<u>ASSIGNMENT – 5</u>

Critical Section

Name- Ashutosh Soni
Id- 2018ucp1505

Question: Explain Lamport Bakery algorithm in detail justifying as to how it meets all four requirements for a solution to Critical Section Problem in N-processes.

Answer:

This algorithm satisfies

1) Mutual Exclusion Principle.
2) Free from Starvation. (No infinite postponement)
3) Free from Deadlock.
4) No strict alternation

- **<u>Mutual Exclusion Principle</u>**

For Entering into critical section this condition needs to be satisfy...

(ticket[i] < ticket[x]) or (ticket[i] == ticket[x] and i > x)

for entering into the critical section. This condition says that either the ticket of this process is minimum form all the left process or if equal to some process then the index of this process is less than all the left process having equal ticket number.

The above condition can never be simultaneously true for more than one processes So this depicts the Mutual exclusion principle.

Let's elaborate it for two processes…

Let the two process are P and Q and ticket numbers are np and nq respectively which is initially 0 Now both processes increment its ticket number to 1 by choosing the minimum ticket number of all the process either of them gets 1 and 2 on the condition which one enters first then one with lower number get entry and make ticket to 0. Then like wise they proceed no two process get simultaneously entry to critical section. This is the clarification of holding mutual exclusion principle.

- **Freedom From starvation.**

  As we have set of ticket numbers form them there is minimum number among them and if that number is equal to other number then in that case, we can judge according to their index number and we will definitely get some index and ticket number which will definitely satisfies that condition. So, among these set of process there is one such process that will get entry into critical section otherwise they wait for there sequence till that came.

- **Freedom from deadlock**

  From the algorithm,

  It is clear that eventually every process gets the minimum ticket number. As after executing critical section the ticket number becomes 0 and the next time when it further come to execute the critical section it will get the maximum ticket number among all so the condition form which it gets enter into critical section becomes false and let that process enter to critical section.

- **No strict alternation**

  In this algorithm,

  It is not always the case the sequence in which the process gets enter to execute is always same so accordingly they will get different ticket number and they will execute accordingly their critical section here there is no such sequence will generate every time so we can say that here there is no strict alternation in which they execute the process.