# ©Homework: Defining Classes

This document defines the homework assignments from the ["OOP" Course @ Software University](https://softuni.bg/trainings/coursesinstances/details/8). Please submit as homework a single zip / rar / 7z archive holding the solutions (source code) of all below described problems. The solutions should be written in C#.

## Laptop Shop

Define a class **Laptop** that holds the following information about a laptop device: **model**, **manufacturer**, **processor**, **graphics card**, **battery**, **battery life** in hours, **screen size** and **price**.

* Define 2 separate classes (class **Laptop** holding an instance of class **Battery**).
* Define several constructors that take different sets of arguments (full laptop/battery information or only part of it). Use proper variable types.
* All non-numeric data should be mandatory. All numeric fields should have a default value of 0.
* Add a method in the Laptop class that displays information about the given instance. (Tip: override **ToString()**);
* Use properties to validate the data. String values cannot be empty, and numeric data cannot be negative. Throw exceptions when improper data is entered.

## PC Catalogue

Define a class **Computer** that holds **name**, **several components** and **price**. The components (processor, graphics card, motherboard, etc.) should be objects of class **Component**, which holds **name**, **details** (optional) and **price**.

* Define several constructors that take different sets of arguments. Use proper variable types. Use properties to validate the data. Throw exceptions when improper data is entered.
* Add a method in the Computer class that displays the **name**, each of **the components' name** and **price** and the **total computer price**. The total price is the **sum of all components' price**. Print the prices in BGN currency format.
* Create several Computer objects, **sort them by price**, and print them on the console using the created display method.

## \*\* Software University Learning System

Define a class **Person** and the classes **Trainer**, **Student**. There are two types of trainers – **Junior** and **Senior Trainer**. There are three types of Students – **Graduate**, **Current** and **Dropout Student**. There are two types of Current Students – **Online** and **Onsite Student**. Implement the given structure below. **A class down the hierarchy should implement the fields, properties and methods of the classes above it.** (Tip: Use **Inheritance** to achieve code reusability). The classes should implement the following fields/methods:

* Person – fields **first name**, **last name**, **age**
  + Trainer – method **CreateCourse([courseName])** that prints that the course has been created
    - Senior Trainer – method **DeleteCourse([courseName])** that prints that the course has been deleted
  + Student – fields **student number**, **average grade**
    - Current Student – field
      * Onsite Student – field **number of** **visits**
    - Dropout Student – field **dropout reason**, method **Reapply()** that prints all information about the student and the dropout reason

Write a class **SULSTest** that tests the implemented class structure. Create a **list of objects from each class**. Extract only the **Current Students**, **sort them by average grade** and **print information** about each one on the console. (Tip: Use the LINQ extension methods **.Where()** and **.OrderBy()** with lambda expressions.)

