CS101- Algorithms and Programming I

Lab 09

Lab Objectives: `ArrayList`

- For all labs in CS 101, your solutions must conform to the CS101 style guidelines (rules!)

1. Create a project, Lab09_Q1 in your Lab09 folder. Write a Java program that inputs a set of integer values from the user and stores them in an ArrayList. Create a new ArrayList that is the same length as the original list, where each element in the new list contains the sum of all elements in the original list, up to the current element, that are divisible by a number input by the user.

   Your program should define and make use of the following helper methods:

   - `void removeDuplicates(ArrayList<Integers>)`: removes duplicates from the parameter ArrayList
   - `ArrayList<Integers> fillList()`: fills and returns an ArrayList with values input by the user.
   - `void sumDivisible(ArrayList<Integers>, ArrayList<Integers>,int)`: takes 2 ArrayLists as parameters, the first list contains a set of Integer values, the second list should be updated to contain the sum of the elements divisible by the int value passed as a parameter.

   Sample Run:

```
Enter values:
4
16
15
25
230
27
34
19
5
10
6
*
Enter divisor: 5
List: [4, 16, 15, 25, 230, 27, 34, 19, 5, 10, 6]
Sum List: [0, 0, 15, 40, 270, 270, 270, 270, 275, 285, 285]
Sum List (no duplicates): [0, 15, 40, 270, 275, 285]
```

2. Create a Time class with the following data and methods. All attributes should be **private** and should be named appropriately.
   a. Constructor should initialize the hours and minutes to the values passed as parameters.
   b. addTime() which **updates** the Time (hours and minutes) with the minutes passed as an int parameter. E.g. (2 hour and 50 min) + (80 minutes) is (4 hr and 10 min)
   c. lessThan(): that compares two Time objects, and returns True if the time is earlier than the parameter time, False if not.
      toString(): returns a string representation of the time

3. Create a main class with the following functionality:
    - `sortAppointments()`: takes an ArrayList of Time objects as a parameter, and sorts in ascending order. You should use the insertion sort algorithm to sort the list (found at: https://en.wikipedia.org/wiki/Insertion_sort). Use functionality defined in the Time class to compare the Time object.
    - `printAppointments()`: takes an ArrayList of Time objects as a parameter, and displays each Time in the list as shown in the sample run.
    - `main()`:
        i. Inputs a set of Times from the user and stores in an ArrayList. The list represents the schedules appointment times for a doctor's office.
        ii. Print the list of appointments.
        iii. Input a delay start time (`m` for morning, `a` for afternoon) and the number of minutes the appointments will be delayed. In case of a delay, the appointment Times in the list will be shifted. If the delay starts in the morning, all appointment times will be shifted by the number of minutes input by the user. If the delay starts in the afternoon, only the appointments from 12:30 onward will be shifted.
        iv. Sort the list of appointments and display the updated list.

Sample Run:

```
Enter appointment time: 8:40
Enter appointment time: 12:55
Enter appointment time: 9:15
Enter appointment time: 14:25
Enter appointment time: 18:45
Enter appointment time: 23:55
Enter appointment time: 00:15

Scheduled Appointments:
08:40
12:55
09:15
14:25
18:45
23:55
00:15

When will delay start (M)orning / (A)fternoon: M
How many minutes will the doctor be late: 15

Updated appointments:
00:10
00:30
08:55
09:30
13:10
14:40
19:00
```