Висококачествени методи

**What a method should do:**

**A method should do what its name says or should indicate an error (throw an exception). Any other behavior is incorrect!**

!позволено е даден метод да връща грешен резултат при неговото изпълнение, но този метод трябва задължително да е private, а не public

Strong cohesion is used in engineering – in computer hardware any PC component solves a single task (e.g. hard disk performs a single task – storage)

Дънната платка, например, има **слаб cohesion**, тъй като има памет, видео карта и др. неща на нея

Symptoms of wrong methods:

* The method sometimes returns incorrect result 🡪 bug
* The method returns incorrect result when its input is incorrect 🡪 low quality
* Could be acceptable for private methods only
* The method does too many things 🡪 bad cohesion
* The method has side effects 🡪 spaghetti code
* Method returns strange value when an error condition happens 🡪 it should indicate the error

Methods should have strong cohesion and loose coupling:

Acceptable types of cohesion (присъстват много операции, но резултатът от тези операции и целта им е една – да се изпрати имейл):

SendEmail(recipient, subject, body)

1. Connect to mail server
2. Send message headers
3. Send message body
4. Disconnect from the server

DisplayAnnualExpensesReport(int employeeId)

1. Retrieve input data from database
2. Perform internal calculations over retrieved data
3. Build the report
4. Format the report as Excel worksheet
5. Display the Excel worksheet on the screen

!!ако някой метод прави няколко неща, но тези неща са насочени в една посока (последния пример), то това е приемливо

Ако един метод зависи и от други параметри, освен тези, които са му подадени, то най-вероятно метода има tight coupling

The ideal coupling:

* **Methods depend only on their parameters**
* **Does not have any other input**

!има случаи, в които просто не можем да намалим coupling- а

Начини за намаляване на coupling- а:

* Създаваме даден интерфейс с дадени методи и когато създаваме класове, и тези класове имат нужда от имплементирането на методите в конкретния интерфейс, то всеки клас наследява конкретния интерфейс

Когато пишем параметрите в декларацията на даден метод трябва да подреждаме параметрите по важност (най-важните да са най-отпред)

Не трябва да имаме повече от 7 (+/- 2) параметъра в декларацията на един метод

Не трябва да променяме входните данни вътре в метода (изключение когато имаме out параметри):

Incorrect:

bool CheckLogin(string username, string password)

{

username = username.ToLower();

…

}

Correct:

bool CheckLogin(string username, string password)

{

string usernameLowerCase = username.ToLower();

…

}

Pseudocode – начин да си напишем програмната логика без да използваме конкретен език

**Добре е да правим дизайн на логиката и решението на дадена задача, преди да започнем да пишем самия код. По този начин ще ни е по-лесно, ако решим да пренапишем дадена задача на Java, която първо сме написали на C#, например.**

Заключение: няма перфектно написан код. Винаги може дадени фактори да се променят и да трябва да променяме или дори пренаписваме кода, който сме написали. За това трябва да се стараем да намерим оптималното решение на даден проблем, а не перфектното, просто защото перфектно решение няма.