**University of Westminster**
School of Computer Science & Engineering

# 4COSC010C Programming Principles 02

| | |
|---|---|
| Module leader | Guhanathan P |
| Unit | COURSEWORK 2 |
| Weighting: | 50% |
| Qualifying mark | 30% |
| Description | This coursework requires students to apply Object Oriented Programming and Design principles to a given set of requirements from case study. |
| Learning Outcomes Covered in this Assignment: | This assignment contributes towards the following Learning Outcomes (LOs):<br>- LO1<br>- LO2<br>- LO3<br>- LO4 |
| | |
| Handed Out: | 4th of July 2019 |
| Due Date | 5th of Aug 2019 |
| Expected deliverables | Submit on Blackboard a zip file containing:<br>**A folder with all the UML documents attached**<br>**A folder with the developed project (IntelliJ Solution with your Java code) A document in Word or pdf with the code copied and pasted**<br>**Demonstration of your work** |
| Method of Submission: | Electronic submission on BB via a provided link close to the submission time. |
| Type of Feedback and Due Date: | Written feedback by 21/08/2019 and generic feedback during the demonstration. |
| BCS CRITERIA MEETING IN THIS ASSIGNMENT | **2.1.1 Knowledge and understanding of facts, concepts, principles & theories**<br>**2.1.2 Use of such knowledge in modelling and design**<br>**2.1.3 Problem solving strategies**<br>**2.2.1 Specify, design or construct computer-based systems**<br>**2.2.4 Deploy tools effectively**<br>**2.3.2 Development of general transferable skills**<br>**3.1.1 Deploy systems to meet business goals**<br>**4.1.1 Knowledge and understanding of scientific and engineering principles**<br>**4.1.3 Knowledge and understanding of computational modelling** |

**Assessment regulations**

Refer to section 4 of the “How you study” guide for undergraduate students for a clarification of how you are
assessed, penalties and late submissions, what constitutes plagiarism
etc.

**Penalty for Late Submission**

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

# Coursework Description

**Objective**:
The aim of this assessment is to assess the skills and learning that you have acquired about object-oriented programming during the module. You are asked to implement a program in which objects interact in order to fulfill a set of functional requirements.

**Analyse the statement**:
An important skill that you expect to develop in this module is the ability to analyze a problem statement in order to identify the requirements to develop a solution.

In this assignment, the first task you should perform is a careful analysis of the problem statement in order to make sure you have all the information to elaborate a solution. Do not make assumptions about what is needed! If you are not sure, about the information provided, please send an email to the module leader.

**Design a solution**:
The design of your system should be consistent with the Object Oriented principles and easy to understand by an independent programmer.

You are required to identify the requirements **(5 marks)** and draw the following:

- An use case diagram for the system ***(5 marks)*.**
- Use case description for the use cases identified in the above diagram **(5 marks)**

You are also required to design your program using UML diagrams. In particular you have to draw:

- A class diagram with necessary relationships, roles and multiplicities ***(5 marks)***
- An activity diagram for each of the identified functionality in the use case **(10 marks)**

**Problem description and requirement statement**
You are required to develop a program that implement a basic online music store management system. In the system you will be able to store only 1000 items in total.

In this assignment, you will be required to implement the following functionality*:*

1. Design and implement a class **MusicItem** (abstract) to hold information about the *itemID*, the *Title*, the *Genre*, the *Release Date*, the *Artist* and the *Price*. ***(2 marks)***
2. Also, design and implement the following subclasses of MusicItem:

   - The **CD** class should include a specific method to hold information about the total *duration* of the songs ***(2 marks)***.

   - The **Vinyl** class should include methods and information about the *speed* and the *diameter* ***(1 marks)***.

   - You should implement a class **Date** to represent the date of when the item has been released. Do not use any predefined library for date and time and you can refer as example to the class that has been provided during the tutorials ***(2 marks)***.

3. Design and implement a class called **WestminsterMusicStoreManager**, which implements the interface **StoreManager** ***(1 marks)***.

   WestminsterStoreManager maintains the list of the items ***(2 marks)*** in the store and provides all the methods for the music store manager.

The class should display **in the console a menu** containing the following management actions from which the manager can select one.

- **Add a new item** in the store and display the number of spaces left (remember that the system can store max 1000 items). Display a message in case there are no spaces available. The user can select if they would like to add a CD or a vinyl and enter the corresponding information ***(2 marks)***.

- **Delete an item**, given the item ID, from the store and display the number of free spaces left. Display the type of the item that has been deleted (if it is a CD or vinyl) **(*2 marks*).**

- **Print the list of the items** in the store. For each item print the ID, the type of item (if it is a CD or vinyl ) and the title ***(2 marks).***

- **Sort the item** in ascending order according to the title ***(4 marks)***.

- **Buy an item**: the user can buy an item selecting the item ID. The user can also buy more than one copy (the system will ask how many items to buy) and he/she will be able to see on the screen the total cost. ***(3 marks)***.

- **Generate a report:** write in a file all the items that have been sold. Every time the user buys an item (using the previous command) write in the file the title, the ID, the price and the selling time/date ***(5 marks)***.

**Create a Graphical User Interface** (GUI) using JavaFX from where it is possible to see all the information for each item. The GUI can be opened selecting an option from the menu console *(1 mark).*

Regarding the GUI, note that all code must be manually written, **do not use any tools that generates code automatically**. You are NOT allowed to use design tools with drag and drop functionality (such as those found in Netbeans) to create the graphical user interface for any part of this coursework!

a. You are asked to show the list of items in the store with the main information. You MUST use a table component from JavaFX to visualise them *(**5 marks**)*.

b. The user can search a specific item by title ***(4 marks)***.

3. Some challenges: **(5 marks)**

- According to the items that have been sold you can propose an algorithm that can be use to recommend to the store manger how many copies of each item should be added in the store every month in order to avoid the unavailability of the item.
  Add in your console menu this option. You can print on the screen the information for each item.
  *Write in comments how you ended up with your solution!*

4. Derive test cases using black box (**5 marks**) and white box (**10 marks**) for each of the identified functionalities and show evidence that you have used both black box and white box.

**Demonstration**

You will be asked to demonstrate the system *(**5 marks**)*. The following will be evaluated:

- your ability in articulating and explaining the code
- your ability in defending your implementation choices

Note: no marks will be given to a task if there is no understanding of the code!

Also, the following will be marked:

- quality of the code (e.g. good use of comments, naming of variables, structure of the code) *(**5 marks**)*.
- error handling (in particular: if the user insert a wrong parameter, is the code robust to handle it?, are the IDs unique? Etc.) ***(3 marks)***