SET: B

EAST WEST UNIVERSITY BANGLADESH
Department of Computer Science & Engineering

CSE325: Operating System (1)

Midterm II Exam — FALL 2016

Total Marks: 30 | Instructor: Dr. Md. Shamim Akhter | Time: 90 minutes

1.

a) What are the **four requirements (criteria, properties)** of good solution to the critical-section problem? (3)

b) Following two processes are running concurrently in a shared memory single-processor system and supports atomic **load and store instructions**. X is the shared variable and initialized to 0, and X must be loaded into a register before being incremented (and stored back to memory afterwards).
What can be the highest possible value of X after completion of both processes? Give explanation for your answer. (3)

Process A	Process B
`for(i=0; i<5;i++)` `X=X+1;`	`for(i=0; i<5;i++)` `X=X+2;`

c) Does the busy waiting solution with turn variable work well when two processes are running on a shared-memory multiprocessor system (e.g. two CPUs share a common memory)? Explain with figure. (4)

2. By "**Multilevel Queue Fixed priority**" scheduling algorithm, draw the CPU scheduling Gantt chart and complete waiting time table for the given processes information. (10)

	Queue Name	Process	Burst Time	Arrival Time	Algorithm
Highest Priority	1st Foreground	P1	40	0	RR interval:20 Non Preemptive
		P2	15	10	
		P3	45	20	
		P4	10	170	
	2nd Foreground	P5	30	0	SRTF Preemptive
		P6	10	110	
		P7	6	112	
		P8	4	280	
Lowest Priority	Background	P9	30	60	FCFS Preemptive
		P10	20	130	

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Waiting time										

SET: B

3.

a) The arrival time (in second) of each individual process is given in Table-1. A **counting semaphore variable N = 3** and Table-2 has the execution code of all processes. Critical section requires six (6) seconds. Your task is to evaluate **the total waiting time** of all processes to enter the critical section. (4)

Process	Arrival Time
P0	1
P1	2
P2	3
P3	4
P4	4
P5	5
P6	5
P7	6
P8	7

Table-1

Execution Code
Wait (N); **Critical Section** **…..** **…..** Signal (N)

Table-2

b)

i. How does the bakery algorithm (**Figure-1**) ensure- **the process will come first will serve first**, in the following multi-process critical section solution.
Note that the bakery algorithm cannot guarantee that two processes receive the same number. (3)

ii. Why does this algorithm need to test **while(choosing[j]);** at line 7. (1.5)

iii. Why does this algorithm need to test **(number[j]!=0)** in the while statement at line 8.(1.5)

```
1.  do{
2.      choosing[i]=true;
3.      number[i]=max(number[0], number[1], … , number[n-1]) +1;
4.      choosing[i]=false;

5.      for(j=0; j<n; j++)
6.      {
7.          while(choosing[j]);
8.          while( (number[j]!=0) && (  (number[j],j)<(number[i],i)  )  );
9.       }

10.      //Critical section

11.       number[j]=0;

12.  } while(1);
```

Figure-1