**Semaphore**

Semaphores ensure the mutual exclusion of data and function when multiple threads/processes try to access a particular resource simultaneously.

Q1

Before starting I have defined:

ssize\_t count:- this is the size of the semaphore to ensure that the object won’t be accessible more than that .

2) pthread\_mutex\_t lock: This allows to lock/unlock the semaphore object

3) pthread\_cond\_t unlock: Some specific conditions may or may not be fulfilled.

Now there are five forks and five philosophers, and they all need a fork to eat. If every philosopher takes a fork along with them, then in this scenario, none of the philosophers would be able to eat. Thus this would be a situation of deadlock.

To avoid Deadlock:

One thing we can do is if one philosopher are with them then there is a high chance we can avoid a deadlock situation by he must take another fork as well which is done by giving only even forks to the philosophers first and subsequently another fork.