



TICKET BOOTH

An operating system project on ticket counter using thread pool in c++

Project member 1: Akshara Gupta 211340

Project member 2: Shivam Thakur 211365

Submitted to - Dr. Monika Bharti
Assistant Professor (SG) , Dept. of CS & IT JUIT, Waknaghat

INTRODUCTION

Ticket booth is an operating system project which uses thread concept. In this project we have made a ticket counter in which customers will be arriving and buying tickets.

After each buy we will be providing the details of the tickets like their total cost, movie name, customer id.

And at the end this project will be displaying the sales insights for the administrator of the theater for daily records.

AIM & OBJECTIVE

The aim of our project is to make a ticket counter using thread pool and c++ language.

Our objective is to make an application which runs smoothly with increase in number of customers and to calculate daily sales insight of the ticket counter. We provide an environment where a theater manager can add movies according to his schedule, list them, sell them and can also get daily insights of his sales.

TOOLS USED

Threads

To run sequence of instructions that can be executed concurrently with other such sequences in multithreading environments, while sharing a same address space.

Constructor & Destructor

To allocate and deallocate memory space as per need

Queue

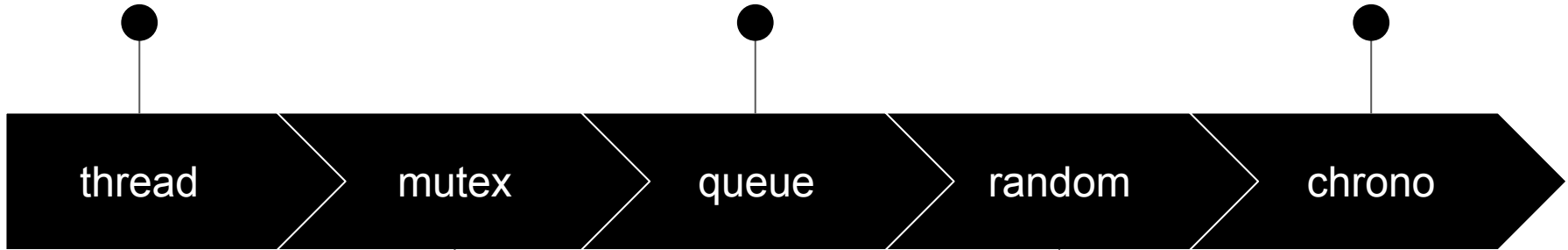
This data structure is used in threads and so in our project to implement the first come customer first out customer

LIBRARIES USED

It is used to destruct a thread and used to get thread id.

Adaptor that gives the programmer the functionality of a queue - specifically, a FIFO (first-in, first-out) data structure.

Header that provides a collection of types and functions to work with time.



Used to protect shared data from being simultaneously accessed by multiple threads.

This library allows to produce random numbers using combinations of generators and distributions.

