CSE/IT Batch : Hinglish

Operating System

Process Synchronization Part - 1

DPP-01

1. Among the below three critical section problem solutions If we want to say, solution is to be perfect to critical section, then that solution must satisfy.

1. Mutual exclusion
2. Progress
3. Bounded waiting

(a) only 1 (b) 1 & 2
(c) 2 & 3 (d) all 1, 2, & 3

2. Two processes A and B needs to access a critical section. Consider the following synchronization construct used by both processes.

Process A

```
    /* other code of process A*/
       while (true)
         {
           varA = true;
           while(varB = = true)
         {
              /* critical section */
                varA = false;
         }
       }
```

Process B

```
    /* other code for process B*/
       while (true)
         {
           varB = true;
           while(varB = = true)
         {
              /* critical section */
                varB = false;
         {
    }
  /* other code of process B*/
```

Note that, varA and varB are shared variables and both are initialized to false, which one of the following statements is true?

(a) The given solution fails to prevent deadlock and also fails to guarantee mutual exclusion.
(b) The given solution prevents deadlock but fails to guarantee mutual exclusion.
(c) The given solution guarantee mutual exclusion and prevents deadlock.
(d) The given solution guarantees mutual exclusion but fails to prevent deadlock.

3. Consider the following algorithm as a solution to the critical-section problem. Find whether all the three necessary conditions are satisfied by below algorithm or not.

Boolean Flag[i] = Flag[j] = False initially

Process (P_i) **Process (P_j)**

```
do {                          do {
1. Flag [i] = true;           1. Flag [j] = true;
2. While (flag[j]);           2. while (flag(i));
3. /* critical solution */    3. /* critical solution */
4. Flag [i] = False;          4. Flag [j] = False;
5. Remainder Section;         5. Remainder Section;
} while (1);                  } while (1);
```

(a) ME is guaranteed.
(b) ME, progress is guranteed.
(c) ME, and Bounded Waiting is guaranteed but progress is not guaranteed.
(d) All the three (ME, progress and bounded waiting) is guaranteed.

4. Two processes, x_1 and x_2 needs to access a critical section of code. Consider the following synchronization construct used by the processes.

```
/* x1 */                        /* x2 */
While (true)                    While (true)
{                               {
syncA1=true;                    syncA2=true;
while (syncA2==true);           while (syncA1==true);
/* critical section */          /* critical section */
syncA1=false;                   syncA2=false;
}                               }
/* Remainder section */         /* Remainder section */
```

Here, $syncA_1$, and $syncA_2$ are shared variables, which are initialized to false. Which one of the following statements is true about the above construct?

(a) Above constructs doesn’t prevent deadlock, but ensures mutual exclusion.

(b) It does not guarantees M.E.

(c) It does not guarantee B.W.

(d) It requires that processes enter the C.S in strict alternation.

5. Inter process communication

(a) Helps processes to synchonize activity.

(b) It is not helpful.

(c) Is required to all processes.

(d) None

6. Choose the correct statements from the following

(i) Inter process communication is used to exchange data between multiple processes.

(ii) Shared memory is a memory, shared among only two processes.

(iii) IPC method helps to speed up modularity.

(iv) None

(a) (i) & (ii)

(b) (ii) & (iii)

(c) (iii) & (i)

(d) (iv) only

Answer Key

1. **(d)**
2. **(b)**
3. **(c)**
4. **(a)**
5. **(a)**
6. **(c)**

Hints and solutions

1. **(d)**
If we want to say a solution is perfect to critical solution problem then all the three conditions (mutual exclusion, progresses and bounded waiting) must be satisfied.

2. **(b)**
As we can see both the process can enter cs together hence mutual exclusion is violated. As both process can enter into critical section.

3. **(c)**
As we can see, after the preemption in first process (P_i) after line 1, process (P_j) starts execution and gets executed. Process P_j gets preempted after line 2 and hence it creates a deadlock situation hence progress get violated.

NOTE: If deadlock is occurred then surely progress get violated.

4. **(a)**
Based on strict alternation, if both x_1 and x_2 processes makes $syncA_1$ and $syncA_2$ concurrently to true, no one can enter C.S and hence Busy-waiting always in the loop.

5. **(a)**
Synchronization is important part of IPC which helps process to synchronize activity.

6. **(c)**
Shared memory is a memory shared between two & more processes.