DESIGN AND ANALYSIS OF DATA STRUCTURES AND ALGORITHMS

# COURSEWORK B

Student Name : Timo Andrei – Claudiu

Student ID : 19000915

Educational Institution : University of the West of England

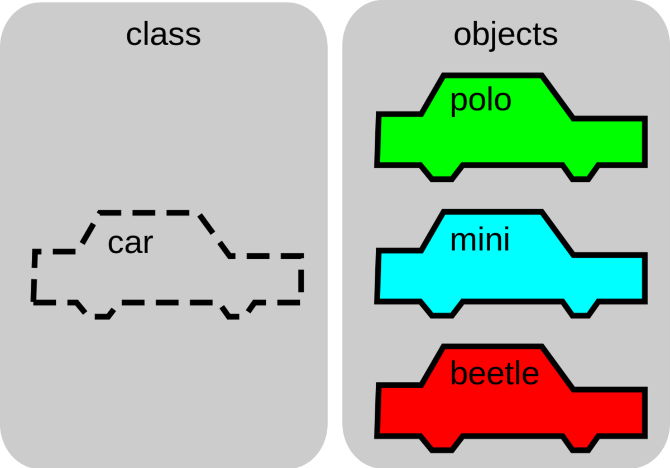
Module Leaders: Dr. Gordon Downie & Dr. Elias Pimenidis

Module Tutors : Dr. Kun Wei & Mr. James Lear

Year: 2020 – 2021

## Justification of the design choices

* 1. Introduction

 In object-oriented programming, a ***class*** is a blueprint for creating ***objects*** (a particular data structure), providing initial values for state (member variables or attributes), and implementations of behaviour (member functions or methods). The user-defined objects are created using “**class**” keyword. The class is a blueprint that defines a nature of a future object. An **instance** is a specific object created from a particular class. Classes are used to create and manage new objects and support **INHERITANCE** – a key ingredient in object-oriented programming and a mechanism of reusing code.

The image above shows how a ***Car*** object can be the template for many other ***Car*** instances. In the image, there are three instances: polo, mini, and beetle. A class can define types of operations, or methods, that can be performed on a ***Car*** object. For example, the Car class might specify an accelerate method, which would update the speed attribute of the car object.

* 1. Explain the designed choice

For this particular coursework, I decided to create and use a class named “Patient” that will store the information needed to meet the requirements - Name, Date of birth, Conditions of a patient, Height, Weight, and Build Mass for each patient. For a better manipulation of the data extracted from the .csv file, I used an array of objects through, in which I stored the information contained by the attributes of the Patient class – Patient\_List[].

By using the Patient\_List array, I made my work easier by storing several types of data, such as integers, real numbers, strings and characters. It enabled me to keep data together that belongs together, to condense my code, and perform the same methods or operations on multiple values at once. Having a look on the .csv file which stores information about the patients name, their birth date, but also conditions of each patient, creating and using an array of objects helped me to manipulate the data more easily and to perform several operations at the same time – reading data and calculating the BMI and age for each patient, by using the height, weight and date of birth attributes.

* 1. Conclusion

In conclusion, classes are a strong point for this coursework for the following reasons: organization: OOP defines well known and standard ways of describing and defining both data and procedure in code. Both data and procedure can be stored at varying levels of definition, helping the programmer and others understand, edit, and reuse the code; state: OOP helps the programmer to define and keep track of state. For instance, in a classic example, if you're creating a program that processes students (for instance, a grade program), you can keep all the info you need about them in one spot (name, age, gender, grade level, courses, grades, teachers, peers, diet, special needs, etc.), and this data is persisted as long as the object is alive, and is easily accessible; [encapsulation](https://en.wikipedia.org/wiki/Encapsulation_(computer_programming))(with encapsulation, procedure and data are stored together. Methods <an OOP term for functions> are defined right alongside the data that they operate on and produce).