**Imperative Programming**

Imperative programming consists of sets of detailed instructions that are given to the computer to execute in a given order. It's called "imperative" because as programmers we dictate exactly what the computer has to do, in a very specific way.

**Procedural Programming**

Procedural programming is a derivation of imperative programming, adding to it the feature of functions (also known as "procedures" or "subroutines").

In procedural programming, the user is encouraged to subdivide the program execution into functions, as a way of improving modularity and organization.

**Functional Programming**

Functional programming takes the concept of functions a little bit further.

In functional programming, functions are treated as **first-class citizens**, meaning that they can be assigned to variables, passed as arguments, and returned from other functions.

**Declarative Programming**

Declarative programming is all about hiding away complexity and bringing programming languages closer to human language and thinking. It's the direct opposite of imperative programming in the sense that the programmer doesn't give instructions about *how*the computer should execute the task, but rather on *what* result is needed.

**Object-Oriented Programming**

One of the most popular programming paradigms is object-oriented programming (OOP).

The core concept of OOP is to separate concerns into entities which are coded as objects. Each entity will group a given set of information (properties) and actions (methods) that can be performed by the entity.