SECTION A

TRUE & FALSE QUESTIONS (10 MARKS)

(**<u>INSTRUCTION</u>**: Write your answer either **TRUE (T)** or **FALSE (F)** based on the following statements.)

---

1. A thread also known as a light-weight process can be defined as the smallest unit of execution which can be scheduled and executed [T]
2. It takes far less time to create a new process than to create a new thread. [F]
3. Process that effect or be affected by the execution of another process is called an independent process. [F]
4. Race condition occurs when only one process try to access the global variables. [F]
5. In concurrent processing systems, set of instruction were executed in sequence. [T]
6. Semaphore (S) is an integer variable that is accessed only through stop() and signal(). [F]
7. Test and Set is a hardware support mechanism to perform synchronization in OS. [T]
8. In order to control access to shared resources, semaphore can be used for mutual exclusion implementation. [T]
9. In Round Robin scheduling if the time quantum is very large the scheduling is the same as Shortest Remaining Time First (SRTF). [F]
10. Shortest Job First (SJF) scheduling algorithm could result an aging. [T]

SECTION B

STRUCTURED QUESTIONS (50 MARKS)

(**INSTRUCTION**: Please answer all questions in the space provided)

QUESTION 1 [6 Marks]

a) Which of the followings are shared across threads in a multithreading process? Tick in the appropriate boxes for your answer. [2 marks]

Components	Shared	Not Shared
Program Counter		/
Address Space	/	
Variables	/	
State		/

b) Discuss TWO (2) benefits of using threads. [4 Marks]

QUESTION 2 [17 Marks]

a) Why do we need scheduling in operating system? [1 Marks]

b) Consider the following set of processes in Table 1, with the length of the CPU-burst/cycle/time given in milliseconds and the priority.

Table 1

Process	Arrival Time	CPU Cycle	Priority
A	0	5	3
B	2	4	4
C	3	6	2
D	6	4	1

i. Draw a timeline chart to illustrate the execution of the above processes using **Priority scheduling (preemptive)** (smaller number implies higher priority). [3 Marks]

A B D C

0 5 9 13 19

ii. Complete Table 2 for the **Priority scheduling**. Show your calculation. [4 Marks]

Table 2

Process	Waiting Time	Turnaround Time
A		
B		
C		
D		

Average Waiting Time (AWT):

4

Average Turnaround Time (ATT):8.75

iii. Draw a timeline chart to illustrate the execution of the above processes using S**hortest Remaining Time (SRT).** [3 Marks]

iv. Complete Table xx for the S**hortest Remaining Time (SRT)** scheduling. Show your calculation. [4 Marks]

Table 3

Process	Waiting Time	Turnaround Time
A		
B		
C		
D		

Average Waiting Time (AWT):

Average Turnaround Time (ATT):

v. Compare and analyse the results between the above scheduling. [2 Marks]

QUESTION 2 [13 Marks]

a) Regarding the Producer-Consumer problem, state the function: [2 marks]

i. Producer process

Produce information

ii. Consumer process

Consume information

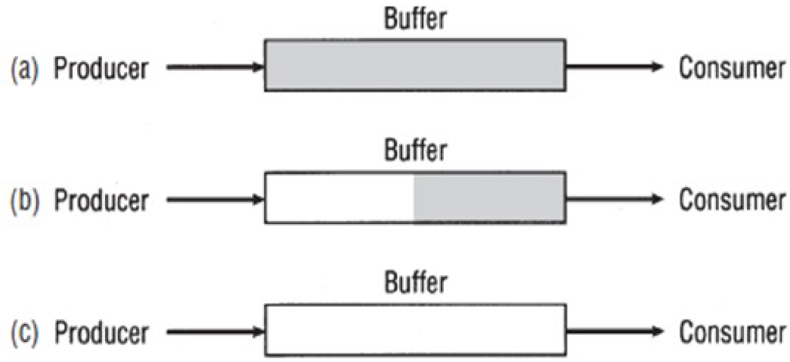
iii. Unbounded- buffer

Unlimited buffer size

iv. Bounded buffer

Fixed buffer size

c) The buffer can be in three states. Draw the buffer states. [3 Marks]



