University of Newcastle
SENG1110 Programming Assignment 1 – Semester 1, 2023
Assignment 01 – due date Fri 28 April by 11.59pm (week 8)

MUSIC COLLECTION

Introduction

The objective of this assignment is to implement an object-oriented program using Java, to manage a collection of songs. The assignment 2 will be an extension of this assignment. This assignment can be completed **individually or in pairs**.

Your program will **ONLY** use the concepts we learned in weeks 1 to 6.

Carefully read the specification below. Make sure you have all the information necessary to start writing the program. If you are uncertain of something, do not make assumptions. Instead, post your questions to the "assignment 1" discussion board forum and check it regularly.

Note that you must complete this assignment individually or in pairs. Your solution must be your own work. You may be required to participate in an oral exam for quality assurance, consisting of an interview with one staff and will last approximately 10 minutes. During this time, we will ask you questions about your assignment.

Try to start the assignment as soon as possible. There is a document "HelpToStartAssign1" and some Java files on Canvas, which you can use as a starting point.

Specification

The program will keep track of up to 3 albums, and up to 4 songs each album. Each song has the following information: name, artist, duration (in seconds), and genre (can be only "rock", "pop", "hip-hop", or "bossa nova"). Each album will have a name and up to 4 songs. In program requirements you will find more details.

When run, the program will display a menu of actions to the user, including one that exits the program. Until the user chooses to exit, the menu is displayed again after each action is completed.

The program should have the following functionalities:

1. Will allow the user to create **albums**.
2. Will allow the user to **enter a new song** into an **album**.
3. Will prevent the user from adding a **song** in an **album** if exceeds a time limit of 20min (including all songs).
4. Will allow the user to request the list of all **songs** (and the details of each song) from an **album**.
5. Will allow the user to request a list of all **albums** (including all the songs in each album).
6. Will allow the user to request a list of all **songs** whose duration is under a certain time (in minutes).
7. Will allow the user to request a list of all **songs** of a specific genre.
8. Will allow the user to delete an **album.**
9. Will allow the user to delete a **song** from an **album**.

Your program must give appropriate messages to the user on an attempt to:

- create an album that already exist (two albums are identical if they have the same name)
- create another album when there is no room anymore.
- add a song to an album that is full OR exceed a time limit OR in an album that does not exist.
- add a song that already exist in an album (note that 2 songs are identical if name, artist, and duration are the same).
- delete an album that does not exist.
- delete song that does not exist.
- Request list of albums/songs under a certain criterion (functionalities 4, 5, 6, and 7) and the number of albums/songs is zero.

Program Requirements

Your program should implement three classes, which store the following data:

- `Song.java` – storing the following details about a song.
  - `name` – the name of the song.
  - `artist` – the person/s performing the song.
  - `duration` – the length of the song in seconds
  - `genre` – the genre can be only "rock", "pop", "hip-hop" or "bossa nova"
- `Album.java` – stores up to 4 Songs at a time.
  - `name` – the name of the Album.
  - `song1, song2, song3, song4` – Song objects
  - `totalTime` – the total playing time of all songs stored
  - `MAX_TIME` – a constant which stores the maximum playing time for the album, set to a value of 20 (minutes).
- `SongCollection.java` – stores all albums in the system.
  - `album1, album2, album3` – All current Album objects stored in the system.

All the data components of your classes need to be **private** (this means that you are applying the principles of ***encapsulation***).

Additionally, your classes need to have methods that provide the functionality outlined in the problem description. The only class which should have a **main method** is `SongCollection.java`, which should create an instance of the class SongCollection and call the `run()` method which will have code to provide the user with a menu to allow them to perform any of the tasks outlined in the problem description. The template for this is below. The class SongCollection also will be the only one that will receive inputs and show outputs.

```
public class SongCollection {
	private void run() {
		//This method should control the flow of the program
		//and have all code for accessing the Albums
	}
	public static void main(String[] args){
		SongCollection sg = new SongCollection ();
		sg.run();
	}
}
```

Again, a reminder that your program will **ONLY** use the concepts we learned in weeks 1 to 6.

You can choose TIO or GUI, as seen in lectures. **Marks** will be awarded for: layout, both visual (variable names, indentation) and structural (scope of variables, use of methods); documentation (comments); and ability of the submission to perform as specified. A more detailed marking schema is available.

