School of Computing Sciences

Module: CMP-4008Y Programming 1
Assignment: Coursework 1 - Music Information System

Set by:	Dr Gavin Cawley (g.cawley@uea.ac.uk)
Date set:	8th November 2024
Value:	35%
Date due:	6th December 2024 3pm
Returned by:	N/A
Submission:	Blackboard
Checked by:	Dr Jason Lines J.Lines@uea.ac.uk

Learning outcomes

The aim of this assignment is for the student to gain experience in the design and implementation of a relatively small program using an object oriented approach. This will further reinforce the student's grasp of fundamental concepts such as classes, objects, instance variables and methods, but will also introduce the use of multiple classes, each of which defines a different type of object. During the assignment, the student will also begin to gain object-oriented design skills by deciding how the state and behaviours of the objects should be implemented and how the objects should interact.

On successful completion of this exercise, the student will have reinforced a basic understanding of the concepts of classes and objects, will be familiar with the basic syntax of Java programming constructs used to implement classes, instance variables and methods and will be able to select the instance variables and methods required to represent the state and behaviours of simple types of real-world objects. The student will have gained familiarity with some of the classes in the Java standard libraries used for file based I/O and collections. The student will also have an improved ability to use the ideas of cohesion and coupling to guide the design of programs comprised of more than a single class definition.

Specification

Overview

You are required to implement a Java program to store and manipulate the details of a collection of music albums and provide answers to a small set of simple database queries. Questions regarding the specifications will only be answered if posed via the discussion forum for this coursework. You are strongly advised to subscribe to this discussion forum so that you do not miss any modifications or clarifications of the specification.

Description

The file `albums.txt`, available on BlackBoard, contains information describing a number of music albums, including the recording artist, album title, and the duration and title of each track comprising the album. The program must load the information in this file into a suitable collection of objects, so that database queries could be answered. The program will consist of the following classes:

1. `Duration` is a class used to store the duration of a `Track`, `Album` or `AlbumCollection`. The class should have three integer member variables, representing the number of hours, minutes and seconds. The class must implement the `Comparable` interface.

2. `Track` is a class used to store the details of an album track. It should have member variables storing the title of the track as a `String` and the `Duration` of the track.

3. `Album` is a class used to store the details of an album, with instance variables storing the name of the artist and title of the album as `String` objects, the release year of the album as an `int`, and also a collection of `Track` objects, representing the contents of the album.

4. Class `AlbumCollection` is a class used to store the details of a collection of music albums. It should have a member variable that is a collection of `Album` objects. It should have methods that can be used to answer general (not specific) queries regarding the content of the collection.

Each class should have a separate `.java` file for ease of compilation and provide an appropriate set of constructors and accessor methods as well as overriding the `toString` method. The classes will need to have a variety of other methods in order to implement the required functionality.

The program must also have a `main` function, in a class called `AlbumDatabase`, that performs the following operations:

1. Read in an AlbumCollection from the file `albums.txt` provided on BlackBoard.

2. Display the entire album collection, arranged in alphabetical order of the album artist. If more than one album exists for a given artist, they should be displayed in ascending order of the year of release (oldest Album first).

3. Display the total play time of all Kraftwerk albums in the collection.

4. Display the album with the shortest title.

5. Display the details of the longest track in the album collection.

The main function should give a clear top-down overview of what the program does, without being overly cluttered with details of *how* it is done (demonstrating the advantages of *abstraction* and *encapsulation*).

Hints:

- Tackle the problem one class at a time (I would start with `Duration`; have it fully implemented and tested before moving on to `Track` and then `Album`, and so on). Only once you have implemented and tested classes representing each type of object, should you try to implement the album information program. Note this is an example of “top-down design, bottom-up implementation” discussed in the lectures.

- For each class, it is a good idea to work out what methods the class should provide in order to provide the services required to implement the program, before starting to code.

- Make sure you have a good idea of what you have to do *before* you start writing code.
- It is important to test each class in isolation (don't write all the code for all four major classes and expect them to work first time). Each class should have a `main` method, used as a *test harness*.
- Implementation of `Album` and `AlbumCollection` require the use of arrays, or more advanced types of collection, which we will cover in later lectures, However `Duration` and `Track` can be implemented and tested based on the material already covered.

All questions regarding the specifications **must** be asked via the appropriate discussion board on BlackBoard. Note that `albums.txt` may change before the due date for the assignment, and the answers to the queries must be based on the final version of that file.

Relationship to formative assessment

This assignment builds on the skills gained from the laboratory exercises for cmp-4008Y that have already been completed, for which formative feedback can be obtained from the teaching assistants during your scheduled laboratory session.

Deliverables

Your solution **must** be submitted via blackboard. The submission **must** be a single `.pdf` file generated using PASS, using the PASS server at `http://pass.cmp.uea.ac.uk/` . The PASS program formats your source code, and compiles and runs your program, appending any compiler messages and the output of your program to the `.pdf` file. If there is a problem with the output of PASS, contact me (gcc@uea.ac.uk). Do not leave it until the last moment to generate the `.pdf` file, there is a limit to the amount of help I am able to give if there is little time left before the submission deadline.

