OOPs or Object-Oriented Programming is a programming approach that centers on organizing a program around its data and well-defined interfaces, with the aim of making code more closely aligned with the real world.

Class- Blueprint of object. It is a collection of similar types of objects.
- It is a user-defined data type. Also, it doesn't occupy any memory.
- It is composed of name, attributes, and methods. Access Modifiers (public, private, default, protected) are used for having limited or public access to that class so that it can be used by other programs in Java.

Object- Instance of class. It is a building blocks of OOP.
- Objects are real-world entities which have a certain state and behavior.

Class	Object
Fruit	Apple, Mango
Plane	P1, P2

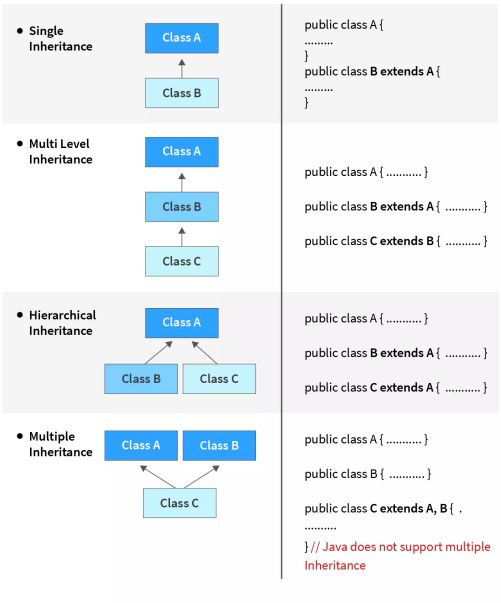
Member Functions- The functions are written inside a class are known as member functions.
Instance variables- Variables that are declared within the scope of the class.

Four Pillars of OOPs in Java

Inheritance-

- It is a mechanism of deriving one class from another.
- It allows the derived class to inherit features from a parent class.
- Reusability of code can be achieved because of inheritance.

Parent Class/ Superclass/ Base: A class from which member functions and data members are used by another class.
Derived class/ Subclass/ Child class: A class that uses the properties of the base class and can also add its properties.



SCALER Topics

Types of Inheritance

```
Class Shape{
   String color;
}
Class Triangle extends Shape{
}
```

Single Inheritance
Class A <— — Class B
B extends A

Example

```
Public class OOPs{
   psvm(){
   Triangle T1=new Triangle();
   T1.color="red";
}}
```

Triangle(Dervied Class/ Subclass) extends Shape(Parent Class/ Superclass)

• Single Inheritance-

```
Class Triangle extends Shape{
   public void area(int l, int h){
      sysout(1/2*l*h);}
Class EquilateralTriangle extends Triangle{
   public void area(int l, int h){
      sysout(1/2*l*h);
}}
```

• Multilevel Inheritance-

Class Shape <— Class Triangle <— Class Equilateral Triangle

• Hierarchical Inheritance-

```
Class Shape{
   ……
}
Class Triangle extends Shape{
   ……
}
Class Circle extends Shape{
   ……
}
```

Polymorphism-

• It means many forms.
• It is the ability to create a function, variable, or an object that has more than 1 form.

Poly - Many Morphism - Form

Types of Polymorphism in Java:

• Compile Time Polymorphism- Also known as static or early binding. The binding is performed during compilation time.
• Run Time Polymorphism- Also known as dynamic or late binding. The binding occurs during runtime, depending on the type of object.

Types of Polymorphism are achieved in Java by:

- Overloading- In Java, it is possible to create methods that have the **same name**, but **different parameter** lists and different definitions. This is called overloading.
- Overriding : It is a technique of re-implementing or re-writing a method of a superclass in its subclass.

Function Overloading —> Compile Time Polymorphism
Function Overriding —> Run Time Polymorphism

```
Class Student{
   String name;
   Int age;
public void printInfo(String name){
   sysout(name);
public void printInfo(int age){
   sysout(age);
public void printInfo(String name, int age){
   sysout(name+" "+age);

public class OOPs{
   psvm(){
   Student S1=new Student();
   S1.name="";
   S2.age=;
   S1.printInfo(S1.name, S1.age);
  }
}
```

Encapsulation-

- It is the process of binding data and methods together in a single unit.
- This is done to protect the data from outside interference and to camouflage the implementation from the rest of the program.
- In encapsulation, data in the class is hidden from other functions and classes.
- Data Hiding + Abstraction = Encapsulation
- Data Hiding - Used by Access Modifier

```
public class Plane{
  private int plane_number;
}
```

Abstraction-

- Hiding the implementation details and showing only important/useful part to the user.
- Focus on the essential details rather than the irrelevant things.
- Show- Important parts of the user. Hide- Non important things

Data Hiding- The process of protecting members of class from unintended changes.

Access Modifiers in Java

- It help to restrict the scope of a class, constructor, variable, method, or data member.
- It provides security, accessibility, etc to the user depending upon the access modifier used with the element.

- public: It is accessible everywhere.
- protected: Accessed within the class and outside the package through child class.
- private: Accessible only within the class.
- default (declared/defined without using any modifier): It is accessed within the package.

	default	private	protected	public
same class	yes	yes	yes	yes
same package subclass	yes	no	yes	yes
same package non-subclass	yes	no	yes	yes
different package subclass	no	no	yes	yes
different package non-subclass	no	no	no	yes