What is the difference between object-oriented programming and functional programming?

**What is Object-Oriented Programming?**

Object-Oriented Programming (OOP) is a programming paradigm in computer science that relies on the concept of **classes** and **objects**. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects. There are many object-oriented programming languages, including JavaScript, [C++](https://www.educative.io/blog/how-to-learn-cpp-the-guide-for-beginners), [Java](https://www.educative.io/blog/object-oriented-programming-concepts-java), and [Python](https://www.educative.io/blog/how-to-use-oop-in-python).

OOP languages are not necessarily restricted to the object-oriented programming paradigm. Some languages, such as JavaScript, Python, and PHP, all allow for both procedural and object-oriented programming styles.

A **class** is an abstract blueprint that creates more specific, concrete objects. Classes often represent broad categories, like Car or Dog that share **attributes**. These classes define what attributes an instance of this type will have, like color, but not the value of those attributes for a specific object.

Classes can also contain functions called **methods** that are available only to objects of that type. These functions are defined within the class and perform some action helpful to that specific object type.

## What is functional programming?

Functional programming is a declarative programming paradigm style where one applies pure functions in sequence to solve complex problems. Functions take an input value and produce an output value without being affected by the program. Functional programming mainly focuses on what to solve and uses expressions instead of statements. Functional programming excels mostly at mathematical functions where the values don’t have any correlation and doesn’t make use of concepts like shared state and mutable data used in object-oriented programming.

### **First-class functions**

First-class functions in functional programming are treated as data type variables and can be used like any other variables. These first-class variables can be passed to functions as parameters, or stored in data structures.

### **Recursion**

Unlike object-oriented programming, functional programming doesn’t make use of “while” or ”for” loops or “if-else” statements. Functional programs avoid constructions that create different outputs on every execution. Instead, recursive functions call themselves repeatedly until they reach the desired state or solution known as the base case.

### **Immutability**

In functional programming, we can’t modify a variable after being created. The reason for this is that we would want to maintain the program's state throughout the runtime of the program. It is best practice to program each function to produce the same result irrespective of the program's state. This means that when we create a variable and assign a value, we can run the program with ease fully knowing that the value of the variables will remain constant and can never change.