SRM Institute of Science and Technology
College of Engineering and Technology
School of Computing

Mode of Exam
OFFLINE
SET A

DEPARTMENT OF COMPUTING TECHNOLOGIES

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu

Academic Year: 2021-2022 (Even)

Test: CLAT-2 **Date: 26-05-2022**
Course Code & Title: 18CSC205J: Operating Systems **Duration:** 2 Period
Year & Sem: 2022 & IV Semester **Max. Marks:** 50 Marks

Course Articulation Matrix: ***(to be placed)***

Part - A
(10 x 1 = 10 Marks)

Instructions: Answer all

Q. No	Question	Marks	BL	CO	PO	PI Code
1	d. 9	1	5	2	3	3.6.1
2	c. A and D	1	4	2	3	3.6.1
3	d. The size of the time quantum	1	1	2	3	3.6.1
4	a. A and B	1	3	2	3	3.6.1
5	a. Increases	1	2	2	3	3.6.1
6	a. First fit is the fastest but results in internal fragmentation.	1	1	3	1	1.6.1
7	b. CPU	1	1	3	1	1.6.1
8	c. Both 1 and 2 are true	1	1	3	1	1.6.1
9	b. Statements 1 and 3 are true	1	1	3	1	1.6.1
10	b. External fragmentation	1	1	3	1	1.6.1

Part - B
(4 x 5 = 20 Marks)

Instructions: Answer Any 5

11	**Compare Multilevel queue scheduling and Multilevel feedback scheduling algorithm with neat diagram and justify which one is best.**	5	2	2	3	3.6.2

Multilevel Queues (MLQ)	Multilevel Feedback Queues (MFQ)
It multilevel queues, algorithm required to schedule process is less complex and inflexible.	The algorithm required to schedule processes in multi-level feedback queues is more complex and flexible.
In multi level queues the various processes cannot move between different subsequent.	In multilevel feedback queues the processes in different sub-queues can move between different queues.
The various processes assigned to different sub-queues on the basis of number of factors such as memory size, priority or process types.	The various processes separated in different queues on the basis of their CPU burst characteristics.
It offers the advantage of low scheduling overhead as processes do not change their queues.	Moving processes around the queues produces more CPU overhead.
In such queues, same processes may starve for CPU if some higher priority queues are never becoming empty.	There is no problem of starvation as the process that waits too long in lower priority queue may be moved to at higher priority queue.

12	**Suppose the following two processes, foo and bar are executed concurrently and share the semaphore variables S and R (each initialized to 1) and the integer variable x (initialized to 0).**	**5**	**4**	**2**	**3**	**3.6.2**

```
void foo( ) {
do {
semWait(S);
semWait(R);
x++;
semSignal(S);
SemSignal(R);
} while (1);
}
```

```
void bar( ) {
do {
semWait(R);
semWait(S);
x--;
semSignal(S;
SemSignal(R);
} while (1);
}
```

a. **Can the concurrent execution of these two processes result in one or both being blocked forever? If yes, give an execution sequence in which one or both are blocked forever.**

b. Can the concurrent execution of these two processes result in the indefinite postponement of one of them? If yes, give an execution sequence in which one is indefinitely postponed.

ANSWER:

a. Yes. If foo( ) executes semWait(S) and then bar( ) executes semWait(R) both processes will then block

	when each executes its next instruction. Since each will then be waiting for a semSignal( ) call from the other, neither will ever resume execution. b. No. If either process blocks on a semWait( ) call then either the other process will also block as described in (a) or the other process is executing in its critical section. In the latter case, when the running process leaves its critical section, it will execute a semSignal( ) call, which will awaken the blocked process. Gradation guideline: Pretty straightforward question again. 2.5 marks can be given for each of the options					
13	**What is the effect of allowing two entries in a page table to point to the same page frame in memory? Explain how this effect could be used to decrease the amount of time needed to copy a large amount of memory from one place to another. What effect would updating some byte on the one page have on the other page?** By allowing two entries in a page table to point to the same page frame in memory, users can share code and data. If the code is reentrant, much memory space can be saved through the shared use of large programs such as text editors, compilers, and database systems. "Copying" large amounts of memory could be affected by having different page tables point to the same memory location. However, sharing of non reentrant code or data means that any user having access to the code can modify it and these modifications would be reflected in the other user's "copy."	**5**	**3**	**3**	**1**	**1.7.1**
14	**Consider a system in which a program can be separated into two parts: code and data. The CPU knows whether it wants an instruction (instruction fetch) or data (data fetch or store). Therefore, two base– limit register pairs are provided: one for instructions and one for data. The instruction base–limit register pair is automatically read-only, so programs can be shared among different users. Discuss the advantages and disadvantages of this scheme.** The major advantage of this scheme is that it is an	**5**	**3**	**3**	**2**	**2.6.2**

