Name: Darien Booth
Date: October 7, 2023
CS 3331 – Advanced Object-Oriented Programming – Spring 2023
Instructor: Dr. Mejia
Assignment: Programming Assignment 2

This work was done individually and completely on my own. I did not share, reproduce, or alter any part of this assignment for any purpose. I did not share code, upload this assignment online in any form, or view/received/modified code written from anyone else. All deliverables were produced entirely on my own. This assignment is part of an academic course at The University of Texas at El Paso and a grade will be assigned for the work I produced.

1. Program Explanation

In this section, explain the overview of the assignment.
What did you do?
How did you tackle the problem?
What techniques did you use to solve the problem?
Did you break the problem into smaller problems? Explain.

For this assignment I was tasked with adding an administrator login, an action that allows them to inquire information from an event and write two new updated customer list and event list csv files. In addition, all previous actions in my program needed to work. This included reading two csv files that stored information about different events (like sports, concerts, and festivals) and information about different customers. Moving the information within a data structure so the any customer on the list is able to purchase a ticket to any event on the list. The user would purchase from 1-6 tickets and after they received an invoice about their overall purchase. To start I created a new class called factory that I used to help create instances of classes I used. The UML class diagram was helpful in seeing how the classes interact and relate to one another. I planned which steps I should start working on and if I could break those steps into smaller parts. Next is my main file which I approached by drawing how the variables and data would flow throughout the program. The UML use case diagram was helpful in seeing what actions the actors can perform in my program. For the administrator login, I decided to create and break down a main menu that would be used by the administrator into two parts (main menu adm and inquire). I did this to capture the main actions with the admin menu and prevent confusion when everything is in one place. Finally, for creating the new csv files I made two new arraylist and filled all information that I would need and wrote them to the file.

2. What did I learn?

What did you learn as a result of this assignment?
How can my solution be improved?
What ideas do I have about another way to solve the problem?
How long did it take me to complete this lab assignment?

One of the things I learned is the usefulness of the factory design pattern. When I plan to make an instance of multiple children that inherit from a parent class, using the factory helps in the creation. Another point I learned is using the string format command that helps when dealing

with doubles that have multiple digits after the decimal. I used this for displaying the balance of a customer and when creating the updated customer csv file. Another use of this command is when displaying the profit as when the amount is large the number may be displayed in scientific notation. As stated in my previous lab report, I believe that another way I can alter my program is using a switch case for the different state of my menu. Since I am creating another admin menu using a switch case for that states could work. I did not do this since I believe that breaking it down was better in helping me see the data flow and preventing confusion. I also still believe using a dictionary or hashmap could work better but with the csv files we have right now my arraylist works. The arraylist is also easier to manipulate. Overall, it took me about six hours to complete.

3. Solution Design

What did I do in this program?
What was my approach to solving this problem?
What data structures did I use? Why?
What assumptions, if any, did I make?

To plan this out, I updated my UML diagrams with how my program would work and all the interactions that can be performed by my code. My first approach to this problem is creating the new design pattern that I will be using for the children of event and venue classes. After making the class, I moved to the main file. This is where I had most of my obstacles. I began creating a new function that identified the user, which can be either a customer or administrator. The next step was to make the main menu for the administrator that asked how the user would like to inquire information on an event (by id or name). For this section, I split it into two parts, the main menu and inquire. For the inquire function I included a switch case that moved to whatever state the user wished to inquire the information by. In addition, before displaying the information, I would compute all information about revenues for seats and profit. The final part was storing the updated information of customers and events in a new csv file. To do this I made two new arraylist. I used arraylist as my data structures, as to me it is the easiest to add and change the values. The first line in my arraylist consists of the headers and then I looped through either event or customer list to store which information I need into parts and add them to the arraylist. I included try-catch blocks where they were needed incase of user input is invalid. When completed I made sure that all actions in previous assignment were still working.

4. Testing

How did I test my program?
Did I use black-box, white-box testing, or both? Why?
Did I test my solution enough? How can my testing practices be improved?
What are the test cases I used?
Did I break my program and use that to improve it?

When it came to testing my program, I performed a small test after completing a part of code to ensure that it is working as intended. For instance, printing out the information of an event after making calculation of remaining seats and profit. I picked different data points in the list also so that I can double check the information was correctly moved or calculated. Also checking that the data is accurate as it travels through the user menu methods. For example, after wanting to inquire event information by the name, the data displayed accurate. I would say I used both

