**Summary**

In this project a theatre is simulated by using threads to design customer and employee behaviour. The threads communicate using various semaphores which has two operations (signal and wait). The project is implemented in JAVA using Eclipse IDE. One thread for each box office agent, customer, and ticket taker and concession stand worker is used. All the movies and there count is stored in an array which can be used by both box office agent and customer. Also ticket taker, Box office and concession stand have queues to have the count with semaphores which are mutually exclusive. All three queues wait for at least 1 customer and there initial values set to 0. This count is saved using another set of semaphores. An array of semaphores (finish []) is created with size of number of customers. This is used by the agents and workers to signal customers when they are done.  The flow of the project is as follows.

When program starts it reads the contents of a specified file and copies the contents to an array of movies. Then agents and workers are created. These wait for a customer to be enqueue which is done by a counting semaphore. When a customer is created they will choose a random movie from the array of movies and then the customer is enqueue in the box office queue. There are two box offices which will serve customers one by one which work in parallel. So we require mutual exclusion for the array of movies. The ticket availability is shown in a variable for each customer. When the customer gets a ticket the customer will be enqueuer in the ticket taker queue. Customer then waits in the ticket taker queue. Once ticket taker signals customer decides whether he wants to enter the concession stand. If the customer decides against visiting the concession stand customer will enter the theatre to watch the movie. But, if customer decides to go to the concession stand, the customer will select will select what to eat based on a random number. This customer enters the concession stand queue. When this is finished customer enters the theatre with his order. After all customers enter the theatre the customer threads join and execution ends.

Some of the major difficulties encountered are as follows.

Initially I found it difficult to understand the concept of semaphores and how they were used for thread synchronization. My initially design was incorrect as I did not make correct queues. Another difficulty was regarding simultaneous execution of the two box office agent. But the way to create queue and use mutual exclusion solve it.

I learnt many things through this project. I learnt about synchronization and why we need threads need to be synchronized. I gained an understanding of semaphores which is one of the ways thread synchronization is achieved. I also learnt how separate modules can communicate with the help of semaphores. It taught me to understand the problem and to give importance to the overall design before actually implementing the project. I also learnt the importance of unit testing my code.