Упражнения: Качествени класове

Problem 1. Голям проект

Разгледайте голям C# проект. Можете да погледнете в GitHub – потърсете проекти на езика C#, сортирайте ги по брой разклонения (forks, в низходящ ред) и изберете един от първите няколко проекта. Проектът, който сте избрали да прегледате, трябва да съдържа поне 30 класа. Попълнете таблицата по-долу, като документирате това, което намерите.

Документирайте всичко, което харесате (или не харесате) в кода.

Клас / Метод	Линк	Ноти
System.IO.File	http://referencesource.microsoft.com/#mscorlib/system/io/file.cs	Класовете са разделили своите задължения, нивата на абстракцията са постоянни…
		Интерфейсите са малки, всеки интерфейс има собствени отговорности
…	…	…

Някои неща, които да потърсите:

- ООР принципи
- Потвърждения и справяне с изключения
- Правилни нива на абстракция

Допълнително можете да потърсите и неща, които вече сте научили:

- Форматиране на кода
- Ясно именуване – променливи, методи, пространства от имена и т.н.
- Документация и коментари
- Използване на променливи
- Вмъкване на цикли и условни команди
- Праволинеен код
- Силна специализация и слаба зависимост

Problem 2. * Шаблони в дизайна

Опитайте да намерите шаблони в дизайна на кода, който вече разгледахте в задача 1. Документирайте това, което откриете, в таблицата по-долу:

Шаблон в дизайна	Пример	Описание
Singleton	https://github.com/apache/log4net/blob/trunk/src/log4net/LogManager.cs	Класът **LogManager** използва Singleton защото трябва да има само една инстанция на log-мениджъра за дадено приложение
…	…	…

Problem 3. Абстракция

Използвайте VS решението "**Abstraction**".

- Преправете кода му, за да има добра абстракция.
- Преместете свойствата и методите от класа Figure на правилните им места.
- Преместете общите методи в тялото на базовия клас.
- Махнете всичкия повтарящ се код (свойства / методи / друг код).

Problem 4. Капсулиране

Използвайте VS решението "**Abstraction**".

- Осигурете добро капсулиране във всички класове.
- Уверете се, че във вътрешното състояние на класовете не могат да бъдат зададени неверни стойности.

Problem 5. Специализация и зависимост

Използвайте VS решението "**Cohesion-and-Coupling**".

- Преправете кода му така, че да следва принципите на добра абстракция, слаба зависимост и силна специализация.
- Разделете класа Utils на други класове, които имат силна специализация и са слабо зависими вътрешно.

Problem 6. Наследяване и полиморфизъм

Използвайте VS решението "**Inheritance-and-Polymorphism**"

- Следвайте най-добрите практики за висококачествен програмен код.
- Проектирайте наново класовете и пренапишете кода.
- Извлечете абстрактен базов клас и преместете в него всички общи свойства.
- Капсулирайте полетата и се уверете, че задължителните полета не са оставени без стойност.
- Използвайте отново (reuse) дублирания код чрез методи в базовия клас.

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).