**Baran Çimen**

**26336**

**Barrier Implementation:**

To create barriers, I used the Semaphores which are stored in an array called barriers. Since their initial permits were set to 0 it meant that none of the threads would be able to acquire it and had to wait until someone released them. So, I wrote a loop in the main thread which checked all of the barrier Semaphores if there were any queued threads with the Semaphore class function hasQueuedThreads(). When all of them has queued threads, the main function releases all of the Semaphores one by one, thus, starting the algorithm.

**Dining Algorithm Implementation:**

Threads are in an infinite loop to do 4 main things; thinking, taking the forks, eating and putting the forks back.

**Thinking:**

With an RNG integer, the thread sleeps.

**Taking the Forks:**

In order to take the forks, first the thread enters the critical region by locking the mutex. Then, it changes the state and GUI to HUNGRY. It tries to take the forks from the table by checking if the neighbor threads are EATING or not. If they aren't, it changes it's state to EATING and in the GUI it will take the forks and releases it's respective Semaphore. The reason it releases the Semaphore is if thread fails to take the forks, the thread waits for neighbor threads to release their forks. It does this waiting process after releasing the mutex, by acquiring it's respective Semaphore. If it already has the forks, the Semaphore will easily let it acquire and move on to eating.

**Eating:**

The thread will eat for 3 seconds and change it's GUI to Eating by turning the plate to blue.

**Putting the Forks Back:**

Firstly, the thread will enter the critical region by acquiring the mutex. Then, it will change it's state to THINKING and change it's GUI (StopEating and ForkPut). After that it will check it's neighbor threads if they are HUNGRY and if they are (which means they failed to take the forks and waiting for a release In their respective Semaphores). It will check if those neighbors' neighbors are EATING. If they aren't, it will change the neighbors' state to EATING and change the GUI by taking the forks and it will release their respective Semaphores. Finally, the thread will exit the critical region by releasing the mutex.