Birla Institute of Technology & Science - Pilani, Hyderabad Campus

CS F372 : Operating Systems

Comprehensive Test

Type: Closed Book **Time:** 180 mins **Date:** May 3rd, 2019 **Max Marks:** 70 (35% weightage)

Please Notes:

- You must answer all the parts of a question together as only the first occurrence of the answer to a question will be graded.
- Be cognizant of time. Be concise. Show work when needed. Write your answers legibly.
- Assume suitable data if necessary. Don't cripple the question with unnecessary assumptions.

---

1. (a) Draw the state diagram of a process from its creation to termination, including all transitions. Enumerate and elaborate the conditions on every transition in the diagram. **[10 Mark(s)]**

   (b) Consider n processes sharing a CPU in a RR fashion. Assuming that each process switch takes s seconds, what must be the quantum size q such that the overhead resulting from process switching is minimized but, at the same time, each process is guaranteed to get its turn at the CPU at least every t seconds? (Show your work) **[4 Mark(s)]**

2. (a) Explain progress and bounded wait conditions and enumerate their differences. **[8 Mark(s)]**

   (b) Suppose n processes are contending for a critical section, and we are using a counting semaphore to provide controlled access to the critical section. What will happen if the value of the counting semaphore is initialized to 3? **[2 Mark(s)]**

   (c) The function named "some_function", defined below is used in a multi-threaded program. The variables shared_data1 and shared_data2 are shared by all the threads in the process. Assume a variable of class Semaphore has been declared, named lock, and is accessible to all the threads. **[4 Mark(s)]**

      i. Edit function "some_function" by adding operations on the variable named lock to ensure mutual exclusion in the critical section.

      ii. What value should lock be initialized to?

```
(void *)some_function(void *arg){
 if(arg){
  shared_data1 += 1;
  shared_data2 *= shared_data1;
 }
 else
  printf("error");
}
```

3. (a) Consider a system having m instances of a resource, which are shared by n processes P_1, P_2, ... and P_n. A process P_i requires R_i resources (i.e. count/instances of the above resource type). Now determine the value of m for which the system will not face a deadlock? Show your work. **[4 Mark(s)]**

   (b) Two processes, A and B, each need three records 1, 2, and 3, in a database. If A asks for them in the order 1,2,3, and B asks for them in the order 1,2,3 deadlock is not possible. However, if B asks for them in the order 3,2,1, then deadlock is possible. With three resources, there are 3! = 6 possible combinations each process can request resources. What fraction of all combinations are guaranteed to be deadlock free? **[5 Mark(s)]**

(c) In a system using a single processor, a new process arrives at the rate of six processes per minute and each such process requires seven seconds of service time. What is the CPU utilization? Show your work. **[2 Mark(s)]**

(d) Given a total of 5 units of resource type 1 and 4 units of resource type 2, tell whether the following system (Table 1) is in a safe or unsafe state. Show your work. **[3 Mark(s)]**

Table 1: System State (Used and Maximum demand)

Process	**Type 1 Used**	**Type 1 Max demand**	**Type 2 Used**	**Type 2 Max demand**
P1	1	2	1	3
P2	1	3	1	2
P3	2	4	1	4

4. (a) Consider a paging system with the page table stored in memory. A memory reference takes 200 nanoseconds. In order to decrease the effective memory access time, we have added associative registers, and 75 percent of all page-table references are found in the associative registers. What is the speed up in the effective memory reference time after the addition of associative registers? (Assume that finding a page-table entry in the associative registers takes zero time if the entry is there.) **[4 Mark(s)]**

(b) Consider the two-dimensional array A:

```
int A[][] = new int[50][50];
```

where A[0][0] is at location 200 in a paged memory system with pages of size 200 (A single page can hold 200 elements of the array). A small process that manipulates the matrix resides in page 0 (locations 0 to 199). Thus, every instruction fetch will be from page 0. For three page frames, how many page faults are generated by the following array-initialization loops, using LRU replacement and assuming that page frame 1 contains the process and the other two are initially empty? Explain your answer with necessary diagram assuming the array is stored in row-major order and i is used for accessing row and j is used for accessing column.

```
for (int j = 0; j < 50; j++)
 for (int i = 0; i < 50; i++)
  A[i][j] = 0;
```

[4 Mark(s)]

(c) Consider the page table shown in Figure 2 for a system with 12-bit virtual and physical addresses and with 256-byte pages. The list of free page frames is D, E, F (D is at the head of the list, E is second, and F is last). Convert the following virtual addresses to their equivalent physical addresses in hexadecimal. (A dash for a page frame indicates that the page is not in memory.)

i. 0x111 ii. 0x4EF iii. 0x700

Assume that there was a page fault due to a logical address which was issued by the CPU just before the logical addresses 0x111. **[2+2+2 Mark(s)]**

5. (a) Consider 98, 183, 37, 122, 14, 124, 65, 65, 67 are the disk request sequence for a disk with 200 tracks. The initial position of the R/W head is 53 and is moving in the left (track

Table 2: Page Table

Page	Page Frame
0	-
1	2
2	C
3	A
4	-
5	4
6	3
7	-

0) direction. Find the number of head movements in cylinders using SCAN scheduling. Write in which order the disk access requests are fulfilled. **[2 Mark(s)]**

(b) A disk rotates at a speed of 3600 rpm. It has 4000 cylinders, 16 surface and 206 sectors per track (64000 tracks). Track storage capacity is 2048 bytes. What are the average latency time (in msec) and data transfer rate (in kbps) respectively? What is the capacity of the disk? **[2+1 Mark(s)]**

(c) Consider a system having buddy system with physical address space 128 KB. Calculate the size of a partition for 18 KB process. Explain your answer with a neat figure if required. **[2 Mark(s)]**

(d) A CPU scheduling algorithm determines an order for the execution of its scheduled processes. Given n processes to be scheduled on one processor, how many possible different schedules are there? Give a formula in terms of n. **[1 Mark(s)]**

(e) Suppose that the following processes (Table 3) arrive for execution at the times indicated. Each process will run the listed amount of time. In answering the questions, use non-preemptive scheduling and base all decisions on the information you have at the time the decision must be made. **[6 Mark(s)]**

Table 3: Process Arrival and Burst Time

Process	arrival time	Burst time
P1	0	8
P2	0.4	4
P3	1.0	1

i. What is the average turnaround time for these processes with the FCFS scheduling algorithm?

ii. What is the average turnaround time for these process with the SJF scheduling algorithm?

iii. The SJF algorithm is supposed to improve performance, but notice that we chose to run process Pi at time 0 because we did not know that two shorter processes would arrive soon. Compute what the average turnaround time will be if the CPU is left idle for the first 1 unit and then SJF scheduling is used. Remember that processes p1 and P2 are waiting during this idle time, so their waiting time may increase. This algorithm could be known as future-knowledge scheduling.

–End–