Object-Oriented Programming

Spring Semester 2022

Homework Assignment 1.

03/03/2022

Contents

Chapter 1

Introduction

1. I strongly recommend you work on this homework assignment as soon as possible, due to reasons that are related to familiarity with the programming environment:
   - Make sure your IDE (recommended: Visual Studio or Code Blocks) is running on your computer.
   - Make sure you can compile C/C++ codes.
2. Try to provide clear and careful solutions.
3. You should provide comments for your code, so it will be completely clear what you are trying to achieve.

1.1 Definitions

In your first homework, you will be introduced to the most common concepts of working with C++ and OOP paradigm. You are going to implement part of a system that helps to manage a Computer Store. Part of this system design includes the following basic models: **CPU, Computer** and **Store**. In the next sections, you will find specifications for these classes.

Your goals are:

- Given class definitions, write C++ classes such that they conform to the requirements.
- Given class definitions, distinguish private properties from public properties.
- Having method descriptions, you should provide implementation while keeping in mind basic OOP concept of *encapsulation*.
- **Keep in mind that the included main files are very basic and do not cover all possible scenarios and end cases.**
- Separate implementation from declaration code using the convention: implementation in "*.cpp" files, and declaration in "*.h files".
- Please note the following few points which may lead to points reduction during the submission check:
    - Avoid code duplication as much as possible.
    - You should not **globally** enable *std* namespace usage or any other namespaces, e.g. *using namespace std;*.

1.1.1 Cpu

The class “Cpu” has the following attributes:

- Clock rate (GHz) – can be a decimal number.
- Manufacturer name – string with a maximum of 10 characters.
- Year of manufacturing – int.

And the following methods:

- Construction – should initialize all the fields in the class – by default: number values to 0, strings to “~”.
- Getters – for each field in the class you should provide a “get” method, that returns the demanded attribute.
- Setters – for each field in the class you should provide a “set” method, that sets a given value in the required attribute. (If input to long then output the massage "Manufacturer's name length is too long" and set the default value)
- **`void print() const`** – a method that prints the cpu details, each attribute in new line in the following form:

  <attribute1_name>: <attribute1_value>

  <attribute2_name>: <attribute2_value>

  ..

1.1.2 Computer

A class called “Computer” has the following attributes:

- Cpu
- Manufacturer name – string with a maximum of 10 characters.
- Year of manufacture – int.
- Color – string with a maximum of 10 characters.
- Is it a Laptop - boolean

And the following methods:

- Construction – should initialize all the fields in the class – by default: number values to 0, strings to “~”.
- Getters – for each field in the class you should provide a “get” method, that returns the demanded attribute.
- Setters – for each field in the class you should provide a “set” method, that sets a given value in the required attribute. (If input to long then output the massage "Manufacturer's name length is too long" or "Color string length is too long" and set the default values)
- **`void print() const`** – a method that prints the computer details, each attribute in new line in the following form:

  <attribute1_name>: <attribute1_value>

  <attribute2_name>: <attribute2_value>

..

1.1.3 Store

A class called "Store" has the following attributes:

- **Name** – the name of the store, string with a maximum of 10 characters.
- **Computers –** a **finite** array that contains limited number of computers.

  (you can use `#define MAX_COMPUTERS_NUMBER 10` for example).
- **Computers number** – current number of computers in the store.

And the following methods:

- **Construction** – gets as a parameter the name of the store, and should initialize all the fields in the class – by default: number values to 0, strings to "~", booleans to false.
- **Getters –** for each field in the class you should provide a "get" method, that returns the demanded attribute.
- **Setters** –you should provide a "set" method, that sets a given value in the attribute **Name**.

  (If input to long then output the massage "store name length is too long" and set the default value)
- **`void addComputer(const Computer& computer)` -** add a computer to store computer list. If the store is full, the method should print "The store is full!". (no need to support removal of computers from the store).
- **`void printComputersByName()`** – a method that prints the details of the computers in the store in an increasing order of their lexicographic manufacture's name.
- **`void printComputersByYear()` –** a method that prints the details of the computers in the store in an increasing order of their year of manufacturing.

  *Both prints should print all the computers in the store sorted by the selected attribute in the following form:

  There are < computersNumbers> computers in the store. The computers:

  Computers 1:

  Cpu clock rate : < *computer1_cpu_attribute1_value*>

  Cpu manufacturer: < *computer1_cpu_attribute2_value*>

  Cpu year : < *computer1_cpu_attribute3_value*>

  Computer manufacturer: <*computer1_attribute1_value*>

  Computer year : <*computer1_attribute2_value*>

  Computer color : <*computer1_attribute3_value*>

  Computer is laptop : <*computer1_attribute4_value*>

  Computer2:

  Cpu clock rate : < *computer2_cpu_attribute1_value*>

Cpu manufacturer: < *computer2_cpu_attribute2_value*>

…

Note#1: Use the exact signature for the methods detailed above.

Note#2: (For this exercise) Booleans should be printed as "True" if 1, and "False" if 0.

1.2 Evaluation

Homework exercise provided with the following example program file and corresponding output:

1. main.cpp
2. output.txt

You should be able to compile your code with “main.cpp” file and receive correct output, which placed in “output.txt” file.

*Please note that line 15 in the main.cpp file is commented out.

The function “freopen("output.txt", "w", stdout)” saves the program output to a text file named “output.txt” in the solution directory, so you can uncomment this line in order to compare your output to mine. (I recommend comparing the outputs via DiffMerge , Link: https://sourcegear.com/diffmerge/)

In order to use the “freopen” function please follow the following video tutorial:

How to handle Error C4996 in Visual Studio

Link: https://www.youtube.com/watch?app=desktop&v=qWxGZLjwKL0

Make sure you keep the C++ syntax convention as described in “Oop – Cpp – Conventions and Requirements” file in the Moodle.

GOOD LUCK! ☺