

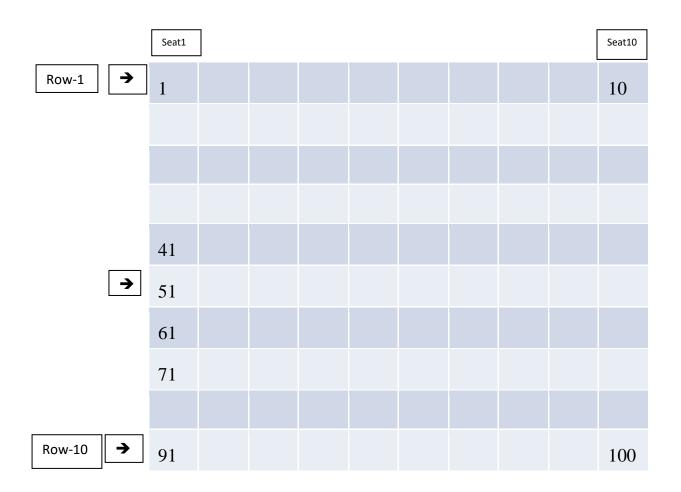
## INTERNATIONAL TECHNOLOGICAL UNIVERSITY

2711 N. 1st St., Santa Jose, California 95134 Phone: (888) 488- 4968 https://itu.edu

Project-3 Preview (10 pts)
Instructor: Ahmed Ezzat

## **Multi-threaded Ticket Sellers**

We will build simulation written in C or Java programming language that experiment with 10 ticket sellers to 100 seats concert during one hour. Each ticket seller has their own queue for buyers.



H1 Seller: one seller and sell tickets starting from row-1

(M1, M2, M3) Sellers: 3 sellers start selling from row 5 then 6, then 4, then 7, etc.

(L1, L2, L3, L4, L5, L6) Sellers: 6 sellers start selling from row 10 then row-9, etc.

Each seller has in their queue N customers arriving at random time during the one hour; where N is a command-line option.

No 2 sellers can sell the same seat to different customers

Time for H-Seller to complete the ticket sale is random either 1 or 2 minutes.

Time for M-Seller to complete the ticket sale is random either 2 or 3 or 4 minutes.

Time for L-Seller to complete the ticke sale is random either 4 or 5 or 6 or 7 minutes.

## **Simulator:**

- 1. Assume 10 threads, each represents a ticket seller: H1, M1, M2, M3, L1, L2, L3, L4, L5, L6.
- 2. Each seller has their own queue and customer stands in the queue using FIFO.
- 3. Initialize the simulation clock initially to zero (0:00)
- 4. Create 10 threads and suspend them.
- 5. Think of the simulation as main() that include:
  - a. Create the necessary data structures including the 10 queues for the different sellers and initialize each queue with its ticket buyers up front based on the N value.
  - b. Create the 10 threads and each will be set with a sell() function and seller type as arguments.
  - c. Wakeup all 10 threads to execute in parallel; wakeup\_all\_seller\_threads();
  - d. Wait for all threads to complete
  - e. Exit

```
#include <stdio.h>
#include <pthread.h>
pthread_cond_t cond = PTHREAD_COND_INITIALIZER;
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
// seller thread to serve one time slice (1 minute)
void * sell(char *seller_type)
  While (having more work todo)
       pthread_mutex_lock(&mutex);
       pthread_cond_wait(&cond, &mutex);
       pthread_mutex_unlock(&mutex);
      // Serve any buyer available in this seller queue that is ready
      // now to buy ticket till done with all relevant buyers in their queue
       // thread exits
  return NULL;
}
void wakeup_all_seller_threads()
  pthread_mutex_lock(&mutex);
  pthread cond broadcast(&cond);
  pthread_mutex_unlock(&mutex);
int main()
  int i;
  pthread t tids[10];
  char Seller_type;
  // Create necessary data structures for the simulator.
  // Create buyers list for each seller ticket queue based on the
  // N value within an hour and have them in the seller queue.
  // Create 10 threads representing the 10 sellers.
  seller-type = "H";
  pthread_create(&tids[0], NULL, sell, &seller-type);
  seller-type = "M";
  for (i = 1; i < 4; i++)
```

```
pthread_create(&tids[i], NULL, sell, &seller-type);
seller-type = "L";
for (i = 4; i < 10; i++)
    pthread_create(&tids[i], NULL, sell, &seller-type);

// wakeup all seller threads
wakeup_all_seller_threads();

// wait for all seller threads to exit
for (i = 0; i < 10; i++)
    Pthread_join(&tids[i], NULL);

// Printout simulation results
......
exit(0);</pre>
```

}