ans:**

Associated with every monitor is a binary semaphore, initialized to 1. So any process invoking the monitor, does a wait()/signal() on the associated semaphore, whenever the procedures in the monitor are entered and exited.

So the first process calling any of the monitor methods is the only one allowed entry into the monitor. Other processes calling any of the methods are put on wait queue associated with the semaphore, until the first process exits the method. Thus mutual exclusion is guaranteed.

b) Explain the difference between the wait/signal operations associated with semaphores and that associated with conditional variables used in Monitors.? ------ 5 M

**Ans:**

Wait (semaphore) : decrement and block if necessary;

Wait (condition variable) : unconditional block (no decrement);

Signal(semaphore) : increment and unblock if necessary

Signal(conditional variable) : resumes only 1 suspended process, if no process is suspended the signal has no effect (i.e. lost).

Q4. Consider 2 functions func1( ) run by thread1 and func2( ) run by thread2, in a non-preemptive thread scheduler of a process P. Thread 1 is started before thread 2.

```
Func1( ) {
   int a =10;
   int b=20;
   Threadsleep( 5000);
  int c = a +b;
   cout << c;
   thread_yield();
  Threadsleep(3000);
   d = c +10;
   cout << d;
}
```

```
func2( ) {
   int x =10;
   int y=20;
   Threadsleep( 2000);
   Int z = x + y;
   cout << z;
   thread_yield();
}
```

a) Specify the times during which the values of c, d and z will be printed. ----- 6 M

**Ans:**

| Variable | Time |
|---|---|
| c | 5000 |
| z | 7000 (5000+ 2000) |
| d | 10000 (7000+3000) |

b) Suppose before the thread_yield() in func1, we add a statement thread_exit(), the what will happen to the outputs in part (a) ---- 4 M

**Ans:**

| Variable | Time |
|---|---|
| c | 5000 |
| z | 7000 (5000+ 2000) |
| d | Will not be printed |

---

# BITS, Pilani – Dubai

Dubai International Academic City, Dubai

**III Year - CS**

**First Semester 2009-2010**

COURSE NO. : CS C372

COURSE TITLE : Operating Systems

**Date : 25-10-2009.**

**Test 1 (Closed Book)**

**Total marks=50, Weightage=25%**

## Answer all the questions

*Answering and evaluation scheme*

Q1.What will be the effect of the following in the execution of OS

a) Hardware does not support kernel/user mode (6 Marks)

If the HW does not support kernel mode of operation then the OS code has to be run in user mode as similar to any other user application. Hence user applications can try to overwrite in the memory space occupied by the OS and corrupt it. Further malfunctioning user programs can also try to access the I/O hardware directly resulting in system crash

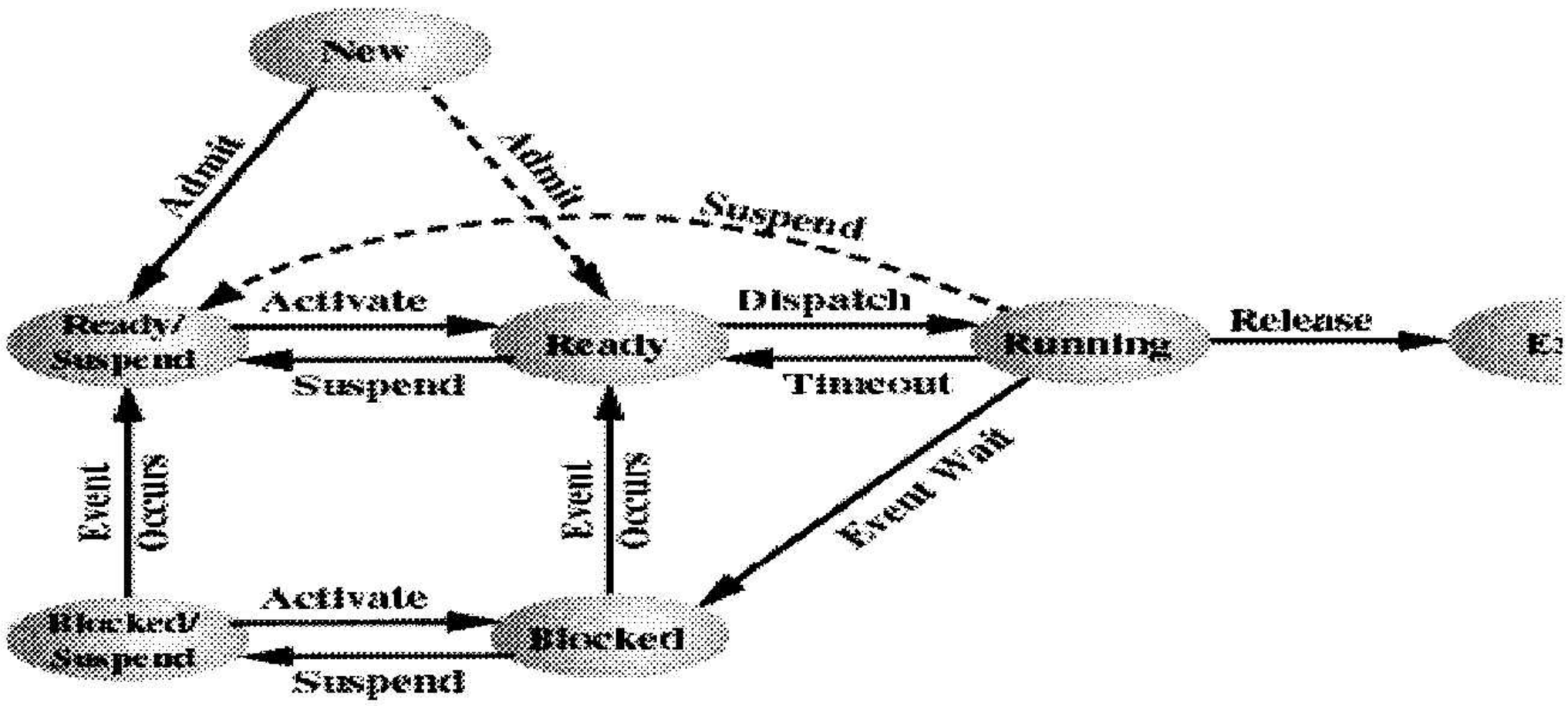
b) Hardware does not support timers (7 Marks)

