Ellike Mishra, 3322)

# Assignment 6

Title :- Deadlock avoidance using semaphores

Problem Statement :- Implement the deadlock-free solution to Dining Philosophers problem to illustrate the problem of deadlock and /or starvation that can occur when many synchronized threads are competing for limited resource.

Theory :- Deadlock is a situation where a set of processes iare blocked because each process is holding a resourse and waiting for another resource acquired by some other process.

Deadlock can arise if the following 4 conditions hold simultaneously :-

- Mutual Exclusion: One or more resource are non-sharable
- Hold and Wait : A process is waiting for resources and holding atleast one resource.
- No Preemption : A resource cannot be taken from a process unless the process releases the resource.
- Circular Wait : A set of process are waiting for each other in circular form

Ellike Mishra, 33221

Algorithm (Semaphore Solution)

```
Process P[i]   // for each Philosopher
while true do
{
  THINK;
  PICKUP(CHOPSTICK[i], CHOPSTICK[i+1 mod 5]);
  EAT;
  PUTDOWN(CHOPSTICK[i], CHOPSTICK[i+1 mod 5]);
}
```

Conclusion : I have successfully implemented the deadlock - free solution to Dining Philosophers problem to illustrate the problem of Deadlock and/or starvation that can occur while many synchronized threads are competing for limited resources.