Homework: Defining Classes

This document defines the homework assignments from the ["OOP" Course @ Software University](). 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#.

Problem 1. Persons

Define a class **Person** that has **name**, **age** and **email**. The **name** and **age** are mandatory. The **email** is optional. Define **properties** that accept non-empty name and age in the range [1...100]. In case of invalid argument, throw an exception. Define a property for the email that accepts either **null** or non-empty string containing '**@**'. Define two **constructors**. The first constructor should take name, age and email. The second constructor should take name and age only and call the first constructor. Implement the **ToString()** method to enable printing persons at the console.

Problem 2. Laptop Shop

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

- The **model** and **price** are mandatory. All other values are optional.
- Define two separate classes: a 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.
- Add a method in the **Laptop** class that displays information about the given instance
    - Tip: override the **ToString()** method
- Put **validation** in all property setters and constructors. String values cannot be empty, and numeric data cannot be negative. Throw exceptions when improper data is entered.

Sample laptop description (full):	
model	Lenovo Yoga 2 Pro
manufacturer	Lenovo
processor	Intel Core i5-4210U (2-core, 1.70 - 2.70 GHz, 3MB cache)
RAM	8 GB
graphics card	Intel HD Graphics 4400
HDD	128GB SSD
screen	13.3" (33.78 cm) – 3200 x 1800 (QHD+), IPS sensor display
battery	Li-Ion, 4-cells, 2550 mAh
battery life	4.5 hours
price	2259.00 lv.

Sample laptop description (mandatory properties only)	
model	HP 250 G2
price	699.00 lv.

Problem 3. PC Catalog

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.

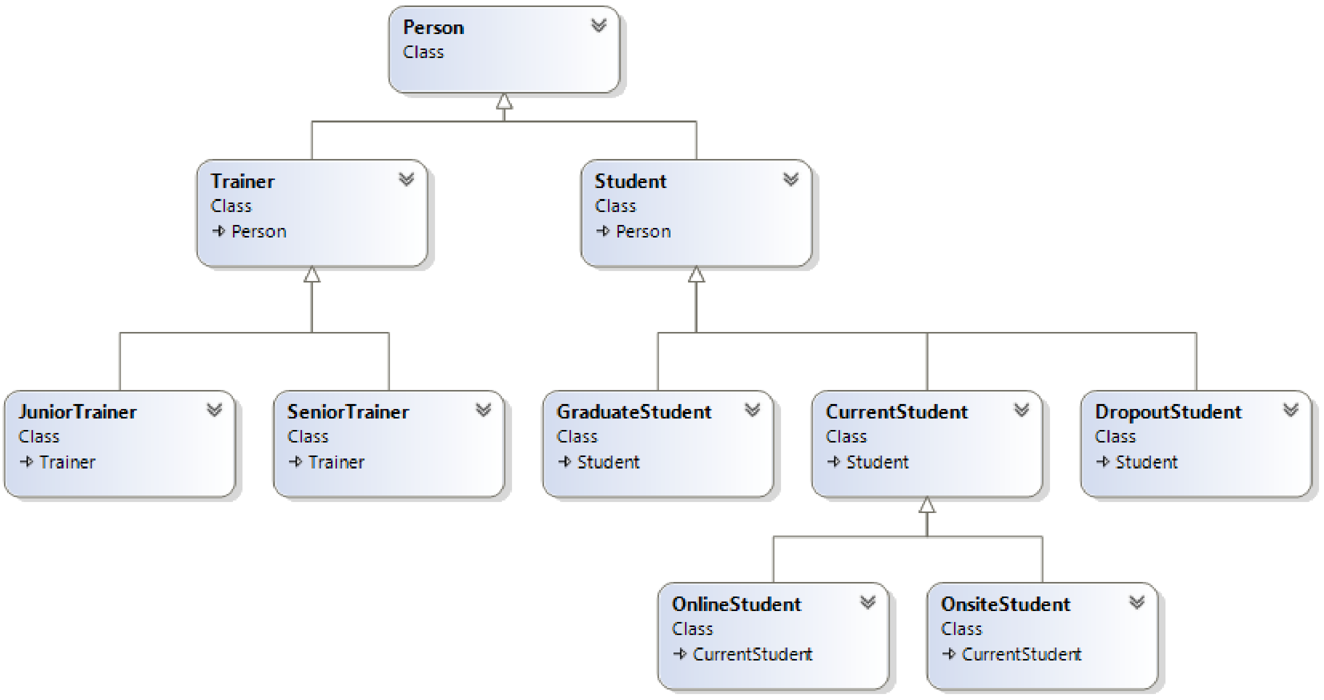
Problem 4. ** Software University Learning System

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 in 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 **current course**
      - **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.


Person
Class
Trainer
Class
Person
Student
Class
Person
JuniorTrainer
Class
Trainer
SeniorTrainer
Class
Trainer
GraduateStudent
Class
Student
CurrentStudent
Class
Student
DropoutStudent
Class
Student
OnlineStudent
Class
CurrentStudent
OnsiteStudent
Class
CurrentStudent