Висококачествени класове

Classes as well as methods should have strong cohesion and loose coupling

Loose coupling – тогава може лесно да подменим едно нещо с друго. Няма смесване на отговорности.

Tight coupling – дадено нещо (напр. UI) не може да се замени с друг UI. Има смесване на отговорности.

Един начин да намалим dependency- то между класовете е вместо един клас да наследява друг клас, да има интерфейс, който се наследява от 2-та класа

Polymorphism – the ability to handle the objects of a specific class as instances of its parent class

* It is a tool to enable code reuse
* Common logic is taken to the base class
* Specific logic is implemented in the derived class

Енкапсулация

* Трябва да правим нещата в класа колкото се може по- малко видими

Minimize the visibility of classes and members (in C# start from **private** and move to **internal, protected** and **public)**

**Anything which is not part of the class interface should be declared private!**

Be suspicious of base classes of which there is only one derived class

* Do you really need this additional level of inheritance?

Когато един клас е static, то този клас не е предназначен за наследяванe

Добре е всички мембъри да се инициализират в конструкторите

Един клас не трябва да знае за своите наследници

Use design patterns for common design situations:

* Creational patterns
* Structural patterns
* Behavioral patterns