Programming is a creative process carried out by programmers to instruct a computer on how to do a task. A program is a set of instructions that tells a computer what to do in order to come up with a solution to a particular problem. There are a number of alternative approaches to the programming process, referred to as **programming paradigms**. Different paradigms represent fundamentally different approaches to building solutions to specific types of problems using programming. Most programming languages fall under one paradigm, but some languages have elements of multiple paradigms. Two of the most important programming paradigms are the procedural paradigm and the object-oriented paradigm. Let's look at each of these in a bit more detail.

Procedural programming uses a list of instructions to tell the computer what to do step-by-step. Procedural programming relies on - you guessed it - procedures, also known as routines or subroutines. A procedure contains a series of computational steps to be carried out. Procedural programming is also referred to as imperative programming. Procedural programming languages are also known as top-down languages.

Procedural programming is intuitive in the sense that it is very similar to how you would expect a program to work. If you want a computer to do something, you should provide step-by-step instructions on how to do it. It is, therefore, no surprise that most of the early programming languages are all procedural. Examples of procedural languages include Fortran, COBOL and C, which have been around since the 1960s and 70s.

Object-oriented programming, or **OOP**, is an approach to problem-solving where all computations are carried out using objects. An **object** is a component of a program that knows how to perform certain actions and how to interact with other elements of the program. Objects are the basic units of object-oriented programming. A simple example of an object would be a person. Logically, you would expect a person to have a name. This would be considered a property of the person. You would also expect a person to be able to do something, such as walking. This would be considered a method of the person.

A method in object-oriented programming is like a procedure in procedural programming. The key difference here is that the method is part of an object. In object-oriented programming, you organize your code by creating objects, and then you can give those objects properties and you can make them do certain things.

A key aspect of object-oriented programming is the use of classes. A class is a blueprint of an object. You can think of a class as a concept and the object as the embodiment of that concept. So, let's say you want to use a person in your program. You want to be able to describe the person and have the person do something. A class called 'person' would provide a blueprint for what a person looks like and what a person can do. Examples of object-oriented languages include C#, Java, Perl and Python.