CSCI-101 Programming I

Exam 3

Instructions

- Once you leave the testing room you cannot return to continue working on the exam.

If you need to use the restroom, please use it now.

- Please leave all notebooks and electronics (**including cell phones and smart watches**) at the front of the room.
- This is a closed book/closed notes exam.
- You have 50 minutes to complete this exam.
- Partial credit is awarded.
- Please write legibly. If I cannot read your answers, I cannot give you credit.
- Your code must be written to behave as specified.
- You must properly use all identifiers that are explicitly stated.
- Please use proper and consistent coding conventions (spacing, naming identifiers, etc.).
- Please stay in your seat until you are ready to hand in your exam. You may leave when you are finished.

Write a class named **Customer** that satisfies the following requirements.

- The class has fields to store the customer's account number (int), first name (String), last name (String), and outstanding balance (double).
- The class has a single constructor that sets all four fields of the instance to the values passed to it.
- The class has a getter for each field along and a setter for the outstanding balance field.
- The class overrides the toString() method and returns a string containing the values in all of the fields; all separated by commas.

Suppose in a file named **accounts.csv**, each line of the file contains a customer's account number, first name, last name, and outstanding balance (in that order); all separated by commas.

Write a program in a class named **FindCustomer** that satisfies the following requirements.

- The program has a method named **printCustomers** that takes an array of **Customers** as an argument and prints to the screen the information about each customer in the array.

In main, the program does the following:

- The program first creates an array of **Customers** using the data in **accounts.csv**.
- The program prints the information about each customer in the array by calling the **printCustomers** method.
- The program then asks the user to type on the keyboard an account number.
- The program reads in the account number and checks to see if a customer with the same account number exists in the array. If a customer with the same account number exists in the array, the program prints the information about the customer to the screen, otherwise it prints "Customer does not exist".