Ex/PG/MTCT/T/121A/2022

Master of Technology in Computer Technology

First Year Second Semester Examination, 2022

Subject: Advanced Operating Systems

Time: 3 Hours | Full Marks: 100

Answer Any Five Questions

1. (a) "Implementing preemptive scheduling needs hardware support". State whether this statement is true or false. Provide necessary justifications.

   (b) What is the difference between a long term and a short term scheduler? What is the role of a mid term scheduler?

   (c) Consider the following set of processes that need to be scheduled on a single CPU. All the times are given in milliseconds.

Process	Arrival Time	Execution Time
A	0	6
B	3	2
C	5	4
D	7	6
E	10	3

   Using the shortest remaining time first scheduling algorithm, determine the average turnaround time and waiting time (in msec).

   (d) For the following code segment determine the output and state the number of new processes that will be created. Provide necessary justifications.

```
for (i=0; i<=4; i++){
    if(i==2)
        fork();
    else
        printf("%d\n", i);
}
```

3+3+6+8=20

2. (a) Consider a set of three periodic tasks with the execution profile as given in the following table.

Task	**Period**	**Execution Time**
A	6	2
B	8	2
C	12	3

   Draw the scheduling diagrams considering the following scheduling policies.

   i. Earliest Deadline First

ii. Rate Monotonic

(b) Consider a set of five aperiodic tasks with the execution profile as given in the following table.

Task	Period	Execution Time	Starting Deadline
A	10	20	110
B	20	20	20
C	40	20	60
D	50	20	70
E	60	20	80

Draw the scheduling diagrams considering the following scheduling policies.

i. Earliest Deadline
ii. Earliest Deadline with Unforced Idle Times
iii. First Come First Served

(c) Describe how the priority ceiling approach avoids the priority inversion problem.

(d) What is a `TestAndSet` instruction?

6+9+3+2=20

3. (a) In an operating system, standard atomic operations wait and signal on a semaphore S are implemented as follows.

```
wait(S){
    while(S<=0);
    S--;
}
signal(S){
    S++;
}
```

Is the solution free of busy waiting? If not suggest a suitable solution to overcome this problem.

(b) P_1, P_2 and P_3 are three processes executing their respective tasks. They should synchronize among themselves using semaphores such that the string "ABCBCACAB" gets printed infinite times. Determine, minimum number of semaphores required and their initial values. Also identify places where operations on those semaphore should be inserted in the code of P_1, P_2 and P_3. Provide necessary justifications.

P_1
```
while(true){
    print("A");
}
```

P_2
```
while(true){
    print("B");
}
```

P_3
```
while(true){
    print("C");
}
```

(c) Three processes P_1, P_2 and P_3 execute the following code using two semaphores `a` and `b` initialized to 1 and 0, respectively. Assume that `count` is a shared variable initialized to 0 and not used in `CODE SECTION P`.

```
<CODE SECTION P>
wait(a);
```

```
count=count+1;
if(count==3)
    signal(b);
signal(a);
wait(b);
signal(b);
<CODE SECTION Q>
```

Is there a possibility that while one process is executing in `CODE SECTION Q` other processes are executing in `CODE SECTION P`? Provide necessary justifications.

(d) How resource allocation graphs are used for deadlock detection?

4+8+6+2=20

4. (a) State how deadlock can be prevented by ensuring that circular wait condition does not hold.

(b) Consider the following snapshot of a system running n concurrent processes. Process i is holding X_i instances of a resource R, $1 \leq i \leq n$. Assume that all instances of R are currently in use. Further, for all i, process i can place a request for at most Y_i additional instances of R while holding the X_i instances it already has. Of the n processes, there are exactly two processes p and q such that $Y_p = Y_q = 0$. If the condition $X_p + X_q < Min\{Y_k \mid 1 \leq k \leq n, k \neq p, k \neq q\}$ holds true, then determine whether the system is approaching a deadlock state. Provide necessary justification.

(c) Consider the following snapshot of a system:

	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	4	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

Answer the following questions using the Banker's algorithm:

i. Is the system in a safe state?

ii. If a request from process P1 arrives for (0, 4, 2, 0) can the request be granted immediately?

(d) A system has 3 user processes P_1, P_2 and P_3, where P_1 requires 2 units of resource R, P_2 requires 3 units of resource R, P_3 requires 4 units of resource R. Determine the minimum number of units of R that ensures no deadlock.

4+4+8+4=20

5. (a) Differentiate between external and internal fragmentation.

(b) What type of fragmentation does paging solve and how?

(c) In a virtual memory system, size of virtual address is 44-bit, size of physical address is 34-bit, page size is 16 Kbyte and size of each page table entry is 32 bit. The main memory is byte addressable. Determine the maximum number of bits that can be used for storing protection and other information in each page table entry? Also, calculate the size of the page table.

(d) Consider a three-level page table to translate a 40-bit virtual address to a physical address as follows:

<– – – – –40-bit virtual address– – – – –>

Level 1 offset	Level 2 offset	Level 3 offset	Page offset
9 bits	9 bits	9 bits	13 bits

The page size is 8 KB (1KB=2^{10} bytes) and page table entry size at every level is 16 bytes. Determine the amount of memory (in KB) required for storing the page tables of of all levels.

3+4+6+7=20

6. (a) How does use of dirty bit reduces the overhead of page replacement?

   (b) Describe one factor that determines the minimum number of page frames that must be allocated to a running process in a virtual memory environment.

   (c) Consider a main memory with four page frames and the following sequence of page references: 3, 8, 2, 3, 9, 1, 6, 3, 8, 9, 3, 6. Determine the number of page faults using page replacement policies (i) First-In-First Out (FIFO) (ii) Least Recently Used (LRU) and (iii) Most Recently Used (MRU).

   (d) What are the advantages and disadvantages of having large chunk size in Google File System?

   (e) How does Google File System withstand failure of (i) the master node, and (ii) any chunk server?

   2+2+9+3+4=20

7. (a) What is seek time and latency time?

   (b) A hard disk has 64 sectors per track, 16 platters each with 2 recording surfaces and 1024 cylinders. Assuming that sector size is 512 bytes, determine the size of the disk.

   (c) A FAT (file allocation table) based file system is being used and the size of each entry in the FAT is 4 bytes. Given a 100×10^6 bytes disk on which the file system is stored and data block size is 10^3 bytes, determine the maximum size of a file that can be stored on this disk in units of 10^6 bytes.

   (d) The index node (inode) of a Unix-like file system has 12 direct, one single-indirect and one double-indirect pointers. The disk block size is 4 kB, and the disk block address is 32-bits long. Determine the maximum possible file size (in GB) supported by this file system.

   (e) Describe the different steps that client module performs when it tries to read a file stored in Google File System.

   2+3+6+6+3=20