(figure 6.5)

The buffer can be in any one of these three states: (a) full buffer, (b) partially empty buffer, or (c) empty buffer.

d) Peterson's Algorithm: Given a code for Process 0 and Process 1 in Table 4. Fill in flags and turn variables if Process 1 initiates the request to enter the critical section. [4 Marks]

Table 4

Process 0	**Process 1**

do {
    flag[0] = TRUE;
    turn = 1;
    while (**flag[1]** && **turn** == 1)
      ; /* do nothing */
    critical section
    flag[0] = FALSE;
    remainder section
} while (TRUE);
	do {
	    flag[1] = TRUE;
	    turn = 0;
	    while (**flag[0]** && **turn** == 0)
	      ; /* do nothing */
	    critical section
	    flag[1] = FALSE;
	    remainder section
	} while (TRUE);

Table 5

time	**flag[0]**	**flag[1]**	**Turn**	**Events**
t_0	**FALSE**	TRUE	0	P_1 request to enter CS

t_1	FALSE	TRUE	0	P_1 enters CS
t_2	**TRUE**	TRUE	1	P_0 requests to enter CS
t_3	TRUE	**FALSE**	1	P_1 executes RS
t_4	TRUE	FALSE	1	P_0 enters CS
t_5	TRUE	FALSE	1	P_1 executes RS
t_6	**FALSE**	FALSE	1	P_0 executes RS
t_7	FALSE	FALSE	1	P_1 executes RS
t_8	**TRUE**	FALSE	1	P_0 requests to enter CS

e) Justify your answer by showing:

i) Mutual exclusion is preserved [1 marks]

ii) The Progress requirement is satisfied [1 marks]

iii) The bounded-waiting requirement is met [1 marks]

QUESTION 4 [18 Marks]

a) Table 6 shows the implementation of Producer and Consumer using Semaphore.

i. Complete the global variable values in Table xx [6 marks]

Initial values	Producer	Consumer

N = 5 (size of buffer)
mutex =1
empty = 1
full = 0/4
	do {
	produce an item
	...
	wait(empty);
	wait(mutex);
	...
	add item to buffer
	...
	signal(mutex);
	signal(full);
	
	} while (TRUE);
		do {
		...
		wait(full);
		wait(mutex);
		.....
		remove an item from buffer
		....
		signal(mutex);
		signal(empty);
		
		} while (TRUE);

Table 7

Producer		
	wait( )	signal( )
empty	0	
mutex	0	1
full		1

Consumer		
	wait( )	signal( )
empty		1
mutex	0	1
full	0	

i) If the initial value of empty is 0, what happen on the Producer side? Explain. [2 Marks]

Producer cannot enter the critical section due to semaphore empty <=0. wait ( ) is not perform.

ii) Shall Consumer consumes the task if empty is 0? Explain. [2 Marks]

When empty 0 means consumer can proceed because the buffer is full/filled.

b) Based on the following sequence of statements for executing the three processes, state the output for Table 8 to 11. Assume that all flags are initialized to 0. [4 Marks]

Table 8

P_0	P_1	P_2
wait(B_flag) printf("B "); signal(C_flag)	printf("A ") signal(B_flag)	wait(C_flag) printf("C ")

Output : A B C

Table 9

P_0	P_1	P_2
		wait(C_flag)
printf("B ");	printf("A ")	printf("C ")
signal(C_flag)	signal(B_flag)	

Output : BCA/ABC/BAC

Table 10

P_0	P_1	P_2
wait(B_flag)	wait(A_flag)	
printf("B ");	printf("A ")	printf("C ")
signal(A_flag)		signal(B_flag)

Output : CBA

Table 11

P_0	P_1	P_2
wait(B_flag) printf("B "); wait(C_flag)	printf("A ") signal(B_flag)	printf("C ") signal(C_flag)

Output : ABC/CAB/ACB

--End of Question --