**Q2.)What is OOP? List OOP Concepts.**

**Answer :-**

**OOP** is an Object-Oriented Programming Language. It has a sweeping impact because it appeals at multiple levels and promises faster and cheaper development and maintenance. It follows a bottom-up approach to develop applications.

The world object-oriented is the combination of two words i.e. object and oriented. **Object** means an article or entity that exists in the real world. **Oriented** means interested in a particular kind of thing or entity. In Layman’s terms, it is a programming pattern that rounds around an object or entity are called **object-oriented programming**.

It is basically a computer programming design philosophy or methodology that organizes models software design around data, or objects rather than functions and logic. An **object** is referred to as a data field that has unique attributes and behavior. Everything in **OOP** is grouped as self-sustainable **objects**.

It is the most popular programming model among developers. It is well suited for programs that are large, complex and actively updated or maintained.

A real world example of **OOP** is the automobile. It more completely illustrates the power of **object-oriented** design.

**Points to Remember**

* Everything is an object
* Developer manipulates object that uses message passing
* Every object is an instance of a class
* The class contains the attribute and behavior associated with an object

**Pillars of OOP**

The major concepts that we have discussed above are known as **pillars of OOPs**.

There are four pillars on **OOP** rests.

1. Encapsulation
2. Inheritance
3. Data Abstraction
4. Polymorphism

**OOP Concept**

The OOPs concepts are following:

1. Object
2. Class
3. Encapsulation
4. Inheritance
5. Data Abstraction
6. Polymorphism
7. **Object** :- An **Object** is a real world entity that has attributes, behavior and properties. It is referred to as an instance of the class. It contains member functions, variables that we have defined in the class. It occupies space in the memory. Different objects have different states or attributes and behaviors.
8. **Class** :- A **Class** is a blueprint of an **object**. It is a user-defined data type. Inside a class, we define variables, constants, member functions and other functionality. It binds data and functions together in a single unit. It does not consume memory at run time.

**Note** that classes are not considered as a data structure. It is a logical entity. It is the best example of data binding. **Note** that a class can exist without an object but vice-versa is not possible.

1. **Encapsulation** :- Encapsulation is a mechanism that allows us to bind data and functions of a class intro an entity. It protects data and functions from outside interference and misuse. Therefore, it also provides security. A class is the best example of encapsulation.

**Class**

**{**

**Data members**

**+ ENCAPSULATION**

**Methods (behavior)**

**}**

1. **Inheritance** :- The concept allows us to inherit or acquire the properties of an existing class into a newly created class. It is known as inheritance. It provides code reusability.
2. **Data Abstraction** :- The concept allows us to hide the implementation from the user but shows only essential information to the user. Using the concept developer can easily make changes and added over time.

**Advantages**:

* It reduces complexity
* It avoids delicacy
* Eases the burden of maintenance
* Increase security and confidentially

1. **Polymorphism** :- The word polymorphism is derived from the word **ploy** and **morphs**. Poly means many and morphs means forms. It allows us to create methods with the same name but different method signatures. It allows the developer to clean, sensible, readable and resilient code.