SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 of the UGC Act, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

School of Computing
Second CIA Exam –Feb 2024
Course Code: CSE208
Course Name: Operating Systems
Duration: 90 minutes Max Marks: 50

PART A (2 * 10 = 20) Answer all the questions

1. Consider the following program execution involving 2 threads accessing the shared variable 'i' = 0.

   Thread1 Thread2
   i++; i++;
   while(i<2); while(i<2);

   Which of the following statement(s) is/are correct? justify .
   A. Both threads will never finish execution
   B. Both threads will always finish execution
   C. At least one of the threads will always finish execution
   D. Either both threads will finish, or none of them of will finish execution
2. Mention the syntax of pthread_create and pthread_ join
3. Define target thread and mention the two scenarios how it can be cancelled
4. Provide any two programming examples to claim multithreading provides better performance than a single-threaded solution.
5. Relate strong semaphore and the need for spinlock.
6. Define monitor and mention its use.
7. Is mutual exclusion guaranteed in counting semaphore state yes or no then justify.
8. Consider a non negative semaphore S. 20P(S) and 14 V(S)operations are performed on S in some order, where P(S) decrements S and V(S) increments. Estimate the largest initial value of S that will keep at least one process blocked.
9. Consider a system having m resources of the same type. These resources are shared by 3 processes A, B and C which have peak demands of 3, 4 and 6 respectively. For what value of m, deadlock will not occur?
10. Consider the following threads, T1, T2 and T3 executing on a single processor, synchronized using three binary semaphore variables, S1, S2 and S3, operated upon using standard wait( ) and signal( ). The threads can be context switched in any order and at any time.

T_1	T_2	T_3
while (true) { wait (S_3); print ("C"); signal (S_2); }	while (true) { wait (S_1); print ("B"); signal (S_3); }	while (true) { wait (S_2); print ("A") signal (S_1); }

Find the initial value of S1,S2,S3 that would print the sequence BCABCABCA..... ?

PART B (3 * 10 = 30) Answer any three questions

11. Develop the structure of the reader and writer process to achieve synchronization between multiple readers and writers. Mention the challenge posed on writers by allowing concurrent readers and how it can be resolved.
12. Elaborate multithreading models and highlight the benefits of multithreaded programming.
13. Consider a system of 6 process P0 to P5 using 4 types of resources A,B,C and D as 15,6, 9,10 instances of each type respectively. For the given allocation and maximum matrix determine safe allocation is possible. If so mention the safe sequence.

	Current Allocation				Maximum Need			
	A	B	C	D	A	B	C	D
P0	2	0	2	1	9	5	5	5
P1	0	1	1	1	2	2	3	3
P2	4	1	0	2	7	5	4	4
P3	1	0	0	1	3	3	3	2
P4	1	1	0	0	5	2	2	1
P5	1	0	1	1	4	4	4	4

14. With reference to the above question no. 13, determine whether a new request from P5 with (3,2,3,3) instances from each type can be accepted for. If it is possible, generate the safe sequence otherwise estimate the minimum no of additional instances required from each type to handle a deadlock free execution.

SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 of the UGC Act, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

School of Computing
Second CIA Exam –Feb 2024
Course Code: CSE208
Course Name: Operating Systems
Duration: 90 minutes Max Marks: 50

PART A (2 * 10 = 20) Answer all the questions

Q. No	Questions
1.	D/H is correct. The threads may not finish execution because i++ is not guaranteed to be atomic. It can be split into a read, a modify and a store operation. If both threads read i=0 and store i=1 just before the while() loop, both threads will be stuck executing the while(). On the other hand, the threads would finish execution if the i++ statements get executed one after the order, after which "i" becomes 2 and then the while() statements get executed. Coherence would ensure that both threads finish in this case.

2.	int pthread_create(pthread_t *thread, const pthread_attr_t *attr, void *(*start_routine) (void *), void *arg);
	pthread_t *thread: A pointer to a variable where the thread ID will be stored after the thread is created.
	const pthread_attr_t *attr: An optional pointer to thread attributes. Can be set to NULL to use default attributes.
	void *(*start_routine) (void *): A pointer to the function that will be executed as the new thread. It takes a single argument (a void pointer) and returns a void pointer.
	void *arg: The argument passed to the start_routine function
	
	int pthread_join(pthread_t thread, void **retval);
	pthread_t thread: The identifier of the thread you want to wait for.
	void **retval: A pointer to a location where the exit status of the joined thread will be stored. It can be set to NULL if the exit status is not required

3.	A thread that is to be canceled is often referred to as the target thread. Cancellation of a target thread may occur in two different scenarios: 1. Asynchronous cancellation. One thread immediately terminates the target thread. 4.4 Threading Issues 159
	2. Deferred cancellation. The target thread periodically checks whether it should terminate, allowing it an opportunity to terminate itself in an orderly fashion.
4.	A Web server that services each request in a separate thread. A parallelized application such as matrix multiplication where different parts of the matrix may be worked on in parallel. An interactive GUI program such as a debugger where a thread is used to monitor user input, another thread represents the running application, and a show third thread monitors performance.
5.	Semaphore can be implemented in user applications and in the kernel. The process that has been blocked the longest is released from the queue first is called a strong semaphore. A lock that dictates busy waiting is called a spin lock that can avoid starvation
6.	Monitor is a highly structured programming language construct. It consists of private variables and private procedures that can only be used within a monitor. Use of monitors: a) It provides a mutual exclusion facility. b) A monitor support synchronization by the use of condition variables. c) Shared data structure can be protected by placing it in a monitor.
7.	Mutual exclusion is not guaranteed in counting semaphores. Counting semaphores are often used to manage and control access to a pool of resources. The semaphore count represents the number of available resources, and threads can acquire or release resources by manipulating the semaphore count.
8.	Let it be X. X-20+14=-1, then X = 5
9.	In worst case, The number of units that each process holds = One less than its maximum demand So, • Process A holds 2 units of resource R • Process B holds 3 units of resource R • Process C holds 5 units of resource R • Maximum number of units of resource R that ensures deadlock = 2 + 3 + 5 = 10 • Minimum number of units of resource R that ensures no deadlock = 10 + 1 = 11 So, any number of units greater than 11 will ensure no deadlock.

10	In order to get the required output only semaphore S1 should be initialized to 1, other semaphore should be initialized to 0. So, S1 = 1; S2 = 0; S3 = 0

PART B(3 * 10 = 30) Answer any three questions

11.	Pseudocode/algorithm/code of reader process Writer process Challenge : Starvation of writer process due to concurrent read Solution: explanation about writers with priority
12.	Explanation about Many-to-One Model (User-Level Threads): .One-to-One Model (Kernel-Level Threads): Many-to-Many Model (Hybrid Model): (4 marks) With relevant diagram (2 marks) Benefits of multithreading (4 marks) Improved Responsiveness-Parallelism-Concurrency-Fault Isolation-Task Decomposition-Scalability- Economy.
13.	Attached in fig
14.	Attached in fig