**Q1) Define Object Oriented Programming Language?**

**Object-oriented programming** (**OOP**) is a [programming paradigm](https://en.wikipedia.org/wiki/Programming_paradigm) based on the concept of "[objects](https://en.wikipedia.org/wiki/Object_(computer_science))", which can contain [data](https://en.wikipedia.org/wiki/Data), in the form of [fields](https://en.wikipedia.org/wiki/Field_(computer_science)) (often known as *attributes* or *properties*), and code, in the form of procedures (often known as [*methods*](https://en.wikipedia.org/wiki/Method_(computer_science))). A feature of objects is an object's procedures that can access and often modify the data fields of the object with which they are associated (objects have a notion of "[this](https://en.wikipedia.org/wiki/This_(computer_programming))" or "self"). In OOP, computer programs are designed by making them out of objects that interact with one another.[[1]](https://en.wikipedia.org/wiki/Object-oriented_programming#cite_note-1)[[2]](https://en.wikipedia.org/wiki/Object-oriented_programming#cite_note-2) OOP languages are diverse, but the most popular ones are [class-based](https://en.wikipedia.org/wiki/Class-based_programming), meaning that objects are [instances](https://en.wikipedia.org/wiki/Instance_(computer_science)) of [classes](https://en.wikipedia.org/wiki/Class_(computer_science)), which also determine their [types](https://en.wikipedia.org/wiki/Data_type).

Many of the most widely used programming languages (such as C++, Java, Python, etc.) are [multi-paradigm](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language) and they support object-oriented programming to a greater or lesser degree, typically in combination with [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [procedural programming](https://en.wikipedia.org/wiki/Procedural_programming). Significant object-oriented languages include [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [PHP](https://en.wikipedia.org/wiki/PHP), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [Perl](https://en.wikipedia.org/wiki/Perl), [Object Pascal](https://en.wikipedia.org/wiki/Object_Pascal), [Objective-C](https://en.wikipedia.org/wiki/Objective-C), [Dart](https://en.wikipedia.org/wiki/Dart_(programming_language)), [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)), [Common Lisp](https://en.wikipedia.org/wiki/Common_Lisp), [MATLAB](https://en.wikipedia.org/wiki/MATLAB), and [Smalltalk](https://en.wikipedia.org/wiki/Smalltalk).

**Q2) List down the Benefits of OOP?**

1. **Simplicity:** software objects model real world objects, so the complexity is reduced and the program structure is very clear;   
2. **Modularity:** each object forms a separate entity whose internal workings are decoupled from other parts of the system;   
3. **Modifiability:** it is easy to make minor changes in the data representation or the procedures in an OO program. Changes inside a class do not affect any other part of a program, since the only public interface that the external world has to a class is through the use of methods;   
4. **Extensibility:** adding new features or responding to changing operating environments can be solved by introducing a few new objects and modifying some existing ones;   
5. **Maintainability:** objects can be maintained separately, making locating and fixing problems easier;   
**6. Re-usability:** objects can be reused in different programs.

**Q3) Differentiate between function and method?**

**FUNCTIONS:**

A **function** is a piece of code that is called by name. It can be passed data to operate on (i.e. the parameters) and can optionally return data (the return value). All data that is passed to a function is explicitly passed.

**METHODS**:

A **method** is a piece of code that is called by a name that is associated with an object. In most respects it is identical to a function except for two key differences:

1. A method is implicitly passed the object on which it was called.
2. A method is able to operate on data that is contained within the class (remembering that an object is an instance of a class - the class is the definition, the object is an instance of that data).

Q4) Define the following terms:

a. Class

b. Object

c. Attribute

d. Behavior

a) **CLASS:**

A class is an entity that determines how an object will behave and what the object will contain. In other words, it is a blueprint or a set of instruction to build a specific type of object.

b) **OBJECT:**

An object is nothing but a self-contained component which consists of methods and properties to make a particular type of data useful. Object determines the behavior of the class. When you send a message to an object, you are asking the object to invoke or execute one of its methods.

From a programming point of view, an object can be a data structure, a variable or a function. It has a memory location allocated. The object is designed as class hierarchies.

c) **ATTRIBUTE:**

An **attribute** denotes a part of an aggregate object, and so is used during analysis as well as design to express a singular property of the class. Using the language-independent syntax, an attribute may have a name, a class, or both, and optionally a default expression:

d) **BEHAVIOR:**

A class's behavior determines how an instance of that class operates; for example, how it will "react" if asked to do something by another class or object or if its internal state changes. Behavior is the only way objects can do anything to themselves or have anything done to them. For example, to go back to the theoretical Motorcycle class, here are some behaviors that the Motorcycle class might have:

* Start the engine
* Stop the engine
* Speed up
* Change gear
* Stall

To define an object's behavior, you create methods, a set of Java statements that accomplish some task. Methods look and behave just like functions in other languages but are defined and accessible solely inside a class. Java does not have functions defined outside classes