	effective mechanism for code and data sharing. For example, only one copy of an editor or a compiler needs to be kept in memory, and this code can be shared by all processes needing access to the editor or compiler code. Another advantage is protection of code against erroneous modification. The only disadvantage is that the code and data must be separated, which is usually adhered to in a compiler-generated code.					
15	**A certain computer system has the segmented paging architecture for virtual memory. The memory is byte addressable. Both virtual and physical address spaces contain 2^{16} bytes each. The virtual address space is divided into 8 non-overlapping equal size segments. The memory management unit (MMU) has a hardware segment table, each entry of which contains the physical address of the page table for the segment. Page tables are stored in the main memory and consist of 2 byte page table entries. What is the minimum page size in bytes so that the page table for a segment requires at most one page to store it?** Given- • Virtual Address Space = Process size = 2^{16} bytes • Physical Address Space = Main Memory size = 2^{16} bytes • Process is divided into 8 equal size segments • Page table entry size = 2 bytes Let page size = n bytes. Now, since page table has to be stored into a single page, so we must have- Size of page table <= Page size **<u>Size of Each Segment-</u>** Size of each segment = Process size / Number of segments	**5**	**4**	**3**	**3**	**3.7.1**

	$= 2^{16}$ bytes / 8 $= 2^{16}$ bytes / 2^3 $= 2^{13}$ bytes = 8 KB <u>Number of Pages of Each Segment-</u> Number of pages each segment is divided = Size of segment / Page size = 8 KB / n bytes = (8K / n) pages <u>Size of Each Page Table-</u> Size of each page table = Number of entries in page table x Page table entry size = Number of pages the segment is divided x 2 bytes = (8K / n) x 2 bytes = (16K / n) bytes <u>Page Size-</u> Substituting values in the above condition, we get- (16K / n) bytes <= n bytes (16K / n) <= n n^2 >= 16K $n^2 >= 2^{14}$ $n >= 2^7$ Thus, minimum page size possible = 2^7 bytes = 128 bytes.					
Part – C **(2 x 10 = 20 Marks)**						

Instructions: Answer All

16.a | 10 | 3 | 2 | 3 | 3.6.4

Consider the set of 5 processes whose arrival time and burst time are given below-

Process Id	Arrival time	Bu
P1	0	
P2	1	
P3	2	
P4	3	

If the CPU scheduling policy is Round Robin with time quantum = 2 unit, calculate the average waiting time and average turnaround time.

If time Quantum is increase in round robin what happens? Justify.

Answer:

Average Turn Around time = (13 + 11 + 3 + 6 + 10) / 5 = 43 / 5 = 8.6 unit

Average waiting time = (8 + 8 + 2 + 4 + 7) / 5 = 29 / 5 = 5.8 unit

It will act as FCFS.

16.b | 10 | 3 | 2 | 3 | 3.7.1

Two processes, P1 and P2, need to access a critical section of code. Consider the following synchronization construct used by the processes:

```
/* P1 */
while (true) {
wants1 = true;
while (wants2 == true);
/* critical section */
wants1 = false;

}
/* Remainder section */
```

```
/* P2 */
while (true) {
wants2 = true;
while (wants1 == true);
/* critical section */
wants2 + false;
}
/* Remainder section */
```

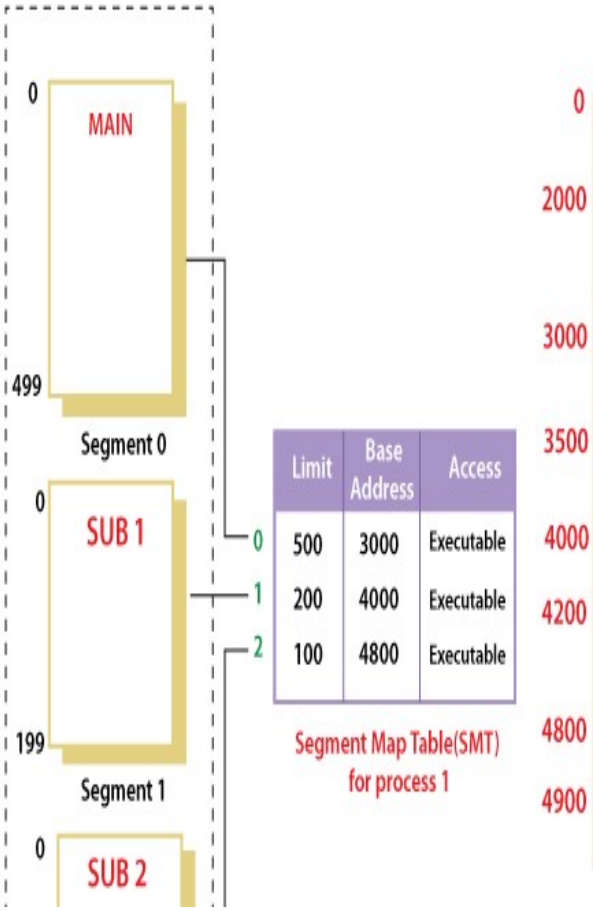
Here, wants1 and wants2 are shared variables, which are initialized to false. Which one of the following statements is TRUE about the above construct by considering all the following statements by giving proper justification for each statement?