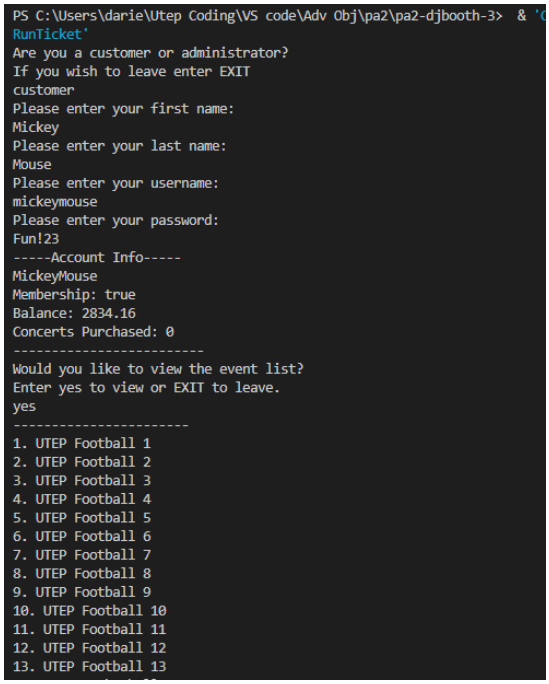
black-box and white-box testing. I used black-box testing mainly after my code was completed so that I could see the behavior of the outputs and fix problems if needed. I used white-box testing when I received an error after completing a part of code, so I can see at which point the code fails. I believe I tested my code enough and something I can improve is the debugging testing I perform. Some test cases I performed were excessive integers and doubles, which helped me improve the program to handle such entries. For the main menu I tested the two pathways that can be taken if the user is a customer or administrator, by the user themselves indicating their role by inputting customer or administrator. For the transition from main menu of admin to inquire, I made sure that the user's choice in how to inquire the information was unchanged and that it was properly executed in inquire. For testing the path of inquire by id, any number input that was out of the range of event ids cause an invalid entry or if a letter was entered then an input mismatch was caught. If the administrator decides on the name path, a statement checks if the name was in events. However, if I entered a name not in events, then the id of the event would remain as zero and therefore invalid. I tested the write to csv files by first testing the headers and then adding the rest of the information. I made sure all other functions in my code worked properly and I tested them in a similar way in my past assignment.

5. Test results

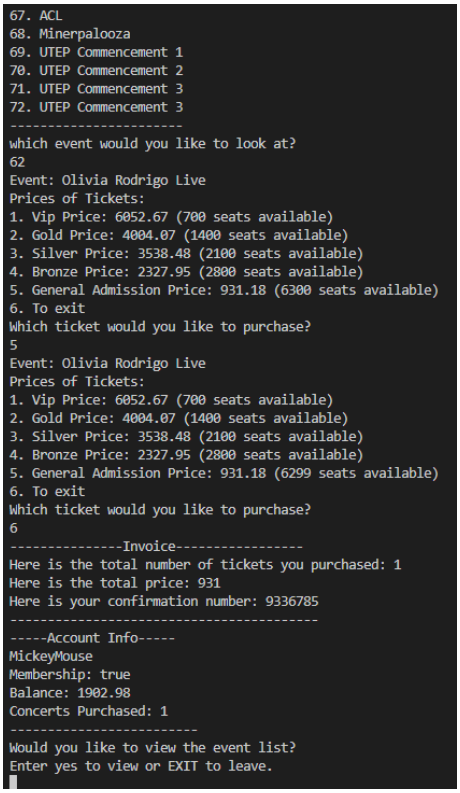
Describe the results of your tests.
Include any console outputs showing your results.
Include any text document results of your tests.

After completing my code, the results that I received from the program were as I expected and to my satisfaction. I attempted multiple entries to see if results changed but they were overall consistent. My scanner read each line and correctly added it to my arraylists. For the user menu, all actions that the customer was able to do worked and any actions are not allowed were denied. I tested and checked the same for administrator. Finally, all actions such as logging-in and purchasing were added to the log file.

Terminal for user menu



→



→ (next page)

