DESIGN OF MOVIE THEATRE SIMULATION

```
//Design
Semaphore agentQmutex = new Semaphore(1); //Semaphore for mutual exclusion for queue at box
office agent. Its initial value is 1
Semaphore ttQmutex = new Semaphore(1);          //Semaphore for mutual exclusion for queue at
ticket taker. Its initial value is 1
Semaphore cswQmutex = new Semaphore(1);         //Semaphore for mutual exclusion for queue at
concession stand worker. Its initial value is 1
Semaphore moviesMutex = new Semaphore(1); //Semaphore for mutual exclusion for movies and
available seat which are stored in an array. Its initial value is 1
Semaphore agentCheck = new Semaphore(0);    //Box office agent thread checks(with this semaphore)
that if customer is in the queue.
Semaphore cswCheck = new Semaphore(0);          //Concession stand worker thread checks if any
customer is in the queue.
Semaphore ttCheck = new Semaphore(0);           //Ticket taker thread checks if any customer is
in the queue.
Semaphore cusServ ;     finished[noCustomers]; //array of semaphores and one for each customer. Box
office agents and concession workers tell customer when they finish.

void customer()
{
        chooseMovie();
        wait(agentQmutex);
        enqueueAgent(cusId+cusMovieId);
        signal(agentCheck);
        signal(agentQmutex);
        wait(finished[cusId]);
        if(TicketIsAvailable())
        {
                wait(ttQmutex);
                enqueueTt(cusId);
                signal(ttCheck);
                signal(ttQmutex);
                wait(finished[cusId]);
                if(cswVisit)
                {
                        selectSnack();
                        wait(cswQmutex);
                        enqueueCsw(cusId+Snack);
                        signal(cswCheck);
                        signal(cswQmutex);
                        wait(finished[cusId]);
                }
```

```
                    print (customer id +cusId+ has entered theatre);
          }
}

void BoxOfficeAgent()
{
          wait(agentCheck);
          wait(agentQmutex);
          dequeueAgentQ(){
                                                  Customer = agentQ.remove();
                                                  cusId = Integer.parseInt(Customer.split(",")[0]);
                                                  movieID = Integer.parseInt(Customer.split(",")[1]);}
          signal(agentQmutex);
          wait(moviesMutex);
          checkAvailbility();
          signal(moviesMutex);
          sleep(1500);
          print ("Box office agent "+agentId+" sold ticket for "+movieList.get(movieID).movie+" to
customer "+cusId);
          signal(finished[cusId]);
}

void TicketCollector()
{
          wait(ttCheck);
          wait(ttQmutex);
          deQueueAgent(){custId = ttQ.remove();}
          signal(ttQmutex);
          sleep(250);
          print (" Ticket taken from customer"+custId);
          signal(finished[cusId]);
}

void ConcessionWorker()
{
          wait(cswCheck);
          wait(cswQmutex);
          deQueueCsw(){cusId + Snacks}
          signal(cswQmutex);
          sleep(3000);
          print (Snacks+" given to customer "+ cusID);
          signal(finished[cusID]);
}
```