Four main Pillars of OOPS

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

What is polymorphism in OOPs?

**1. Polymorphism** is one of the core concepts of object-oriented programming (OOP) and describes situations in which something occurs in several different forms. In computer science, it describes the concept that you can access objects of different types through the same interface.

In Object-Oriented Programming (OOPS) language, there are two types of polymorphism as below:

**Static Binding** (or Compile time) Polymorphism, e.g., Method Overloading.

**Dynamic Binding** (or Runtime) Polymorphism, e.g., Method overriding.

Why we use polymorphism in oops

Polymorphism allows us to perform a single action in different ways. In other words, polymorphism allows you to define one interface and have multiple implementations. The word “poly” means many and “morphs” means forms, So it means many forms.

**2. Inheritance in OOP** = When a class derives from another class. The child class will inherit all the public and protected properties and methods from the parent class. In addition, it can have its own properties and methods. An inherited class is defined by using the extends keyword.

**Difference between parents class and child class**

Parent class is the class being inherited from, also called base class. Child class is the class that inherits from another class, also called derived class.

**3. Encapsulation** is one of the fundamentals of OOP (object-oriented programming). It refers to the bundling of data with the methods that operate on that data. Encapsulation is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties' direct access to them.

In object-oriented programming, abstraction is one of three central principles (along with encapsulation and inheritance). Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.

**4. Abstraction** is a design level process and it is used to reduce the complexity at the designing stage of a project. Encapsulation is an implementation level process, and it is used to provide privacy and maintain control over the transparency of data at the implementation stage of a project.

What is **constructor**

In class-based, object-oriented programming, a constructor (abbreviation: ctor) is a special type of subroutine called to create an object. It prepares the new object for use, often accepting arguments that the constructor uses to set required member variables.

What is constructor over loading

Constructor overloading means having more than one constructor with the same name. Constructors are methods invoked when an object is created. You have to use the same name for all the constructors which is the class name. This is done by declaration the constructor with a different number of arguments.

What is constructor overriding

In any object-oriented programming language, Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent class

What is **destructor**

A destructor is a member function that is invoked automatically when the object goes out of scope or is explicitly destroyed by a call to delete . A destructor has the same name as the class, preceded by a tilde ( ~ ). For example, the destructor for class String is declared: ~String() .

What is function overloading

Function overloading is a feature of object-oriented programming where two or more functions can have the same name but different parameters. When a function name is overloaded with different jobs it is called Function Overloading.

What is function overriding

Function overriding is a concept in object-oriented programming which allows a function within a derived class to override a function in its base class, but with a different signature (and usually with a different implementation).

Why we **use oops concept in c#**

OOP provides a clear structure for the programs. OOP helps to keep the C# code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug. OOP makes it possible to create full reusable applications with less code and shorter development time.