CSE321: Operating Systems
Quiz-2

Name: ____________________________ **ID:** _________ **Section:** ____

Q01) Explain what is Parallelism and Concurrency? [03]

Q02) Explain what is “Many-to Many” Model? In what cases would you use such model? [03]

Q03) What is “Race Condition”? [02]

Q04) Show Peterson’s solution for the given scenario.

- There are two processes: P_1 and P_2.
- Each Statement takes 2 ms to execute.
- Context Switch will occur after 10 ms.
- Critical section contains 6 statements.
- Remainder section contains 4 statements.
- For P_1: i=1 and j=0
- For P_2: i=0 and j=1
- turn=1
- flag[0] = FALSE, flag[1] = FALSE
- P_1 starts the execution first.

The structure of process P_i in Peterson’s solution.

```
do{
      flag[i] = true;
      turn = j;
      while(flag[j] == true && turn == j){
            //busy wait
      }
      //critical section
      flag[i] = false;
      //remainder section

}while(true);
```

Complete the table given below for processes P_1 and P_2 using Peterson’s solution.

Process 1: i=1, j=0	Process 2: i=0, j=1

flag[1]=True
turn=0
While (flag[0] & turn==0);
Cs1
Cs2	

	flag[0]=True

	turn=1 While (flag[1] & turn==1); → stuck
Cs3 Cs4 Cs5 Cs6 flag[1]=False	
	While (flag[1] & turn==1); Cs1 Cs2 Cs3 Cs4
Rs1 Rs2 Rs3 Rs4	
	Cs5 Cs6 Rs1 Rs2 Rs3
	Rs4

CSE321: Operating Systems
Quiz-2

Name: ______________________________ **ID:** __________ **Section:** ____

Q01) Name and explain Three thread issues. [03]

Q02) Explain what is “One-to-One” Model? In what cases would you use such model? [03]

Q03) What is “Mutual Exclusion” ? [02]

Q04) Show Peterson’s solution for the given scenario.

- There are two processes: P_1 and P_2.
- Each Statement takes 4 ms to execute.
- Context Switch will occur after 16 ms.
- Critical section contains 5 statements.
- Remainder section contains 3 statements.
- For P_1: i=1 and j=0
- For P_2: i=0 and j=1
- turn=0
- flag[0] = FALSE, flag[1] = TRUE
- P_2 starts the execution first.

The structure of process P_i in Peterson’s solution.

```
do{
      flag[i] = true;
      turn = j;
      while(flag[j] == true && turn == j){
            //busy wait
      }
      //critical section
      flag[i] = false;
      //remainder section

}while(true);
```

Complete the table given below for processes P_1 and P_2 using Peterson’s solution.

Process 1: i=1, j=0	Process 2: i=0, j=1

	flag[0]=True
	turn=1
	While (flag[1] & turn==1); → Stuck

flag[1]=True
turn=0
While (flag[0] & turn==0); → Stuck	

	While (flag[1] & turn==1); → false (turn=0) Cs1 Cs2 Cs3
Stuck	
	Cs4 Cs5 flag[0]=False Rs1
While (flag[0] & turn==0); Cs1 Cs2 Cs3	
	Rs2 Rs3
Cs4 Cs5 flag[1]=False Rs1	
Rs2 Rs3	