(A) It does not ensure mutual exclusion.

(B) It does not ensure bounded waiting.

(C) It requires that processes enter the critical

	section in strict alternation. **(D) It does not prevent deadlocks, but ensures mutual exclusion.** ANSWER (2.5 marks for each justification) P1 has control of the critical section provided wants1 is true and wants2 is false. P2 has control of the critical section provided wants2 is true and wants1 is false. So if P1 has control it excludes P2 till it completes and vice versa, so mutual exclusion is ensured. This eliminates choice (A). (B) is false as the time spent by P1 and P2 in their critical sections is not controlled or bounded. (C) is not correct for one can easily see that P1 can use the resource, release it, use it again, release it and so on without P2 ever demanding it. (D) A deadlock can arise as the assignment to wants1 and wants2 is not done as an indivisible operation. So when wants1 is set to true at the same time wants2 can be set to true. This results in endless waiting. So (D) is the answer.					
17.a	**Suppose a 16 bit address is used with 4 bits for the segment number and 12 bits for the segment offset so the maximum segment size is 4096 and the maximum number of segments that can be refereed is 16.Elloborate how the Translation of Logical address into physical address been mapped by segment table method.** When a program is loaded into memory, the segmentation system tries to locate space that is large enough to hold the first segment of the process, space information is obtained from the free list maintained by memory manager. Then it tries to locate space for other segments. Once adequate space is located for all the segments, it loads them into their respective areas. The operating system also generates a segment map table for each program.	**10**	**4**	**3**	**3**	**3.6.2**

	 With the help of segment map tables and hardware assistance, the operating system can easily translate a logical address into physical address on execution of a program. The **Segment number** is mapped to the segment table. The limit of the respective segment is compared with the offset. If the offset is less than the limit then the address is valid otherwise it throws an error as the address is invalid. In the case of valid addresses, the base address of the segment is added to the offset to get the physical address of the actual word in the main memory. The above figure shows how address translation is done in case of segmentation.					
17.b	**In an Operating system, the OS doesn't care about the User's view of the process. It may divide the same function into different pages and those pages may or may not be loaded at the same time into the memory. It decreases the efficiency of the system. So identify which technique is suitable to overcome the drawback and also helps out in better efficiency and performances.**	**10**	**4**	**3**	**3**	**3.6.2**

Segmentation

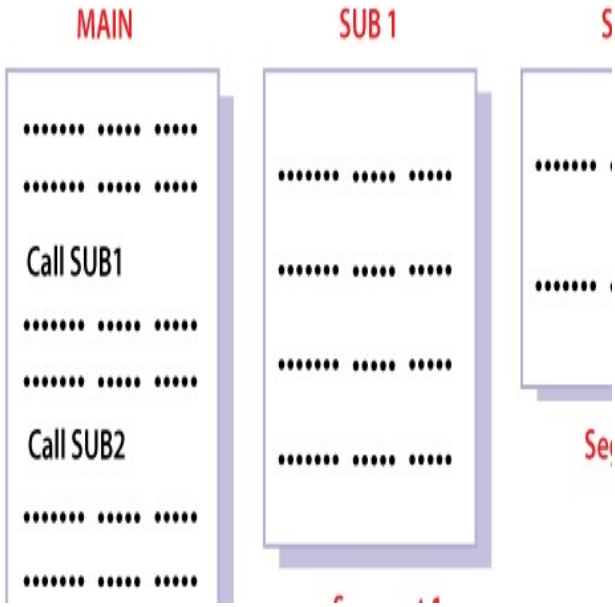
In Operating Systems, Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.

The details about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments.

Segment table contains mainly two information about segment:

1. Base: It is the base address of the segment
2. Limit: It is the length of the segment.

It is better to have segmentation which divides the process into the segments. Each segment contains the same type of functions such as the main function can be included in one segment and the library functions can be included in the other segment.



Question Paper Setter

Approved by Audit Professor/ Course Coordinator