A Command Line Movie Tickets Selling System Implemented in C++

Abstract

To facilitate the process of selling movie tickets in cinemas, a command line tickets selling system for a house of a given cinema was developed. This system can manage various situations including but not limited to sell multiple tickets in a single entry, withdraw sold tickets and calculate the total profit in current stage. The system is easy-to-use and require minimal human inputs. Together with some checking functions to avoid human errors, the system is a good first step of an electronic ticking selling system for commercial use.

Motivation

An electronic system for selling tickets not only increases the efficiency of ticket selling, but also reduces the complexity of management team in profit monitoring and marketing decision making. The target of this project is to develop a starter ticket selling system for future development.

Methodology

Before selling movie tickets, a seating plan of the house must be provided. Such seating plan should be in *m* times *n* matrix format and saved in a text file (.txt) with a single tab as delimiter. The first row is the closest to the screen. For the example seating plan attached (house.txt), it is a house with 16 rows and 9 columns. All available seats should be marked as 0. The seats that are not available, for example corridors, should be marked as -1.

The ticket selling system will create an object with class House for manipulation. Class House provides the following public functions:

1. `showHelp` to display help manual,
2. `setHouse` to initialize the house,
3. `showSeats` to display current ticket selling condition,
4. `buyTickets` to buy 1 or more tickets in single entry,
5. `cancelBooking` to mark the sold seats to available once again,
6. `updateSeats` to update the text seating plan file,
7. `totalProfit` to calculate income based on current ticket selling condition.

Upon successful initialization of the house, the *m* times *n* seating plan will be saved into a vector of length *m* times *n* using library `vector`. Any changes made to the seating plan will be saved to text file immediately after confirmation using `updateSeats`.

Below showed steps to initialize the house, display help manual and seating plan. Fonts marked red are user inputs.

```
D:\cpp>cinema.exe
```

```
Please input txt file name for house:
house.txt
How can I help you? (h for help)
h
b - buy tickets, f = full price, h = half price, e = elderly price
c - cancel booking
h - show this help
q - quit
s - show seats
t - display total profit
How can I help you? (h for help)
s
O: available, X: sold

        -----------------SCREEN----------------
           1   2   3   4   5   6   7   8   9
     A |   O   O       O   O   O       O   O   | A
     B |   O   O       O   O   O       O   O   | B
     C |   O   O       O   O   O       O   O   | C
     D |   O   O       O   O   O       O   O   | D
     E |   O   O       O   O   O       O   O   | E
     F |   O   O       O   O   O       O   O   | F
     G |                                       | G
     H |   O   O       O   O   O       O   O   | H
     I |   O   O       O   O   O       O   O   | I
     J |   O   O       O   O   O       O   O   | J
     K |   O   O       O   O   O       O   O   | K
     L |   O   O       O   O   O       O   O   | L
     M |   O   O       O   O   O       O   O   | M
     N |               O   O   O               | N
     O |               O   O   O               | O
     P |               O   O   O               | P
```

Suppose a customer wants to buy seats H4 to H6, the inputs required by the ticket selling system are as follows:

```
How can I help you? (h for help)
b
Buy tickets
Please enter seats in A1,B2,C4 etc.
H4,H5,H6
Buy H4 - price option (f/h/e):
f
Buy H4 at 100, y to confirm, n to cancel
y
Buy H5 - price option (f/h/e):
h
Buy H5 at 50, y to confirm, n to cancel
y
Buy H6 - price option (f/h/e):
f
Buy H6 at 100, y to confirm, n to cancel
y
How can I help you? (h for help)
s
O: available, X: sold

        -----------------SCREEN----------------
           1   2   3   4   5   6   7   8   9
     A |   O   O       O   O   O       O   O   | A
     B |   O   O       O   O   O       O   O   | B
     C |   O   O       O   O   O       O   O   | C
     D |   O   O       O   O   O       O   O   | D
     E |   O   O       O   O   O       O   O   | E
     F |   O   O       O   O   O       O   O   | F
     G |                                       | G
     H |   O   O       X   X   X       O   O   | H
     I |   O   O       O   O   O       O   O   | I
     J |   O   O       O   O   O       O   O   | J
     K |   O   O       O   O   O       O   O   | K
     L |   O   O       O   O   O       O   O   | L
     M |   O   O       O   O   O       O   O   | M
     N |               O   O   O               | N
     O |               O   O   O               | O
     P |               O   O   O               | P
```

As shown in seating plan, H4-H6 were sold. The text file of seating plan also had corresponding prices of the seats sold saved. Currently, full, half and elderly prices were set as 100, 50 and 20 respectively.

Using similar steps as buying tickets, booking can be cancelled by `cancelBooking`. Finally, `q` can be used to switch off the whole ticket selling system.

Discussion

While the system is ready-to-use with number of useful features, there are some limitations for future development. Firstly, the system is for one house and one movie each time. However, there is no way that a cinema has one and only one house. Also, it will be inconvenient for customers if they can only buy tickets for upcoming movie, but not the movie they interested some time later. Thus, this ticket selling system should be extended to cinema-based, not house-based anymore.

In addition, the seats information is now saved in a text file physically. The amount of such text files will increase dramatically as days pass by. This information should be linked to a SQL database, with each table representing the corresponding house. This can facilitate the management of seats information. And most importantly, no text files will be needed to be saved physically.

Last but not least, the system is now a command line tool. The next step is to introduce a graphical user interface (GUI) for this system to make it more user-friendly. A GUI can not only increase the readability of inputs and outputs of the system, but also reduce human typing error.

Conclusion

A movie ticket selling system was developed for a house in a cinema using C++. While the system has multiple features and ready-to-use, there are still some limitations that can lead to further development. This system can be considered as the first step of developing a real-life movie ticketing selling system for commercial use.