

Question – 1B

Print the following Pattern

A 1 a B 2 b C 3 c ... Y 25 y Z 26 z


Using any one of the following concepts

b. Multithreads (Hint: using 3 Threads)

(PS: Process Synchronization or Thread Synchronization is key thing for pattern printing)

We have to implement 3 threads one prints Uppercase, second prints lower case alphabets and the other one prints numbers from 1-26. We have to implement it in a way that it shows output as A,a,1,B,b,2...

Query about a certain condition If state permits chars, caller will put chars. if state allows nums, do it now and alter state to wake up waiting threads. If not, you should wait. If the state enables numbers and the caller intends to use them, update the state immediately and wake up the waiting threads. If not, you should wait.



Question – 2A

Describe and implement any one of the following [20]

a. Describe the Round Robin (RR) and Modified Round Robin (MRR)

Algorithm. Also mention the difference between the results of both the algorithms and implement them both in C.

Round Robin (RR) – Round robin is pre-emptive algorithm. RR shares equal amount of time between processes. RR is starvation free scheduling algorithm and it is fairest algorithm.

- We start with a queue in which the processes are sorted in order of first come, first served.
- Each process has a quantum value assigned to it.
- The first procedure runs until the quantum value reaches its limit. The status is then stored and an interrupt is produced.
- The CPU then goes on to the next process, which is handled in the same way.
- The same stages are repeated until all of the processes have been completed.

```

D:\OneDrive\Desktop\Q2a.exe
Enter all Details of Process[1]
Arrival Time -0
Burst Time - 4

Enter all Details of Process[2]
Arrival Time -2
Burst Time - 6

Enter all Details of Process[3]
Arrival Time -4
Burst Time - 8

Enter all Details of Process[4]
Arrival Time -5
Burst Time - 10

Enter Time Quantum - 2

Process ID          Burst Time      Turnaround Time      Waiting Time
P[1]                4              10                   6
P[2]                6              16                   10
P[3]                8              20                   12
P[4]                10             23                   13
Average Waiting Time- 10.250000
Avg Turnaround Time- 17.250000n
-----
Process exited after 30.79 seconds with return value 0
Press any key to continue . . .

```

Modified Round Robin –

- The number of Processes is entered by the user.
- For each process, the user specifies Burst Time, Arrival Time.
- Set the beginning time to 0.
- The algorithm arranges the processes in the order in which they arrive.
- The algorithm chooses the processes that will arrive first. The arrival time is less than the current time.
- Each selected process is given a score by the algorithm.
- This is how the processes that have arrived are sorted according to score.
- Time Quantum is calculated by the algorithm according to the processes that were chosen
- Determine the Time Quantum mean and median. The duration of bursts is computed.

D:\OneDrive\Desktop\Q2aM.exe

```
Enter Number of processes - 3
Enter Burst Time of Process [1] - 4
Enter Arrival Time of process [1] - 0
Enter Burst Time of Process [2] - 6
Enter Arrival Time of process [2] - 2
Enter Burst Time of Process [3] - 8
Enter Arrival Time of process [3] - 5

Calculated Dynamic Quantum time - 7
Processes ID    Waiting time    TurnAround Time
P[1]           0                4
P[2]           4                10
P[3]          10                18

-----
Process exited after 13.46 seconds with return value 0
Press any key to continue . . .
```

Question 3

- A fixed-size buffer serves as a queue for both the producer and the consumer. It is the producer's responsibility to generate data and store it in the buffer. The consumer's role is to consume this buffer's data one by one.
- How can you ensure that the producer does not try to put data in the buffer when it is full, and that the consumer does not try to consume data when the buffer is empty?
- When a producer tries to insert data into a buffer that is already full, cpu cycles are wasted. Consumer does the same thing when it attempts to consume from an empty buffer. In these circumstances, it's preferable that they go to sleep so that the scheduler may schedule another task.

D:\OneDrive\Desktop\Q3.exe

```
Producer 1: Insert Item 41 at 0
Producer 2: Insert Item 41 at 1
Producer 3: Insert Item 41 at 2
Producer 1: Insert Item 18467 at 3
Producer 4: Insert Item 41 at 4
Consumer 1: Remove Item 41 from 0
Consumer 2: Remove Item 41 from 1
Consumer 3: Remove Item 41 from 2
Consumer 1: Remove Item 18467 from 3
Producer 2: Insert Item 18467 at 0
Consumer 4: Remove Item 41 from 4
Producer 2: Insert Item 41 at 1
Producer 3: Insert Item 41 at 2
Producer 3: Insert Item 18467 at 3
Consumer 5: Remove Item 18467 from 0
Producer 4: Insert Item 41 at 4
Consumer 2: Remove Item 41 from 1
Consumer 3: Remove Item 41 from 2
Consumer 1: Remove Item 18467 from 3
Producer 5: Insert Item 41 at 0
Producer 1: Insert Item 6334 at 1
Producer 4: Insert Item 18467 at 2

-----
Process exited after 2.08 seconds with return value 0
Press any key to continue . . .
```

