# Какво е качествен програмен код?

Качествен код е този, който е лесен за разбиране, промяна и поддръжка. При създаването му са спазени общоприетите конвенции за конкретния програмен език, форматирането отговаря на логическата структура на програмата и улеснява разбирането й. Архитектурата и дизайна на програмата са ефективни и логически правилни. Отделните компонентни са възможно най-независими и подлежат на автоматизирано тестване. Създадени са разнообразни тестови сценарии, които покриват възможно най-голяма част от програмния код и следят за неговото коректно поведение. Цели се простота и разбираемост в кода – избягват се сложни и неясни операции(KISS - keep it simple, stupid). Оптимизация за бързодействие трябва да се прави само тогава, когато има нужда от нея. По-важно е кода да е лесен за подръжка, отколкото по-бърз с 0,5%.

От гледна точка на потребителя на софтуера е важно той да изпълнява всички заложени изисквания, да работи достатъчно бързо, да се справя с всички непредвидени ситуации, в които може да се попадне.

За да се създава качествен код е нужно известно усилие за запознаване с отделните правила и конвенции, но в последствие този стил на писане става естествен и носи дългосрочни ползи както за отделния програмист, така и за неговата фирма и индустрията като цяло.

Лично на мен ми допада идеята за самодокументиращ се код. Той се характеризира с:

- ясни и смислени имена на идентификатори;

- кратки и строго специфични методи – един метод върши единствено и само това, което казва името му;

- коректно поведение на отделните компоненти – например при въведени невалидни данни в даден метод правилното поведение е да се хвърли ексепшън.

В крайна сметка добре написаният код няма нужда от коментари, описващи какво прави – всичко е ясно от пръв поглед за човек, запознат с материята. Коментари може да се ползват на едно по-високо ниво на абстракция – например кой алгоритъм се имплементира и защо, а не за очевидни неща – завъртане на цикъл, нулиране на променливи и т.н.

Създаването на качествен код би трябвало да е цел за всеки един програмист. Както вече бе изяснено, в общия случай това не отнема повече време във фазата на имплементация, но в последващите фази на подръжка и доразработка се спестява много.