1. **Object-Oriented Programming (OOP)**

**Introduction:**

Object-Oriented Programming (OOP) is a programming paradigm where different components of a computer program are modeled after real-world objects. An object is anything that has some characteristics and can perform a function. It is important to mention here that object-oriented programming is not a language-dependent concept. It is a general programming concept and most of the modern languages, such as Java, C#, C++, and Python, support object-oriented programming

**Example: -**

Consider a scenario where you must develop a Formula 1 car racing game using the object-oriented programming approach. The first thing you need to do is to identify real-world objects in the actual Formula 1 race. What are the entities in a Formula 1 race that have some characteristics and can perform any function? One of the obvious answers to this question is the car. A car can have characteristics like engine capacity, make, model, manufacturer, and so on. Similarly, a car can be started, stopped, accelerated and so on. A driver can be another object in a Formula 1 race. A driver has a nationality, age, gender, and so on, and he can perform functionalities like driving the car, moving the steering or changing the transmission.

1. **Benefits of OOP**

Following are some of the advantages of object-oriented programming:

* Object-oriented programming fosters reusability. A computer program is written in the form of objects and classes, which can be reused in other projects as well.
* The modular approach used in object-oriented programming results in highly maintainable code.
* In object-oriented programming, every class has a specific task. If an error occurs in one part of the code, you can rectify it locally without having to affect other parts of the code.
* Data encapsulation (which we will study later in the article) adds an extra layer of security to the program developed using the object-oriented approach.

1. **Differentiate between function and method**

List down are the major differences between method & function in python.

* A method may alter an object’s state, but Python function usually only operates on it, and then prints something or returns a value.
* Python method is called on an object, unlike a function. Since we call a method on an object, it can access the data within it.
* Method is called by its name, but it is **associated to an object** (dependent). Function is block of code that is also**called by its name.** (independent
* A method is **implicitly passed the object** on which it is invoked. The function can have different parameters or may not have any at all. If **any data (parameters)**are passed, they are **passed explicitly**.
* It **may or may not return any data. Function also may or may not return any data.**
* A method **can operate on the data (instance variables) that is contained by the corresponding class.** Function does not deal with Class and its instance concept.

1. **Define the following terms:**

### Class:

A class in object-oriented programming serves as a blueprint for the object. A class can be considered as a map for the house. You can get an idea of what the house looks like by simply seeing the map. However, a class itself is nothing. For instance, a map is not a house, it only explains how the actual house will look. The relationship between a class and object can be understood by looking at the relationship between a car and an Audi. An Audi is actually a car. However, there is no such thing as a car only. A car is an abstract concept, it is implemented in the form of Toyota, Ferrari, Honda, etc.

**E.g:-**

**class Car:**

### Objects :-

Earlier, we said that a class provides a blueprint. However, to actually use the objects and methods of a class, you need to create an object out of that class. There are few class methods and attributes that can be used without an object. we need to create an object of a class before we can use its methods and attributes. An object is also called an instance; therefore, the process of creating an object of a class is called instantiation. In Python, to create an object of a class we simply need to write the class name followed by opening and closing parenthesis.

**E.g:-**

**ob = MyClass()**

**Attribute:-**

A class attribute is a Python variable that belongs to a class rather than a particular object. It is shared between all the objects of this class and it is defined outside the constructor function, \_\_init\_\_(self,...), of the class. Class Attribute belong to the class itself they will be shared by all the instances. Such attributes are defined in the class body parts usually at the top, for legibility

**Behavior:-**

Objects in Python are generally classified according to their behaviors and the features that they implement. For example, all of the sequence types such as strings, lists, and tuples are grouped together merely because they all happen to support a common set of sequence operations such as *s*[*n*], len(*s*), etc. All basic interpreter operations are implemented through special object methods. The names of special methods are always preceded and followed by double underscores (\_\_). These methods are automatically triggered by the interpreter as a program executes. For example, the operation *x* + *y* is mapped to an internal method, *x*.\_\_add\_\_(*y*), and an indexing operation, *x*[*k*], is mapped to *x*.\_\_getitem\_\_(*k*). The behavior of each data type depends entirely on the set of special methods that it implements.