**Question 1**

Define Object Oriented Programming Language?

Object-oriented programming (OOP) is a programming language model in which programs are organized around data, or [objects](https://searchmicroservices.techtarget.com/definition/object), rather than functions and logic.

**Principles of OOP**

Object-oriented programming is based on the following principles:

* [Encapsulation](https://searchnetworking.techtarget.com/definition/encapsulation)- The implementation and state of each object are privately held inside a defined boundary, or class. Other objects do not have access to this class or the authority to make changes but are only able to call a list of public functions, or methods. This characteristic of [data hiding](https://searchsqlserver.techtarget.com/definition/data-hiding) provides greater program security and avoids unintended [data corruption](https://searchsqlserver.techtarget.com/definition/data-corruption).
* [Abstraction](https://whatis.techtarget.com/definition/abstraction)- Objects only reveal internal mechanisms that are relevant for the use of other objects, hiding any unnecessary implementation code. This concept helps developers make changes and additions over time more easily.
* [Inheritance](https://whatis.techtarget.com/definition/inheritance)- Relationships and subclasses between objects can be assigned, allowing developers to reuse a common logic while still maintaining a unique hierarchy. This property of OOP forces a more thorough data analysis, reduces development time and ensures a higher level of accuracy.
* [Polymorphism](https://whatis.techtarget.com/definition/polymorphism)- Objects are allowed to take on more than one form depending on the context. The program will determine which meaning or usage is necessary for each execution of that object, cutting down on the need to duplicate code.

**Question 2**

List down the Benefits of OOP?

The benefits of OOP are:

* It provides a clear ***modular structure*** for programs which makes it good for defining abstract data types in which implementation details are hidden.
* Objects can also be ***reused*** within an across applications. The reuse of software also lowers the cost of development. More effort is put into the object-oriented analysis and design, which lowers the overall cost of development.
* Reuse also enables ***faster development***. Object-oriented programming languages come with rich libraries of objects, and code developed during projects is also reusable in future projects.
* It provides a good framework for code libraries where the supplied software components can be ***easily adapted and modified by the programmer***. This is particularly useful for developing graphical user interfaces.

**Question 3**

Differentiate between function and method?

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.

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).

**Question 4**

Define the following terms:

1. Class

2. Object

3. Attribute

4. Behavior

**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.

**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.

**Attribute:**

An attribute is an element that takes a value and is associated with an object, such as an item, a region, a page. An example of such an element is Author, whose value is typically the name of the object creator. Typically, an attribute value is provided by a user, though there are some attributes that provide their own default values. For example, the Publish Date attribute has a default value of the current date and time. In most cases, users can revise an attribute's default value.

**Behavior:**

Behavioral Programming (BP) is an approach and technique for software development, which enables incremental development in a natural way. A behavioral application consists of threads of behavior each of which represents an independent scenario that the system should and shouldn’t follow. These independent behavior threads are interwoven at run-time yielding integrated system behavior. For example, in a game-playing application, each game rule, and each playing strategy would be programmed separately and independently with little or no awareness of other modules.