FEATURES

Interactive portal

First in first out

Regular insights of
sales

Easy to access

Manageable

Constructor

```

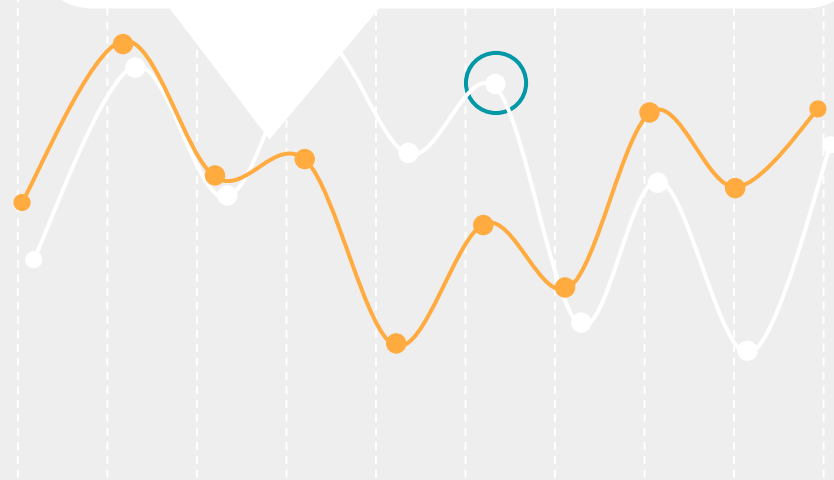
class MovieTheater {
public:
    MovieTheater (size_t num_threads) {
        for (size_t i = 0; i < num_threads; ++i) {
            workers_.emplace_back ([this] {
                while (true) {
                    function < void () > task; {
                        unique_lock < mutex > lock (mutex_);
                        condition_.wait (lock,[this] {
                            return stop_||
                                if (stop_ && tasks_.empty
                                ){}

                                return;
                                }
                                task = move (tasks_.front ()); tasks_.pop ();}
                                task ();}
                                });
        }
    }

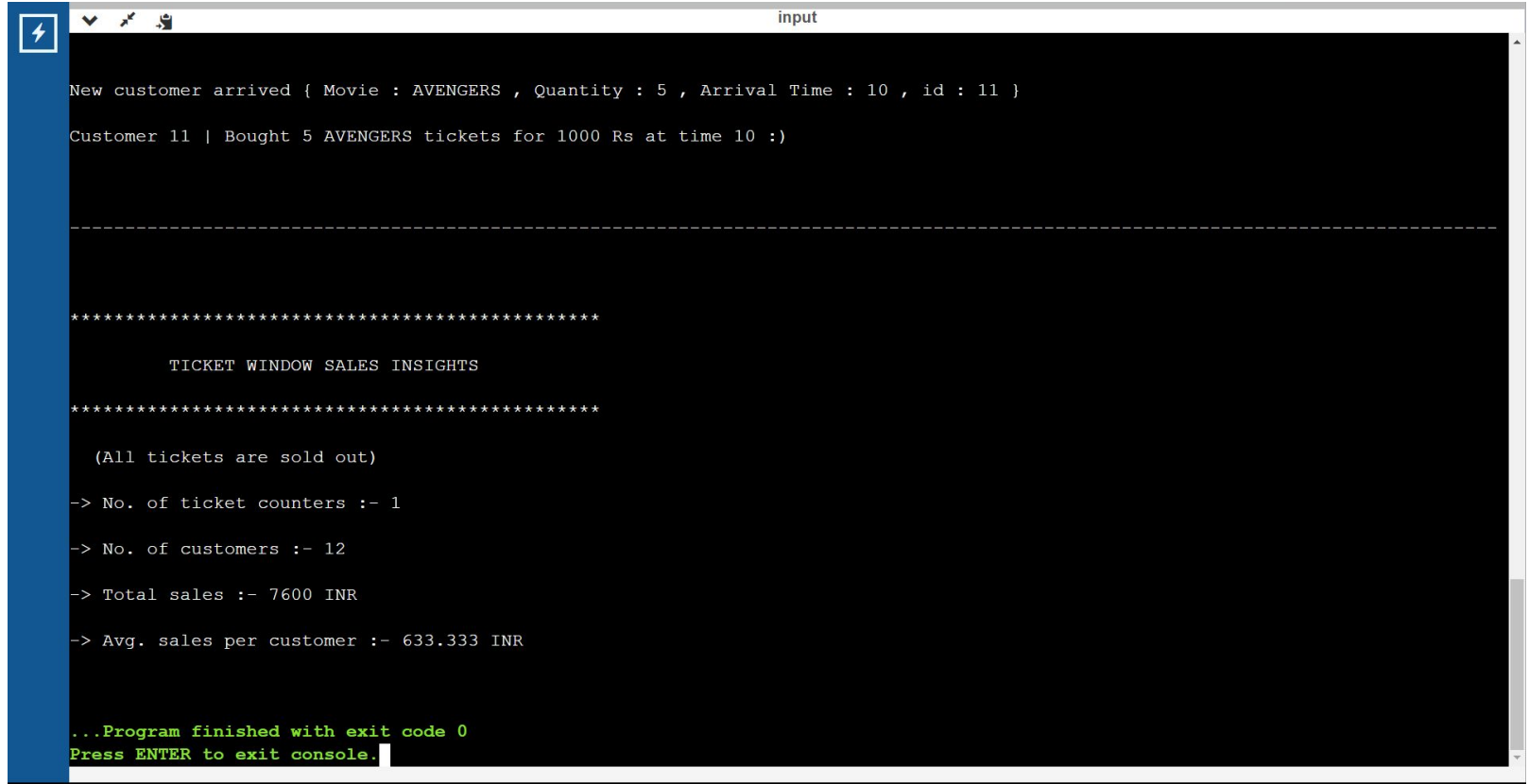
    ~MovieTheater (){
        {
            unique_lock < mutex > lock (mutex_);
            stop_ = true;
        }
        condition_.notify_all ();
        for (thread & worker:workers_)
        {
            worker.join ();
        }
    }
}

```

PSEUDOCODE



OUTPUT

A terminal window titled "input" with a blue sidebar on the left containing a lightning bolt icon. The terminal has a black background with white text. The output shows a customer arrival, a summary of ticket sales, and program completion.

```
input

New customer arrived { Movie : AVENGERS , Quantity : 5 , Arrival Time : 10 , id : 11 }

Customer 11 | Bought 5 AVENGERS tickets for 1000 Rs at time 10 :)

-----

*****

      TICKET WINDOW SALES INSIGHTS

*****

      (All tickets are sold out)

-> No. of ticket counters :- 1

-> No. of customers :- 12

-> Total sales :- 7600 INR

-> Avg. sales per customer :- 633.333 INR

...Program finished with exit code 0
Press ENTER to exit console.
```


CONCLUSION

We would like to conclude our project by thanking our mentor Dr. Monika Bharti Mam and our team mates.

We would be looking forward to grow this project to make it functionalities more better and project productive.

Thankyou