National College of Ireland

BSHCIFSC2, HCCOMP2, BSHCE2, HCCOMPE2 2024

Tuesday 12th March 2024 in class
Saturday 16th March 2024, 23.55pm

Emer Thornbury
Hamilton Niculescu
Arjun Vinayak Chikkankod

Data Structures & Algorithms

Continuous Assessment (CA) Type: in-class quiz and project

Weight: 50% of overall module.
This CA has 2 parts, in-class quiz [25%] and an individual project [75%].

Instructions: The in-class quiz will be on Tuesday 12th March in-class. Duration 25 minutes. This is a closed book moodle quiz and you will be required to sign in.
The project is an individual assignment. It will be released on Tuesday 12th March and the submission date is Saturday 16th March 23.55pm.

SUBMISSION DETAILS:
Submit a zipped NetBeans project on moodle,
Submit a word problem Solving Document in word or PDF format on moodle.

- The document is to have font size of 12 pt. Include student name, student ID, and course name at the top of the page.
- It should have a summary of your design outlining the ADTs, interfaces, classes and your method of managing the songs (max 500 words).
- Include the link to your public GitHub repo.
- Copy and paste your class diagram to the document.

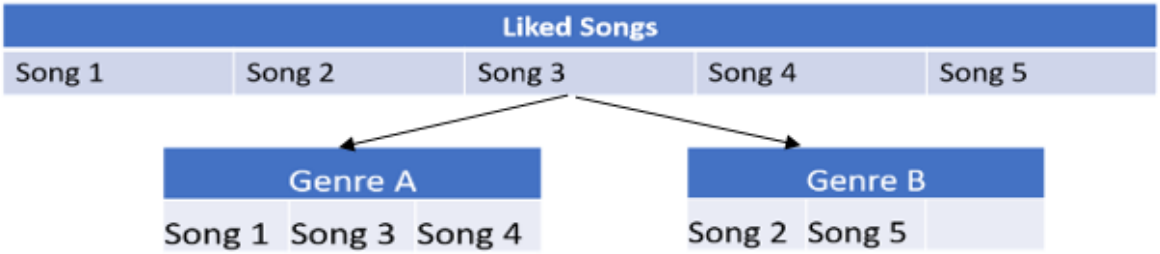
Late submissions will not be penalized if the student applied for an extension through NCI360 and it was approved.
Students may be asked to attend a viva examination to assess their understanding of the work submitted.

TURNITIN: All report submissions will be electronically screened for evidence of academic misconduct (i.e., plagiarism and collusion)

Design and implement an application that makes use of the abstract datatypes you have learnt in class with interfaces and a swing GUI using NetBeans.

Manage your music.
Design an application to allow a user to manage their music.

- They need to be able to create a playlist of liked songs. These songs will then be used to populate at least 2 playlists organised by criteria, such as genre.
- The user needs to be able to add a song from the liked playlist to one of the two genre playlists**.**
- The limitation of the system is that only the last song added to the liked playlist can be moved to the appropriate genre based playlist.
- Users can add, search, delete and move songs around on their playlists.
- The user needs to be able to see a printed list of whichever playlist they wish.
- Show how many songs are in each list.
- The user needs to be given the option to create a playlist that can be set to repeat.



You are required to commit your code to GitHub at least 5 times during the implementation process.

Submit a word doc/pdf, see submissions details above.

Grade Criterion	H1 (> 70%)	H2.1 (60-69%)	H2.2 (50-59%)	Pass (40-49%)	Fail (< 40%)
Problem solving document, GitHub & coding practices 25%	Summary clearly shows and demonstrates an accurate understanding of the problem through the solution. Solution shows objectives are met. Does not exceed the word count. GitHub has 5 or more commits. All classes, variables and objects correctly named. Excellent commenting clearly showing understanding.	Summary clearly describes the problem solution, shows a good understanding of the problem, solution shows objectives are met. Does not exceed the word count. GitHub has only 4 commits. All classes, variables and objects correctly named. Adequate comments present that shows understanding.	Summary adequately describes the problem solution and objectives are clear, and at mostly met. Exceeds the word count. GitHub has only 2 or 3 commits. Most classes, variables and objects correctly named. Adequate, or excessive, comments present, showing some understanding.	Summary describes the problem solution where objectives, which are at least partially met. Exceeds the word count significantly. GitHub has only 1 commit. Some classes, variables and objects are correctly named. Few comments present and do not show understanding.	Summary not present or does not address the topic. Cannot discern objectives and/or if objectives were met. GitHub link is not present or does not show the repo. Classes, variables and objects not correctly named. No comments present.
Demonstration of ADTs and interfaces 20%	Presence of 2 abstract datatypes learnt in class thoroughly and accurately applied to a very high standard. Presence of 2 interfaces.	Presence of 2 abstract datatypes learnt in class appropriately applied and detailed to a good standard. Presence of 2 interfaces.	Presence of 2 abstract datatypes learnt in class adequately applied and detailed. Presence of 2 interfaces.	Presence of only 1 abstract datatype learnt in class and only 1 interface. Work implemented to a basic standard.	Presence of only 1 abstract datatype learnt in class and only 1 interface. Work implemented to a poor standard.
Implementation 35%	Has gone above and beyond to implement all functionality to a high standard. Swing Gui with all objects present and excellently done.	All functionality implemented to very good standard. Swing Gui with all objects present and done to a very good standard.	Most functionality implemented to an adequate standard. Swing Gui with most objects present and done to an adequate standard.	Basic functionality implemented showing confusion. Swing Gui with most objects present.	Some functionality implemented with confusion. Poor swing Gui with most objects missing.
Testing, error free and running 20%	Application runs from the main class. Main class has testing of all functionalities clearly demonstrated. Project is free of logical, syntax and runtime errors.	Application runs from the main class. Main class has testing of all functionalities demonstrated. Project is free of logical and syntax errors.	Application runs from the main class. Main class has testing of most functionality demonstrated. Project is free of syntax errors.	Application does not run from the main class. Main class has limited testing of functionality demonstrated. Project has many errors, not all running.	Application does not run from the main class. Main class has minimal or no evidence of testing. Project has many errors and does not run.