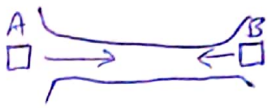
OS & NW

QUIZ - 5

SNEHAL KUMAR
2019101003

Q.1 An old bridge on a busy highway is too narrow to permit two-way traffic, so one way traffic is to.

A: Using semaphores:




```
Semaphore space(INT_MAX)        (no of cars that can be allotted)
Semaphore mutex(1)
enum Direction {LEFT, RIGHT};
Direction currdir = LEFT;
Semaphore direction[Direction];        (initialized to zero)
int activecars = 0, waitingcars = 0;

Direction reverse(Direction A)
{ wait(mutex);
  if (currdir != A && activecars > 0)
  {
     waitingcars++;                    (need to wait)
     direction[A] -> signal();

  }
  else
  {                                    // can travel
     currdir = A;
     activecars++;
     mutex -> signal();
  }
  space -> wait();
}
void exit(Direction A)
{ signal(space);
  wait(mutex);                         (if we are changing the directions)
  if(--activecars == 0)
  { while(waitingcars > 0)
    { waitingcars--;
      activecars++;
      currentdir = reverse(A);
      direction[reverse(A)] -> wait();
    }
  }
  signal(mutex);
}
```

Q.2. Compare the deadlock prevention algo by preventing circular wait with the deadlock avoidance algorithm:

A: A deadlock avoidance algorithm tends to increase the runtime overheads due to the cost of monitoring the current resource allocation.
A deadlock avoidance algorithm requires additional information about the process and these algorithms are usually differing in amount of info required. the Resource allocation state is determines the execution.
A circular wait scheme prevents the formation of deadlock by preventing circular deadlock but is less concurrent than deadlock algorithm.
By this, we find that a deadlock avoidance scheme increases system throughputs

Q.3. Identify whether the following statement is TRUE or FALSE. If the statement is FALSE.

A: This statement is FALSE. Deadlock prevention algorithms limit the no of requests that can be made by a process. these limits ensure that atleast one of the necessary conditions for deadlocks cannot occur. But by this there is lot of overhead and results in overall low device/CPU utilization and reduced system throughput. since resource may be allocated but unused for a long time. Also starvation is possible since a process may need to wait indefinitely.
Thus, the correct statement is: "Deadlock prevention protocol reduce CPU utilization..