Francisco Vazquez
April 4, 2023
CS 3331 – Advanced Object-Oriented Programming – Spring 2023
Dr. Mejia
Programming Assignment 4

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 programming assignment we mainly focused on adding more airport functionality and employee tools. First, I updated my ticket menu for the customer so they could choose between searching for a flight by id or airport codes. If a user selected search by airport codes, they need to provide valid origin and destination airports and the available flights for such will be listed to be selected. I then created an airport menu where employees can search for a specific airport information given the airport code. Then I added an automatic ticket purchasing system using a new method which was like the ticket purchase system method, but this method buys tickets using the information provided in one of the three auto purchaser files. After choosing one of the files, the tickets are bought line by line pending all conditions are met. Finally, I created a menu for the electric ticket summary for the chosen customer. The method I created in the csvwriter class creates a text file displaying all ticket information for all the tickets of a chosen customer.

2. What did I learn?

In this assignment I learned about adding more automation to my code. I was able to create a method that automatically processed 400,000 different transactions with the click of a button. I also learned more about file creation and reading thanks to creating text files for the customers electronic ticket summaries. I can improve my code in general by cleaning it up in many instances, there are many places where I could save many lines in repeating actions that a method could solve. I could also make the csv reader dynamic so that files with different headers can be read without having to change the code. This program took me around 2-3 hours of work each day.

3. Solution Design

The main data structures that I used in this program were hash maps and I also used several while loops. To store the flight, customer, airport, and instructions data from the csv files I used a hash map and multiple while loops to implement the user interface and menus. I also stored the tickets in a hash map to store and retrieve them and cancel them. I assumed that the user first and last names didn't matter as much as the user login as there could be multiple people with the same name and I could get the role from the login information. I

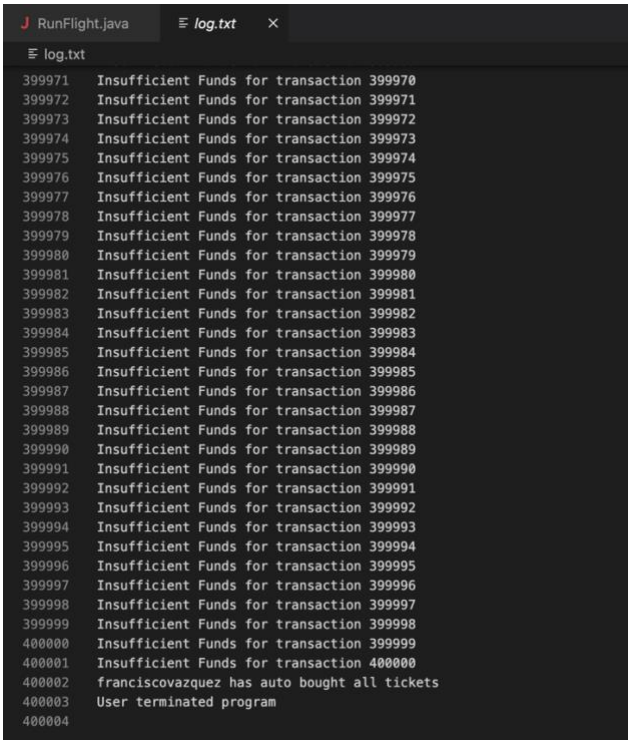
used the username in most of the code as it is easier to handle that two string in the first and last name but in some cases where the username was not available such as the auto buyer file, I matched the names to the customer map to get the username and make things easier. I made more changes to the menus to have as little choices as possible so there was not as much clutter, for example all the flight menus have their own menu separate from the main employee menu where you now only have 4 choices instead of having more of dozen menus in the main menu.

4. Testing

I used a mix of black-box and white-box testing in this program. When a program didn't compile or work correctly, I use white box testing to find the exact error and fix it. I then used black box testing to find errors that would occur when a user does something unexpected such as an incorrect input type. I broke my program into smaller parts, and this allowed me to test specific parts of the program differently such as different user menus, options, and the different classes.

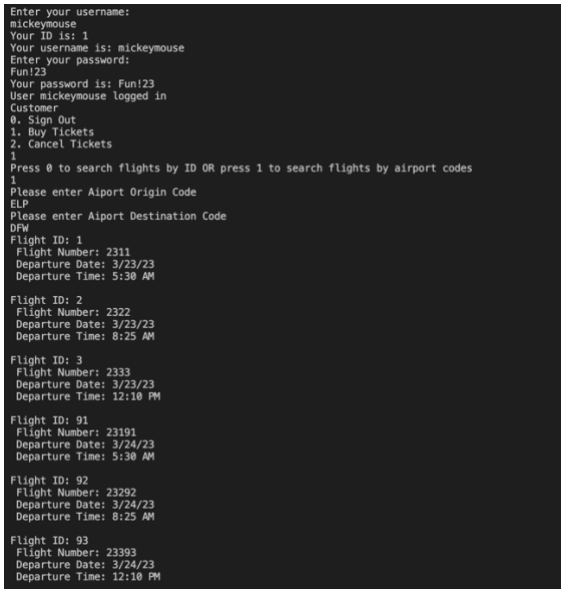
5. Test results

In test case #1 I tested the auto buyer method with the 400k file and checked the logs to make sure they were all processed.



In test case #2 I printed the electronic ticket summary for myself after auto buying.

In test case #3 I tested the flight lookup with airport codes in the customer menu.



6. Code Review

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

For the code review implementation, I checked that my code did what need to be done and it does all of the things described in the pdf. The code is for the most part dynamic, except for reading the files header as I assign the values directly. I believe the code is maintainable as I have documented most of it to explain the methods and what each data structured does.The code can be a little unforgiving in the specific case of updating arrival date/time where an incorrect format will crash the program, however asides from that, for the most part it will ask you to try that action again and will not crash the program. The code is easy to read, however it could be simplified since it is too long now. Methods for duplicate actions could easily save me hundreds of lines. The code is well documented with comments explaining the purpose of almost every big action and contains Javadoc comments. The code complexity doesn't change much with inputs.

Partner Code Review

i. What questions did you have about your partners functionality?

Most of the questions I had about my partners code functionality were mostly about what they wanted me to input in some instances be it an integer or a string.

ii. What concerns do you have about your partners functionality?

The main concern I have with my partners code functionality is handling exceptions as it was very prone to breaking with input mismatch cases.

iii. How did you try to break it?

Mainly went through the different menus and smashed the keyboard to see what would happen if a very random input happened.

1. What test cases did you use?

The test cases I used where mainly making sure that what was asked for was accepted and anything that shouldn't work should break the program.

d. Explain why and how this is a black or white box testing

This is black box testing as the code is not visible and you are only handling the terminal inputs and outputs.