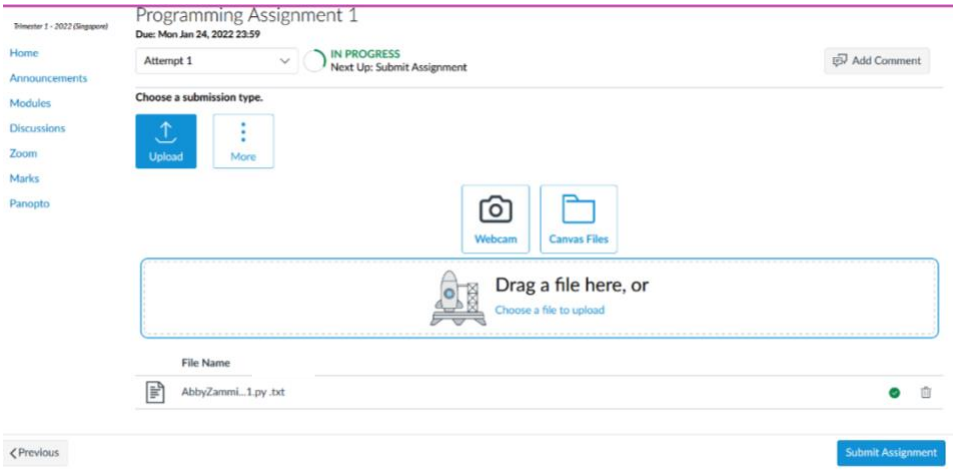
What to submit.

You should submit the Java program (`Song.java, Album.java, SongCollection.java`) via the "Assignment 1" link on Canvas. Do not include .class files in your submission. Add the name of the student(s) on the top of each Java file submitted. If you are completing the assignment in pairs, add both names in each Java file.

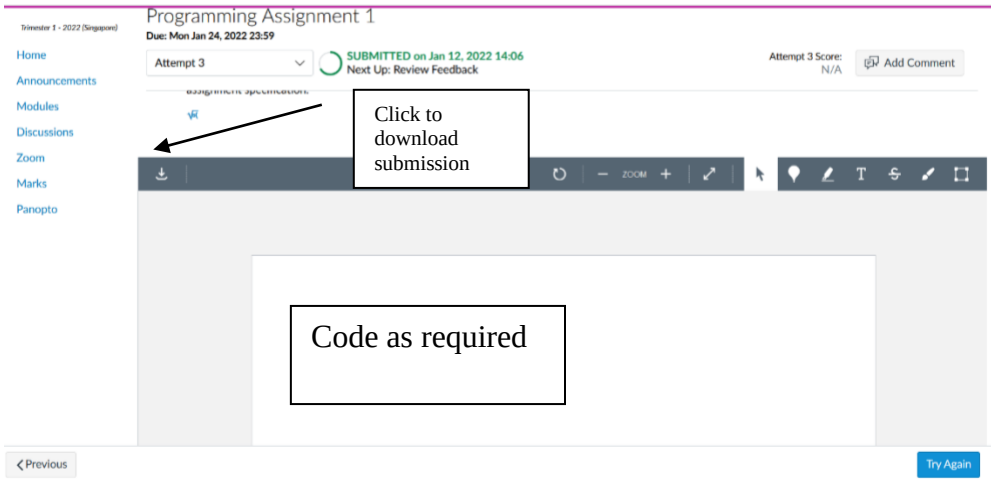
Handing in your work

When you are ready to submit your assignment, log in to Canvas, go to the site for this course, and follow these steps:

- Select the Modules folder.
- Click the Assignment 1 link, which will take you to the appropriate upload page.
- Click or drag the file as below. Then hit Submit Assignment.



- If you wish to see if the correct file has been uploaded then just click the download icon as below.
- If you want to submit an updated version of the assignment, go back to the Assignment link and click Try Again.



- Make sure you're aware of the deadline: the final marking will be applied only to the most recent submission, and if it's submitted late it will be marked as late.

Late Penalty and adverse circumstances

Note that your mark will be reduced by 10% for each day (or part day) that the assignment is late. This applies equally to week and weekend days. You are entitled to apply for special consideration if adverse circumstances have had an impact on your performance in an assessment item. This includes applying for an extension of time to complete an assessment item. See https://www.newcastle.edu.au/current-students/learning/assessments-and-exams/adverse-circumstances for more details.

In the Canvas you will find a new forum in the discussion board: "assignment1". Any question about the assignment 1 you can post there. Check this forum regularly.

Prof Regina Berretta
Mar -2023