PASS is the target environment for the assignment. If your program doesn't operate correctly using PASS, it doesn't work, even if gives the correct answers on your own computer:

- The PASS program is not able to provide input to your program via the keyboard, so programs with a menu system, or which expect user input of some kind are not compatible with PASS. Design your program to operate correctly without any user input from the keyboard.
- Do not use absolute path names for files as the PASS server is unable to access files on your machine. If your program is required to load data from a file, or save data to a file, the file is expected to be in the current working directory. If the program is required to load data from a file called `rhubarb.txt` then the appropriate path name would be just `"rhubarb.txt"` rather than `"C:some\long\chain\of\directories\rhubarb.txt"`.
- If you develop your solution on a computer other than the laboratory machines, make sure that you leave adequate time to test it properly with PASS, in case of any unforeseen portability issues.
- If your program works correctly under InteliJ on the laboratory machines, but does not operate correctly using PASS, then it is likely that the data file you are using has become corrupted in someway. PASS downloads a fresh version of the file to test your submission, so this is the most likely explanation

Resources

- `https://stackoverflow.com/` - An excellent site for finding information about specific issues relating to various programming languages, including Java. It is important however not to become too reliant on sites such as `StackOverflow`, which are great for details, but don't give the "big picture", so it is difficult to get a good understanding of programming in this way. Note that if you re-use or modify code found on-line, then you must provide a comment giving the URL and a brief explanation of what modifications have been made. Re-using code found on-line without proper attribution would constitute plagiarism.

- `https://docs.oracle.com/javase/tutorial/` - A set of tutorials, provided by Oracle, who now develop the Java programming language.

- `https://docs.oracle.com/en/java/javase/17/docs/api/index.html` - Java Application Programming Interface (API) documents. Very useful for finding out what methods a class from the Java standard libraries provide.

Plagiarism and Collusion

Plagiarism is essentially passing off the work of someone else as your own. Collusion means working on the assignment with somebody else (which need not be a fellow student). Stackoverflow and other similar forums (such as the module discussion board) are valuable resources for learning and completing *formative work*. However,for summative (assessed) work such as this assignment, you may not post questions relating to coursework solutions in such forums. Your submission must be your own work, so searching for resources to answer specific programming issues is acceptable but searching for implementations of programs solving similar specifications is not. So for example, searching for information on `std::quoted` in C++ or `ArrayList` in Java is fine, but if the coursework were to implement a program to solve a Sudoku puzzle, then looking at other peoples implementations of Sudoku solvers clearly would not be acceptable. Using (or taking inspiration from) code from other sources/written by others without acknowledgement is plagiarism, which is not allowed (General Regulation 18). If you have any questions regarding plagiarism and collusion, then please ask them via the discussion forum set up for this assignment (anonymous posting is allowed).

Marking Scheme

Marks will be awarded according to the proportion of specifications successfully implemented, programming style (indentation, good choice of identifiers, commenting etc.), and appropriate use of programming constructs. It is not sufficient that the program generates the correct output, professional programmers are required to produce maintainable code that is easy to understand, easy to debug when (rather than if) bug reports are received and easy to extend. The code needs to be modular, where each module (e.g. class) has a well-defined interface to the rest of the program. The function of modules should be made as generic as possible, so that it not only solves the problem specified today, but can easily be modified or extended to implement future requirements without undue cost. The code should also be reasonably efficient. DO NOT use more advanced programming constructs of the Java programming language that were not covered in the lectures as this tends to make it more difficult to demonstrate that the learning objectives have been met.

The marking scheme is as follows (out of 35 marks):

- Class `Duration` 8 marks.
- Class `Track` 3 Marks
- Class `Album` 5 marks
- Class `AlbumCollection` 8 Marks
- `AlbumDatabase.java` (3 Marks), with functionality:
  - Display collection in alphabetical order of artist and then by release year (2 marks).
  - Display total play time of Kraftwerk albums (2 marks).
  - Display the album with the shortest title (2 marks).
  - Display the longest track in album collection (2 marks).

Plagiarism, collusion, and contract cheating

The University takes academic integrity very seriously. You must not commit plagiarism, collusion, or contract cheating in your submitted work. Our Policy on Plagiarism, Collusion, and Contract Cheating explains:

- what is meant by the terms ‘plagiarism’, ‘collusion’, and ‘contract cheating’
- how to avoid plagiarism, collusion, and contract cheating
- using a proof reader
- what will happen if we suspect that you have breached the policy.

It is essential that you read this policy and you undertake (or refresh your memory of) our school’s training on this. You can find the policy and related guidance here:

`https://my.uea.ac.uk/departments/learning-and-teaching/students/academic-cycle/regulations-and-discipline/plagiarism-awareness`

The policy allows us to make some rules specific to this assessment. Note that:

> In this assessment, working with others is *not* permitted. All aspects of your submission, including but not limited to: research, design, development and writing, must be your own work according to your own understanding of topics. Please pay careful attention to the definitions of contract cheating, plagiarism and collusion in the policy and ask your module organiser if you are unsure about anything.