DEPARTMENT OF COMPUTER & INFORMATION SCIENCES

CS 994 OBJECT ORIENTED PROGRAMMING 2018/19

INDIVIDUAL LAB TEST

Duration: 2 hours

Available marks: 100

Contribution to overall mark: This assignment contributes 50% towards your final class mark.

General instructions:

Please read the assignment brief carefully and **attempt all *"Assessed Tasks"***. Even if you do not complete everything, make sure that you submit all your code. This is an **open-book** programming lab test (i.e. you are allowed to use the book/lecture notes, your source code from previous practicals/tutorials, etc.), but it is still **under exam conditions** (i.e. no communication among students is allowed). **The use of web browsing will also be heavily monitored: only resources on MyPlace or external resources that link from MyPlace are allowed (*as detailed in the lab test preparation slides*).**

Aims:

The aim of this assignment is to implement (**in Java**) a number of classes under the paradigm of Object-Orientation.

Learning outcomes:

After completing this assignment, you will have demonstrated experience of:

- understanding and using objects in common object-oriented languages;
- understanding and developing programs using class based object-oriented programming.

[Assignment brief continues on next page]

IMPORTANT - Marking Criteria (breakdown of the 100 available marks):

Your submission will be marked for:

- **Completeness** (i.e. has all required functionality been implemented?), and **correctness** (i.e. does everything work as specified?): As specified by the marks below each ***"Assessed Task"*** – **Total of 90 Marks**
- **Commenting** (i.e. is everything commented as it should?) – **Total 5 Marks**
- **Style** (i.e. code layout, naming conventions, meaningful messages) – **Total 5 Marks**

Submission:

Your lecturer will give you instructions on how to submit your code on MyPlace.

[Assignment brief continues on next page]

Assessed Tasks

Part A: Implementation of classes, collections & iteration

1. Implement a class named **`MusicTrack`** that holds three data fields: the track name, the artist's name, and the track running time (in minutes). Write a constructor that sets all data fields to meaningful default values. Include methods to set and get the values for each data field.

(7 Marks)

2. Implement a second constructor in the **`MusicTrack`** class that accepts three parameters and uses their values to initialize the respective data fields.

(3 Marks)

3. Implement a method in the **`MusicTrack`** class that prints the details of a music track, i.e. the values of all data fields along with some descriptive text.

(5 Marks)

4. In the **`MusicTrack`** class, override (i.e. re-define) the `equals` method from the Java `Object` class:

```
public boolean equals(Object obj)
```

according to the following specification: it returns `true` if the track name **AND** the artist's name are equal; otherwise, it returns `false`. Your implementation **must ignore**: case and any leading/trailing whitespaces.

(15 Marks)

[Assignment brief continues on next page]

5. Implement a class named **`MyMusicCollection`** that holds an `ArrayList` of **`MusicTrack`** objects as a data field. Include a method to get the value of the data field. Implement a method that takes a **`MusicTrack`** object as a parameter and works according to the following specification:

If the list **already** contains the parameter **`MusicTrack`** object, the method should **discard** the parameter and **print** the message "***This track is already in your collection!"*** on the screen. **Otherwise**, the method should add the parameter **`MusicTrack`** object at the end of the list and **print** the message "***Track added successfully to your collection!"*** on the screen.

(10 Marks)

6. Implement a method in the **`MyMusicCollection`** class that takes no parameters and works according to the following specification:

The method **returns** `true` if the list is empty. Otherwise, it **returns** `false`.

(5 Marks)

7. Implement a method in the **`MyMusicCollection`** class that prints the details of all **`MusicTrack`** objects in the list. Your implementation must use a **for-each** loop.

(5 Marks)

8. Implement a method in the **`MyMusicCollection`** class that prints the details of all **`MusicTrack`** objects in the list. Your implementation must use a **while** loop.

(10 Marks)

[Assignment brief continues on next page]

Part B: Abstract data Types & Interfaces

9. Download the file StackADT.java from MyPlace (***under Week #11→Lab test***). Implement a Java class called `ArrayListStack` that implements the `StackADT` interface. Your implementation must: ii) use an **`ArrayList`** and ii) follow the specifications described by the comments in the StackADT.java file.

(30 Marks)

Good luck!!!

[End of assignment brief]