Without timers we can not generate periodical interrupt signals to OS. Without periodic interrupt signals OS can not perform time-interleaved multitasking effectively.

Also we can not keep track of whether a process has exceeded the time limit.

Q2. a) Draw the state diagram of an OS that has 7 states, indicating clearly the states and the events which cause the transition between the states. (6 Marks)

This is as per the state diagram which incorporated the suspended states (blocked-suspended and Ready-suspended) in addition to the blocked state.




(b) With Two Suspend States

b) Describe the events that cause the transitions from Blocked/Suspended state to Ready/Suspended state and then to Ready state.

A process in the blocked-suspended state is waiting for some event to occur after it has been swapped out in Secondary storage and then becomes ready/suspended state after the event. (6 Marks)

Q3. Justify with reasons as to which type of the OS structure the following characteristics correspond to?

a) Resilience to errant kernel modules.

In microkernel based operating system structure is resilence to errant kernel modules. Since modules like file server, process server all operate in user mode without direct access to I/O devices. Kernel alone operates in protected mode

b) Additional latencies in communication among OS modules

In case of microkernel approach since every client request for files or process has to pass through the kernel via messages .Further the request of the client has to be validated by the kernel.. Hence additional latencies are introduced compared to monolithic approach. In monolithic approach using system call the client can ask for files and the file handler module will accesss the hardware and get the file back. Hence time to serve the request is minimized in monolithic approach

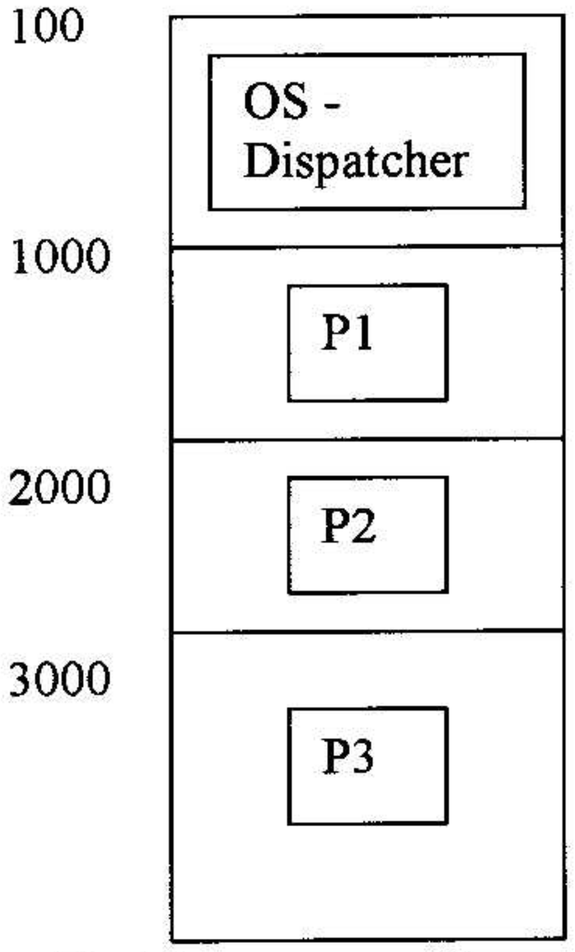
c) Each part of the kernel has access to every other part.

