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 fulfil a set of functional requirements.

Analyse the statement:
An important skill that you expect to develop in this module is the ability to analyse 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. If you have any question please write your queries on the Padlet on BB.

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 design your program using UML diagrams. In particular you have to draw:

- a class diagram *(5 marks)*
- two or more use cases for the system *(5 marks)*.

Problem description and requirement statement
You are required to develop a program that implement a basic online gym management system.
You should implement a **console system** from where the manager can add new members, delete if needed, print the members information, etc. (as described in detailed below).
You should implement a **Graphical User Interface** (GUI) from where a manager can see the list of members and search them.

In the system you will be able to record details of 100 members maximum.

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

1. According to the *Inheritance* principle you have to design and implement a super class **DefaultMember** *(5 marks)* and the subclasses **StudentMember** and **Over60Member**. The classes should include appropriate methods in order to comply with the *encapsulation and inheritance principles* and hold information about the *MembershipNumber,* the *Name* and the *StartMembershipDate.*

   Furthermore:

   - The **StudentMember** class should include specific information about the *SchoolName* and the relative get/set methods *(4 marks)*.

   - The **Over60Member** class should include specific information about the *Age* and the relative get/set methods *(4 marks)*.

   - You should implement a class **Date** to represent the starting membership date (which is the instance variable in the DefaultMember class). 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 *(3 marks)*.

2. Design and implement a class called **MyGymManager**, which implements the interface **GymManager** *(2 marks).*
   MyGymManager maintains the list of the members *(2 marks)* and provides all the methods for the gym 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 member** in the gym and display how many members could still be registered in the system (remember that the system can store max 100 members). Display a message in case there are no spaces available. The manager can select if adding a default member or a student member or an over60 member and enter the corresponding information. *(5 marks)*.

- **Delete a member**, given the membership number, and display the number of free spaces left in the system. Display the type of the member that has been deleted (if it is a default, student or over60 member) *(5 marks)*.

- **Print the list of members** in the system. For each member, print the membership number, the type of member (if is a default, student or over60 member), the member starting date and the member name. *(4 marks)*.

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

- **Write/Save in a file** the list of members with all the relative information (5 marks).

- **Open a Graphical User Interface** (GUI) from the menu console.
  You should implement the GUI according the following specification:

- The manager can visualise the list of members through a table with relative information.*(8 marks)*.

- The user can search the members according at least one parameter (e.g. membership number, or the name, etc.) *(4 marks).*

Note: You can choose how the GUI should look like and how to meet at the best these specifications.

3. Testing and system validation:

- Write a test plan designed to ensure that the coded solution works as expected. The test plan will include specific instructions about the data and conditions the program will be tested with *(5marks).*
- Implement an automated testing (you can use JUnit or any other tool or scripts for unit testing) that runs scenarios of each of the use cases you implemented in the console menu *(10 marks)*.
- The following will be evaluated:
  - The robustness of the code through the use of error handling and input validation *(5marks).*
  - The quality of the code and the adherence to coding standards and conventions *(5 marks).*

Video Recording and Demonstration

- You will be asked to demonstrate all the functionalities implemented in the system and **record it in a video** that you should upload in YouTube and paste the link in the submission notes *(10 marks)*. Your ability in articulating and explaining the code will be evaluated. **Please note that if you fail to provide a video recording of your system the maximum mark shall range from 130% only.**

Please note that a demonstration through a collaborative tool could be arranged to assess further understating of the code.