CS526
Homework Assignment 1

This assignment has two parts and the goal of the assignment is:

- A practice of using basic Java features, including arrays, loops, selection statements, and simple I/O.
- A practice of using Java's linked list and writing a small interactive program.

Part 1 (40 points)

Create a file named *ArrayConrolPractice.java* and write the following methods in it:

- Write a Java method named *sumOfEvenNumbers* that receives an integer *n* and returns the sum of all positive even integers less than or equal to *n*.

- Write a Java method named *allDistinct* that receives an array of integers and determines whether all the integers are different from each other. The method returns *true* if all numbers are different from each other and returns *false* otherwise.

- Write a Java method named *statistics* that receives an array of double numbers, calculates the largest number, the smallest number, and the average of all numbers. The method, then, stores these three statistics, in that order, in an array of size 3 and returns the array.

- Write a *main* method to test the above methods:

    - Invoke *sumOfEvenNumbers* twice once with *n* =100 and then with *n* = 200, and print each result on the screen.
    - Create an array with 20 integers that include some duplicates, invoke *allDistinct* and pass this array as an argument, and print the result on the screen. Repeat the same but this time with an array with all distinct integers.
    - Create an array with 10 double numbers, and invoke *statistics* and pass this array as an argument, and print the result (max, min, average) on the screen. Repeat the same with an array with different double numbers.

Part 2 (60 points)

Your program receives information about students interactively from the user and stores them in Java's *LinkedList*. Then, your program performs a few basic operations on the student list as dictated by the user. Name your program *StudentManagement.java.* A *Student* class is already defined and it is posted along with this assignment (**do not use the *Student* class that comes with the textbook**). You should not modify the given *Student* class. As an example of a simple menu-driven program, *CarManagement.java* file is also posted on Blackboard. You may want to study this file before writing your program.

The requirements are given below:

When your program starts, it must display the following main menu:

Choose an option:

1. Add a student
2. Remove a student
3. Update student GPA
4. Display student information
5. Display all students
6. Exit

If the user chooses option 1, your program performs the following:

- Prompts the user to enter a studentID, and reads it.
- If a student with the given studentID is already in the list, your program displays an appropriate error message, and displays the main menu.
- Otherwise:
  - Prompts the user to enter a name, and reads it.
  - Prompts the user to enter a GPA, and reads it.
  - Creates a student object and adds it to a *LinkedList*, displays an appropriate message indicating a successful operation, and displays the main menu

If the user chooses option 2, your program performs the following:

- Prompts the user to enter a studentID, and reads it.
- If a student with the given studentID does not exist in the list, displays an appropriate error message, and displays the main menu.
- Otherwise, removes the student with the given studentID from the list, displays appropriate message indicating a successful operation, and displays the main menu.

If the user chooses option 3, your program performs the following:

- Prompts the user to enter a studentID, and reads it.
- If a student with the given studentID does not exist in the list, displays an appropriate error message, and displays the main menu.
- Otherwise:
  - Prompts the user to enter a GPA, and reads it
  - Updates the GPA of the student with the new GPA, displays an appropriate message indicating a successful operation, and displays the main menu.

If the user chooses option 4, your program performs the following:

- Prompts the user to enter a studentID, and reads it.
- If a student with the given studentID does not exist in the list, displays an appropriate error message, and displays the main menu.

- Otherwise, displays the student information in the following format:

  Name: John Smith
  Student ID: U1234
  GPA: 3.85

  Then, displays the main menu.

If the user chooses option 5, your program prints all students in the list in the following format:

Name: John Smith
Student ID: U1234
GPA: 3.85

Name: Susan Smith
Student ID: U2345
GPA: 3.92

. . .

Then, displays the main menu.

If the user chooses option 6, your program terminates after displaying an appropriate message, such as "Bye."

Documentation

No separate documentation is needed. However, you must include sufficient inline comments within your program.

Deliverables

Submit your *ArrayControlPractice.java* and *StudentManagement.java* files to Blackboard.

Grading

Part 1: Each method will be tested with different input arguments and 5 points will be deducted for each method if your output is incorrect.

Part 2: 5 operations (menu choices 1 – 5) will be tested in the following way:

1. Add a student: Will be tested twice, once with an existing studentID and once with a non-existing studentID. 3 points will be deducted for each wrong operation.
2. Remove a student: Will be tested twice, once with an existing studentID and once with a non-existing studentID. 3 points will be deducted for each wrong operation.

3. Update student GPA: Will be tested twice, once with an existing studentID and once with a non-existing studentID. 3 points will be deducted for each wrong operation.
4. Display student information: Will be tested twice, once with an existing studentID and once with a non-existing studentID. 3 points will be deducted for each wrong operation.
5. Display all students: Will be tested once and 3 points will be deducted if the output is incorrect.

Points will be deducted if your program does not have sufficient inline comments up to 20 points.