In case of monolithic the above problem is there. In monolithic approach any module of the OS can have access to any other module without any security restriction. On the other hand in case of microkernel any module of the OS has to seek the permission of microkernel to access other module of the OS. ( 3 x 4 Marks)

Q4. a) Justify with an example the need for saving the context of a process (7 Marks)

A multiprogramming/multitasking OS, interleaves the execution of process. That is a executing process is interrupted and the cpu is given to another process to execute. At a later point in time, the interrupted processes must resume the execution from the point at which it was interrupted. This is possible by saving the execution context of the process when it was interrupted, and restoring the context when it scheduled to execute later. The execution context is the set of registers, PC, stack pointer et

b) Given the snapshot of execution of process in an OS outline the activities that are happening in the system. (6 Marks)




| | |
|---|---|
| 1 1000 | 12 100 |
| 2 1001 | 13 101 |
| 3 1002 | 14 102 |
| ----------(Time Out) | 15 103 |
| 4 100 | 16 3000 |
| 5 101 | 17 3001 |
| 6 102 | 18 3002 |
| 7 103 | 19 3003 |
| 8 2000 | 20 3004 |
| 9 2001 | ----------(Time out) |
| 10 2002 | 21 100 |
| 11 2003 | 22 101 |
| ----------(Time Out) | 23 102 |
| | 24 103 |
| | 25 1003 |
| | 26 1004 |

First the process P1 executes from location 1000 its first 3 instructions.(1,2,3). Then the quantum interval get over and the dispatcher gets control (4,5,6,7). This selects process p2 and its executing from location 2000 till ,2003 (8-11). Then another timeout happens and dispatcher gets control (12-15). Then process p3 starts executing from location 3000. (16-20.). Then quantum interval gets over and dispatcher executes (21-24). Process P1 resumes execution from 1003.(25-26). This is possible because the context of P1 was saved when it

**********

A

BITS, PILANI – DUBAI
FIRST SEMESTER 2009 – 2010
FIRST YEAR

Course Code: CS C372
Course Title: Operating Systems
Duration : 20 minutes
Weightage: 7%

Date: 25.11.09
Max Marks: 14
Quiz2(Closed book)

Name: . . . . . ID No: . . . . . Sec / Prog: . . . . . .

Instructions: (if any) closed book- Answer all the Questions-Answering and marking scheme

1. Specify one advantage and disadvantage of Disable Interrupts type of lock compared to spin lock (2 Marks

2.

```
struct lock {
  int held = 0;
}
```

```
void acquire (lock) {
        while (lock->held) ;
         lock->held = 1;
}
```

```
void release (lock) {
        lock->held = 0;
}
```

When does the above type of lock fail mutual exclusion? (2 marks)

3. Why Disable Interrupts type of lock cannot be used for multiprocessor environment? (2 marks)

PTO

A

4. What is the o/p of the Java code shown below ? (2 marks

```
public class HW_Thd extends Thread {

 public void run() {
       Thread.sleep(5000);
       System.out.println("I am ok");
     }

 public static void main(String[] args) {
   HW_Thd thd1 = new HW_Thd();
   thd1.start();
   HW_Thd thd2 = new HW_Thd();
   thd2.start();
    System.out.println("Hi How are you ?");
 }
}
```

5. Explain the operations that take place in the wait() and signal() semaphore primitives (2 mark

6.

| | |
|---|---|
| struct Semaphore {<br>int value;<br>Queue q;<br>} S;<br><br>The semaphore also has wait( ) and signal( ) methods. | withdraw(account, amount) {<br>wait(S);<br>balance = get_balance(account);<br>balance = balance – amount;<br>put_balance(account, balance);<br>signal(S);<br>return balance;<br>} |

In the above code, what will happen if the statements signal(S) and return balance are interchanged? (2 marks)

7. Specify any two important requirements of Critical Section (2 marks)

# BITS, PILANI – DUBAI

## FIRST SEMESTER 2009 – 2010

**FIRST YEAR**

Course Code: CS C372
Course Title: Operating Systems
Duration : 20 minutes
Weightage: 8%

Date: 07.10.09
Max Marks: 16
Quiz1(Closed book)

**Name:** . . . . . **ID No:** . . . . . **Sec / Prog:** . . . . . .

Instructions: (if any) closed book- Answer all the Questions

1) Distinguish between Multiprogramming OS and Multitasking OS (2 marks)

2) In which mode the operating system should run to do the following activities (2 marks)
a) Loading of a Timer

b) Reading of System Time.

3) What is meant by Quantum Interval in multitasking? (2 marks)

4) How does the OS prevent a user program from running too long due to some indefinite loop? (2 marks)

5) What is the difference between a Desktop OS and Real-time system OS ? (2 marks

PTO

6) Distinguish between asymmetric and symmetric multiprocessing (2 marks)

7) How does the OS gives the illusion of infinite memory for any user program? (2 marks)

8) Distinguish between Program and Process. (2 marks)

Please use both the sides and mention **PTO** at the end of 1st page