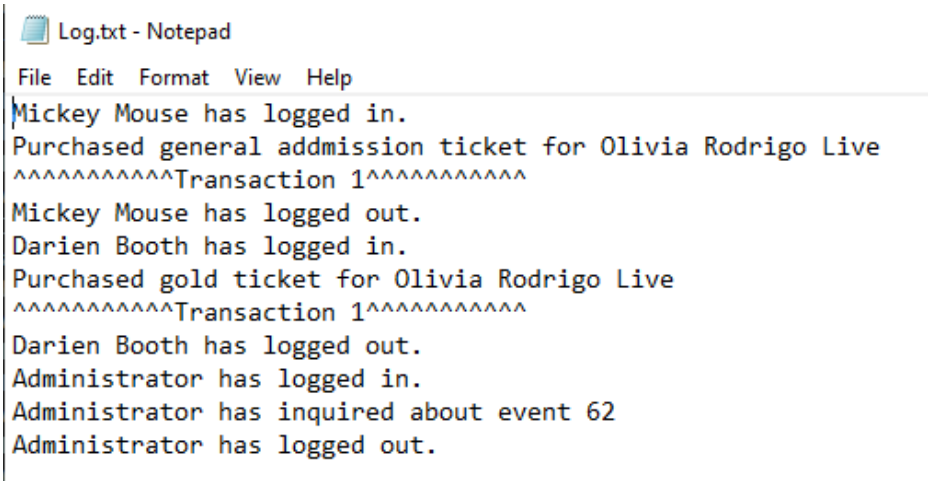
```
Are you a customer or administrator?
If you wish to leave enter EXIT
customer
Please enter your first name:
Darien
Please enter your last name:
Booth
Please enter your username:
darienbooth
Username Incorrect! Please try again.
Please enter your username:
boothdarien
Please enter your password:
Fun!23
-----Account Info-----
DarienBooth
Membership: true
Balance: 5757.35
Concerts Purchased: 0
-------------------------
Would you like to view the event list?
Enter yes to view or EXIT to leave.
```

```
71. UTEP Commencement 3
72. UTEP Commencement 3
----------------------
which event would you like to look at?
62
Event: Olivia Rodrigo Live
Prices of Tickets:
1. Vip Price: 6052.67 (700 seats available)
2. Gold Price: 4004.07 (1400 seats available)
3. Silver Price: 3538.48 (2100 seats available)
4. Bronze Price: 2327.95 (2800 seats available)
5. General Admission Price: 931.18 (6299 seats available)
6. To exit
Which ticket would you like to purchase?
2
Event: Olivia Rodrigo Live
Prices of Tickets:
1. Vip Price: 6052.67 (700 seats available)
2. Gold Price: 4004.07 (1399 seats available)
3. Silver Price: 3538.48 (2100 seats available)
4. Bronze Price: 2327.95 (2800 seats available)
5. General Admission Price: 931.18 (6299 seats available)
6. To exit
Which ticket would you like to purchase?
```

→

```
Would you like to view the event list?
Enter yes to view or EXIT to leave.
EXIT
Bye Bye!
Are you a customer or administrator?
If you wish to leave enter EXIT
administrator
What would you like to do?
A. Inquire event by ID
B. Inquire event by name
EXIT
A
Please enter event ID between 1 - 71:
62
-------------------------------------------
Event ID: 62
Olivia Rodrigo Live
1/14/24
8:00 PM
VIP Price: 6052.67
Gold Price: 4004.07
Silver Price: 3538.48
Bronze Price: 2327.95
General Admission Price: 931.18
Event Type: Concert
Venue Name: Don Haskins Center
Venue Type Arena
Capacity: 14000
Total seats remaining: 13298
Total VIP seats remaining: 700
Total VIP seats sold: 0
Total revenue for VIP tickets: $0.0
Total gold seats remaining: 1399
Total gold seats sold: 1
Total revenue for gold tickets: $4004.07
Total silver seats remaining: 2100
Total silver seats sold: 0
Total revenue for silver tickets: $0.0
Total bronze seats remaining: 2800
Total bronze seats sold: 0
Total revenue for bronze tickets: $0.0
Total general admission seats remaining: 6299
Total general admission seats sold: 1
Total revenue for general admission tickets: $931.18
Total revenue for all tickets: 4935.25
Expected profit (Sale Out): 29493569.00
Actual profit: -159564.75
-------------------------------------------
```

Results of changes stored on log.txt



6. Code Review

Explain how you conducted a review of your code. Describe how you checked each part of the code review checklist.

I reviewed my code and made improvements to the best of my knowledge. I looked at each part separately and together so that I could see what changes could be made. For the implementation section, all actions that were described in the assignment description I make sure I was able to perform those actions in my program. I simplified my code as best as possible by breaking down a large portion of code into smaller parts. An example is the admin menu so that they can inquire information about an event by id or name. I believe my code is mainly dynamic because the admin is able to select which event they wish, as if the event is in the list or number input is in the range. Furthermore, the customer is able to view all events and purchase any type of ticket they wish but they are only restricted to the range of numbers they can input or certain strings they can enter. For making my code maintainable, I tried my best to make it easy to understand and follow how the data flows through each method. I see my code as being consistent and all errors are handled properly. I checked the logic section by testing each functionality of my code. I included exceptions for handling appropriate cases, so I made sure the code did not fail. This can be seen throughout the main menu of customer and administrator, and in the make venue method. For the readability/style, I provided lots of comments and structured my code so that anyone can understand what is happening. The switch statements in my code also are helpful in knowing which states the code is in when a user enters an input. Finally for the performance section, I do not believe my code to have a very high complexity. However, it is not simple/low either. I would say that if a customer were to make one or two purchases the complexity is moderate. However, if the customer jumps around and purchase tickets from different events then the code complexity would grow. The displaying of information would change dramatically if the user were to make multiple ticket purchases. This can also be seen in the updated csv files for customers and events.