OOPs Definition

Definition

Object-Oriented Programming is basically a programming style that we used to follow in modern programming. It primarily revolves around classes and objects. Object-Oriented programming or OOPs refers to the language that uses the concept of class and object in programming. The popular object-oriented programming languages are c++, java, python, PHP, c#, etc. The main objective of OOPs is to implement real-world entities such as polymorphism, inheritance, encapsulation, abstraction, etc.

The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

Class

A class is a logical entity used to define a new data type. A class is a user-defined type that describes what a particular kind of object will look like. Thus, a class is a template or blueprint for an object. A class contains variables, methods, and constructors.

Classes are very useful in programming. Consider the example of where you don't want to use just one car but 100 cars. Rather than describing each one in detail from scratch, you can use the same car class to create 100 objects of the type 'car'. You still have to give each one a name and other properties, but the basic structure of what a car looks like is the same.

The car class would allow the programmer to store similar information unique to each car (different models, maybe different colors, etc.) and associate the appropriate information with each car.

Syntax to define a class:-

```
class class_name{
// class body
        //properties
```

```
        //methods
};
```

Here,

- class: class keyword is used to create a class in C++.
- class_Name: The name of the class.
- class body: Curly braces surround the class body.
- After closing curly braces, a semicolon(;) is used.

Object

An object is an instance of a Class. It is an identifiable entity with some characteristics and behavior. Objects are the basic units of object-oriented programming. It may be any real-world object like a person, chair, table, pen, animal, car, etc.
A simple example of an object would be a car. Logically, you would expect a car to have a model number or name. This would be considered the property of the car. You could also expect a car to be able to do something, such as starting or moving. This would be considered a method of the car.

Code in object-oriented programming is organized around objects. Once you have your objects, they can interact with each other to make something happen.

You need to have a class before you can create an object. When a class is defined, no memory is allocated, but memory is allocated when it is instantiated (i.e., an object is created).

Syntax to create an object in C++:

```
class_name objectName;
```

Syntax to create an object dynamically in C++:

```
class_name * objectName = new class_name();
```

Here,

- objectName: It is the name of the object created by class_name.

The class's default constructor is called, and it dynamically allocates memory for one object of the class. The address of the memory allocated is assigned to the pointer, i.e., objectName.

Features of OOPs:-

Four major object-oriented programming features make them different from non-OOP languages:

- **Abstraction** is the property by virtue of which only the essential details are displayed to the user.
- **Inheritance** allows you to create class hierarchies, where a base class gives its behavior and attributes to a derived class.
- **Polymorphism** ensures that it will execute the proper method based on the calling object's type.
- **Encapsulation** allows you to control access to your object's state while making it easier to maintain or change your implementation at a later date.