This part deals with the Dining Philosopher problem

The dining philosophers problem contains five philosophers sitting on a round table can perform only one among two actions – eat and think. For eating, each of them requires two forks, one kept beside each person.

For implementing these we have simulated the the philosophers using threads, and the forks using global variables.

Part 1a1 - This part deals with the classical dining philosopher problem by strict ordering of resource requests, using mutex to lock the required parts. To manage the resources we make sure that the even numbered philosopher pickup the right fork first and the odd philosopher pick the left fork first to avoid deadlock.

Part 1a2 - This part deals with the classical dining philosopher problem by using semaphores to restrict access tovariables(in this case forks). Whenever the fork is available, any philosopher picks up both the spoons at same time and this is ensured using semaphores

Part 1b - The above system has been repeated only using semaphores now with a system that also has two sauce bowls. The user would require access to one of the two sauce bowls to eat, and can access any one of them at any point of time.