Question 1: Define Object Oriented Programming Language?

Answer:

Object-oriented programming (OOP) refers to a type of computer programming (software design) in which programmers define the data type of a data structure, and also the types of operations (functions) that can be applied to the data structure.

In this way, the data structure becomes an object that includes both data and functions. In addition, programmers can create relationships between one object and another. For example, objects can inherit characteristics from other objects.

OOPL - Object Oriented Programming Languages:

An object-oriented programming language (OOPL) is a high-level programming language based on the object-oriented model. To perform object-oriented programming, one needs an object-oriented programming language. Many modern programming languages are object-oriented, however some older programming languages, such as Pascal, do offer object-oriented versions. Examples of object-oriented programming languages include Java, C++ and Smalltalk.

Question 2: List down the Benefits of OOP?

Answer:

***Advantages of Object Oriented Programming:***

One of the principal advantages of object-oriented programming techniques over procedural programming techniques is that they enable programmers to create [modules](https://www.webopedia.com/TERM/M/module.html) that do not need to be changed when a new type of object is added. A programmer can simply create a new object that inherits many of its [features](https://www.webopedia.com/TERM/F/feature.html) from existing objects. This makes object-oriented programs easier to modify.

Question 3: Differentiate between function and method?

Answer:

Difference Between methods and function in python

**2. Python Functions – A Revision**

Python function is a sequence of statements that execute in a certain order, given a name. They let us implement code reusability. When we talked about Python Functions, we talked about built-in and user-defined functions.

**a. User-Defined Functions**

Like any other programming language, Python lets us define our own function. Let’s take an example.

1. def **add**(a,b):
2. return a+b
3. **add**(3,-3)

We call this function ‘add’. As expected, it adds two values, which it takes as arguments. When we call it with values -3 and 3, it returns 0. And as you can see, the ‘return’ statement returns a value from Python function.

These let us create a modular program, which in turn lets us make a change easily whenever we want to.

b. Built-in Functions

In our write-up on [Python In-Built Functions](https://data-flair.training/blogs/python-built-in-functions/), we discussed about 66 functions in brief. These are the functions that Python provides us with, and some of these, we see and use commonly. Let’s take a look at a program that uses some of these.

1. def **demofunc**(a,b):
2. """
3. //This function is to demonstrate a few built-in functions in Python
4. """
5. **print**("Begin")
6. **print**(**max**(a,b))
7. **print**(**abs**(a),**abs**(b))
8. **print**(**float**(a),b)
9. **print**(**callable**(a))
10. **print**(**hash**(a),**hash**(b))
11. **print**(**len**('ab'))
12. **print**(**type**(a))
13. for i in **range**(2,4): **print**(i)
14. **demofunc**(2,3)

3. Python Methods

Python method is like a function, except it is attached to an object. We call a method on an object, and it possibly makes changes to that object. A method, then, belongs to a class. Let’s take an example.

>>> class vehicle:

def \_\_init\_\_(self,color):

self.color=color

def start(self):

print("Starting engine")

def showcolor(self):

print(f"I am {self.color}")

>>> car=vehicle('black')

>>> car.start()

First, we defined class ‘vehicle’. Then, we created the object ‘car’ from this blueprint. Here, \_\_init\_\_() is a magic method that serves as a constructor for the class. Method start() and showcolor() let us do what we want to. Remember that Python method must have a parameter ‘self’ to allow them to refer to the current object.

4. Comparison Between Method and Function in Python

Now that we’ve revised the Python method and function, we can compare them. Let’s list down the major differences.

Python method is called on an object, unlike a function. In our example above, we call start() on the object ‘car’. Conversely, we call Python function quite generically- we don’t call it on any object. Since we call a method on an object, it can access the data within it.

A method may alter an object’s state, but Python function usually only operates on it, and then prints something or returns a value.

5. Conclusion

After this discussion, we conclude that there’s a thin line between method and function in python with examples. The only major difference is that we call Python method on an object, but it’s not the same with functions. Also, methods may modify an object; Python functions don’t.

Question 4: Define the following terms:

1. Class

2. Object

3. Attribute

4. Behavior

1. Class: A category of objects. The class defines all the common properties of the different objects that belong to it.
2. Object: a self-contained entity that consists of both data and procedures to manipulate the data.
3. Attribute: An instance attribute is a Python variable belonging to one, and only one, object. 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.
4. Behavior: Object Behavior and Special Methods. Objects in Python are generally classified according to their behaviors and the features that they implement. ... User-defined classes can define new objects that behave like the built-in types simply by supplying an appropriate subset of the special methods described in this section.