CS1083 Assignment #0 - Fall 2024

Due: Wednesday, 11 September before 4:30 pm in the Desire2Learn dropbox. (See submission instructions below).

The purpose of this assignment is:

- Review classes and objects (inheritance) and partially-filled arrays

Music Festival Event System

You are working on part of a new system for music festival attendees. Your responsibility is to write two classes, called **`Attendee`** and **`VIPAttendee`**. These keep track of personal information of the attendee and which events at the festival they have gone to. Events are represented by objects of the **`Event`** class, given below. Both types of attendees should store:

- days attended (the festival is held on Thursday, Friday, Saturday, and Sunday – an **`Attendee`** may only attend 1, 2, or 3 days.
- the number of events they have currently visited,
- and a partially filled array of **`Event`** objects representing the events they attended.

There is no maximum number of events that can be attended, but you must be conservative with space – the recommended number of events to consider when instantiating an **`Attendee`** is 5 (therefore you must use 5 to start).

The **`Attendee`** class needs methods to accomplish the following:

1. Add an event (given an **`Event`** object). This method should never fail adding and therefore does not return anything.
2. Remove an event (given an **`Event`** object), returning true or false to indicate if the deletion succeeded.
3. Retrieve a list of all events the attendee is registered for. This list should be the exact length of the number of events they are registered for (in other words, no empty spaces or bad data).
4. A method that returns the cost of the ticket. If an attendee is attending 4 or fewer events, the cost is 189.99 and the cost is 279.99 otherwise.
5. Retrieve a textual list of all events the attendee is registered for, formatted in tab-separated columns as shown in this example.

```
6005  Artist C
6002  Artist A
6010  Artist G
```

Note that the events do not need to be stored in any particular order.

For **VIPAttendee** objects, we require all the functionality that was described above for **Attendee** objects. However, for **VIPAttendee** objects, they attend all four days of the festival and therefore they do not need to pass this in as a parameter, instead, we want to pass in their group attendee status – true if they are attending as a group or false if they are attending alone (this instance variable requires accessor and mutator methods). Also, if a **VIPAttendee** is only attending events by a single artist (match by name, case sensitive), their ticket cost is 209.99 no matter how many events they attend. If the **VIPAttendee** has attended events hosted by different artists, the ticket cost is the same as a regular **Attendee**.

Write the complete **Attendee** and **VIPAttendee** classes, given the following **Event** class:

```
public class Event{
  private String artistName;
  private int id;
  private static int nextID = 6000;

  public Event(String nameIn){
    artistName = nameIn;
    id = nextID;
    nextID++;
  }

  public String getArtistName(){
    return artistName;
  }

  public int getID(){
    return id;
  }
}
```

Notes: You **must use arrays** (and not `ArrayLists`) to store course objects in the **Attendee** and **VIPAttendee** classes. These classes should include only the attributes required to store the values described above. For instance, there is no need to store a ticket price or name of attendee.

Driver Program for Testing

Write a driver program to test the **`Attendee`** and **`VIPAttendee`** classes, covering the following cases:

- Reporting on a mix of regular and VIP attendees
- Events added and dropped, where the current number of events attended never exceeds 5, but the total added + dropped exceeds 5.
- Adding more than 5 events sequentially (causing the creation of a new array and copy over)
- Removing an event that does not exist

Your electronic submission (submitted via Desire2Learn) will consist of two files. Name your files YourName-fileName.extension, e.g. JohnSmith-as0.zip, JohnSmith-as0.pdf:

1. A single pdf file containing a listing of the source code for the **`Attendee`**, **`VIPAttendee`**, and **driver classes**, and a discussion of each test case along with output captured from the terminal window.
2. A zip file containing your three Java classes.