COIS 2240 – Winter 2024

Assignment 2

Deadline: 22nd of March by 11:59PM

Note: The deadline has been pushed back 2 weeks when compared to the Syllabus.

Objective: The objective of this assignment is to cement your knowledge of programming concepts in Java.

Part 1 (Java Programming – 75%)

Program Description: Create a Java application that loads bank account data from a CSV file into objects which are then stored in a list. The application will allow users to interact with this data through a **console-based** menu system. This assignment will improve your ability to work with file I/O, object-oriented programming, collections, and user input handling in Java.

Requirements:

Bank Account Class:

Define a BankAccount class with the following properties:

accountNumber (String)

accountHolderName (String)

accountType (String, e.g., Savings, Checking)

balance (double)

Include appropriate constructors, getter/setter methods, and a toString() method to display account information.

CSV File Structure:

The CSV file bank_accounts.csv (available on Blackboard in the same folder as this assignment file) will have the following structure:

accountNumber,accountHolderName,accountType,balance

12345,John Doe,Savings,1500.50

23456,Jane Doe,Checking,2500.75

Add 5 additional records to the CSV. Prove that you did so in your testing.

Data Loading:

Implement a method loadAccountsFromFile(String filename) that reads the bank_accounts.csv file and creates BankAccount objects for each record. Store these objects in an ArrayList.

User Interaction:

Implement a menu system that allows users to:

View all bank accounts

Search for an account by account number

Deposit money into an account (by account number)

Withdraw money from an account (account balance should not become negative)

Error Handling:

Implement error and exception handling for file I/O operations, invalid user inputs (e.g., non-existent account numbers, insufficient balance), and invalid data in the CSV file.

Testing & Documentation:

Include comments in your code explaining your logic.

Create a class that contains a main method to test your system.

Test all normal use cases, and all the edge cases you can think of (e.g., trying to acess a bank record that does not exist).

Screenshot testing is required. Your testing should follow the following format: Test number, description of the test, expected output, actual output (screenshot).

Part 2 (UML Sequence Diagrams – 25%)

For the bank system data management component above, create a sequence diagram exercise that illustrates the interactions between different parts of the system when a user chooses to deposit money into an account. This scenario will involve the following actors and components:

User: The person using the bank system application.

Main Application: The main class that runs the application and displays the menu.

Account Manager: A component responsible for managing bank account operations, including loading accounts from the CSV file and updating account balances.

BankAccount: The class that represents a bank account.

Steps to Include:

The User selects the option to deposit money from the Main Application menu.

The Main Application prompts the User for the account number and the amount to deposit.

The User inputs the account number and the amount.

The Main Application requests the Account Manager to find the account with the given account number.

The Account Manager searches for the BankAccount object matching the account number.

Once the BankAccount is found, the Main Application requests the deposit operation by providing the amount to the Account Manager.

The Account Manager updates the balance of the BankAccount object.

The Main Application displays a success message to the User.

Note: The software you use to create your UML sequence diagram must be able to export to a PDF or a PNG. Hand drawing your UML sequence Diagram is not permitted.

Submission Instructions:

1- Each class should be a separate code file.
2- Submit the code file for each class, in addition to 1 PDF containing all your screenshot testing and your UML Sequence Diagram.
3- Please DO NOT ZIP your submission, submit each file individually in ONE submission on Blackboard by the deadline (22nd of March by 11:59PM).