Упражнения: Именуване на идентификатори в кода

Problem 1. Да пишеш нов код – по добрия и по лошия начин

Влезте в Judge системата. Изберете един от миналите изпити. Изберете изпит и решете една задача от него. По-добре е да изберете задача, която: 1) не сте решавали преди, и 2) се решава с поне 30 реда код.

Отначало използвайте лошо именуване **нарочно** – имена на променливи, класове, методи и т.н., например **`class Program`**, **`int a`**, **`string[] s`**, и т.н.. След като judge системата ви даде **поне половината точки** за задачата, променете всички имена.

Ще забележите, че избирането на добри и на лоши имена отнема еднакво време. Това, което наистина ви губи времето, са опитите да разбрете каква е била първоначалната ви идея, когато сте писали кода.

Освен това, това беше прост проект, по който работите сам. Какво би се случило, ако цяла софтуерна система беше написана с лоши имена? Ами ако бяхте част от голям международен екип, в който всеки член пишеше код на собствения си език?

Problem 2. Преправяне на собствен код

Намерете някой от по-ранните си проекти, по възможност такъв, в който именуването не ви е интересувало много. Преправете го така, че да следва всички изисквания, конвенции, препоръки и най-добри практики за именуване.

Ако кодът ви е твърде добър, за да намерите в него нещо подобно ☺, използвайте клас, намерен онлайн.

Problem 3. * Оценка на код

Работете по двойки. Използвайте кода, който сте преправили. Покажете първоначалния и преправения код на партньора си. Поискайте им оценка. Може да е писмена или устна.

Оценка на кода е коментар върху вашия код. Трябва да включва неща като: колко се е променило качеството на кода след като сте го преправили, кодът по-лесен ли е за четене и разбиране, има ли друго, което бихте могли да преправите.

Разменете си ролите. Оценете кода на партньора си.

Problem 4. Именуване на програмния код на .NET Framework

Погледнете .NET Framework Reference Source, ще го намерите тук: http://referencesource.microsoft.com/ (може би ще искате да разгледате асемблито **`mscorlib`** по-отблизо). Разгледайте имената на идентификаторите (файлове, асемблита, пространства от имена, класове, структури, свойства, методи, местни променливи, параметрии т.н.).

Ще забележите, че кодът е добре написан и лесен за четене, разбиране и поддръжка.

По желание: Никой не е перфектен и Microsoft също прави грешки. Ако попаднете на пример(и) за лошо именуване, документирайте ги. Може да използвате таблицата по-долу за ръководство:

Тип	Име	Поправено име	Причина

System.MarshalByRefObject.InvokeMember (http://referencesource.microsoft.com/#mscorlib/system/marshalbyrefobject.cs,83)	Type **t** = GetType();	resultType	It is better to understand that "t" is actually a type and it comes as a result of the GetType() operation
…	…	…	…

Нещо, което смятате за лош код, може всъщност да е част от конвенцията за кода (например долна черта преди името на частно поле). Въпреки това документирайте всичко, което смятате, че нарушава принципите на висококачествения код.

Problem 5. Преправяне на чужд код

Намерете чужд код с лошо именуване. Може да търсите в open source хранилища като **GitHub** (http://github.com) или **CodePlex** (http://codeplex.com). Ако искате специфичен пример за зле написан код, погледнете в някой от следните сайтове: **GovnoKod** (http://govnokod.ru) **Bad Programming** (http://badprogramming.com), или **BadCode в reddit** (https://www.reddit.com/r/badcode).

Problem 6. Доброто име сложни програми поправя

Прегледайте програмния файл **ArraySlider.docx** и **ArraySlider.cs**. Това е истинска изпитна задача и примерно решение, което е абсолютно вярно и изкарва 100/100 точки в системата judge. Въпреки верността си, кодът е зле написан и труден за разбиране и поддръжка. Например, всички променливи са наименувани с една буква и, въпреки че кодът е сравнително къс, хората лесно се губят в него.

Вашата задача: Дайте смислени имена на променливите

Общи съвети:

- Използвайте английски! Дори и да е само парче код, която вие и само вие ще четете, просто използвайте английски. **Без извинения**.
- Избягвайте съкращения: **dateTimeFormatter** е по-добре от **dtf** или **dtFormatter**
- Престараването също е лоша идея.

Пример*:*

Като гледаме кода, забелязваме реда **var r = Console.ReadLine();** Очевидно е някакъв вид вход, но не можем да разберем какъв точно, какво прави и защо е там. Описанието на задачата ни казва, че след първия ред код ще получим допълнителни команди, всяка на нов ред. Можем да заменим реда с:
string command = Console.ReadLine();
string[] commandTokens = Console.ReadLine().Split(); също е допустимо, в зависимост от това как решите да структурирате кода си.

Бонус задача: Някои части от кода са дублирани или ненужни. Намерете начин да направите кода по-сбит и по-елегантен.

Problem 7. Непознат метод

Даден ви е проектът **ConsoleApplication1**, който е верен и няма бъгове, но кодът е написан много зле. Преименувайте всички идентификатори, които трябва, за да направите кода по-четим и по-лесен за разбиране и многократно използване.

Може да се наложи да дебъгнете и да разгледате програмата по-отблизо.

Problem 8. LINQ заявки

Дадено ви е приложението **Orders**, което обработва поръчки. Приложението има **продукти**, **категории** и **поръчки**. Всеки артикул има уникално **ID**. Елементите (продукти, категории и поръчки) са свързани чрез своите ID-та. Например, ако има следните категории:

```
ID: 1, Name: Beverages
ID: 2, Name: Condiments
```

и следните продукти:

```
ID: 1, Name: Chai, CategoryID: 1
ID: 4, Name: Chef Anton's Cajun Seasoning, CategoryID: 2,
```

това значи, че продуктът **Chai** се съдържа в категорията **Beverages**.

Дадена ви е програма, която чете отделени със запетая текстови файлове и зарежда информацията в системата. След това програмата изпълнява четири LINQ заявки, както следва:

- Намира имената на петте най-скъпи продукта
- Намира броя продукти във всяка категория
- Намира петте топ продукта (с максимален брой поръчки)
- Намира името на най-доходната категория (тази, която генерира най-голям общ доход)

Програмата работи вярно. Задачата ви е да промените имената на всички идентификатори, които трябва да бъдат преименувани, за да направите кода по-четлив, по-ясен за разбиране и по-лесен за промяна и разширение.

Може да преправите каквото искате (да извадите методи / класове, да добавяте коментари и т.н.). Само се уверете, че програмата все още работи вярно след промените ви.

Problem 9. Игра

Даден ви е проект, наречен **Application2**. Преименувайте всички идентификатори (имена на променливи, методи, класове, пространства от имена и т.н.), които е нужно, за да направите кода по-четлив и по-лесен за разбиран и за многократна употреба.

Преправете и друго в проекта, не само имената, така че кодът да е добре форматиран и лесен за четене, разбиране и промяна.

Идеи: може да добавяте коментари, за да обясните някои по-сложни места в кода; може да подобрите форматирането на кода; може да извлечете код в методи, класове и пространства от имена; може да приложите ООР принципите и други.

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).