# object oriented programming(oops)

**Introduction of oops:**

- Class
- Method
- Object
- Encapsulation
- Inheritance
- Abstraction
- Polymorphism

## oops concept:

- Object Oriented Programming Structure
- OOPS is a method of implementation in which programs are organized as collection of objects, class and methods abstraction.

**Oops principals are:**

1. Class
2. Method
3. Object
4. Encapsulation
5. Inheritance
6. Abstraction
7. Polymorphism

**Class:**

Class is nothing but collection of methods or collection of objects.

- Project name: Should be in Pascal notation
- Pascal notation: Each word of the first letter should be in capital
- src - Source file

- Class name: Pascal notation
- Package creation: ex, org.cts.scope-All small letters.

**Method:**

Set of action to be performed.

- Method name: camel notation.
- Camel notation: First word should be small after every word of the first letter should be capital.

**Object:**

- Run time memory allocation.
- Using object we call the any methods.

Example program:

Student database

```
public class StudentInfo {

        public void Studentname() {
                System.out.println("Name:Vengat");
        }
         public void studentList() {
                 System.out.println();
        }
         public void StudentMark() {
                 System.out.println("Mark:1005");
        }
         public void StudentAddress() {
                 System.out.println("Address: Chennai");
        }
        public static void main(String[] arg) {
                StudentInfo info = new StudentInfo();
                 info.Studentname();
                 info.StudentMark();
                 info.StudentAddress();
        }

}
```

**Encapsulation:**

- Encapsulation in java is a mechanism to wrap up variables(data) and method(code) together as a single unit.
- It is the process of hiding information details and protecting data and behavior of the object.
- It is one of the four important oop concepts.
- The encapsulate class is easy to test,so it also better for unit testing.

**INHERITANCE:**

- Inheritance is an important pillar of oop(object oriented programming).
- It is the mechanism in java by which one class is allowed to inherit the features of another class.
- We can achieving inheritance by using extends keyword.inheritances is also known as "is-a" relationship.
- We can access one class property into another class using 'extend' keyword and reuseable.

## Types of inheritances:

1. Single Inheritance
2. Multilevel Inheritance
3. Multiple Inheritances
4. Hybrid Inheritance
5. Hierarchical Inheritance

### 1.Single inheritance:

One parent class is directly support into one child class using extend keyword.

### 2.Multilevel inheritance:

One child class and more than one parent class

### 3.Multiple Inheritances:

More than one parent class parallely support into one child class but it won't suport in java.

## 4.Hybrid Inheritance:

It is a combination of single and multiple inheritance.

## 5.Hierarchical Inheritance:

One parent class and more than one child class.

**Abstraction:**

- Data abstraction is the process of hiding certain details and showing only essential information to the user.
- Abstraction can be achieved with either abstract classes or interfaces.
- The abstract keyword is a non access modified, used for classes and method.

❖ Abstract class:

The class that cannot be used to create object.

❖ Abstract method:

That can only be used in an abstract class, and it does not have a body. the body is provided by the subclass.

- it has 2 types,

1.Partially abstraction

2.Fully abstraction

1.partially abstraction:

- It will support abstract method and non-abstract method.
- We can’t create object for abstract class because in the method signature we didn't mention any business logic. so
- In abstract method, we only mention abstract signature, won't create business logic.so
- It have 2 class, abstract class (sub class) and super class. we create object and business logic only in super class, won't create in abstract class.

2.Fully abstraction:

- It will support only abstract method, won't support non abstract method
- In interface "public abstract" is default. we no need to mention
- It using implements keywords.

**Polymorphism:**

Imagine you have a piece that can change its shape based on where you put it in your structure.in the programming world, depending on the situation, a function can act a little differently.

- ➢ Poly-many
- ➢ Morphism-forms
- ➢ Taking more than one forms is called polymorphism or one task completed by many ways

❖ It has 2 types,

1.Method overloading polymorphism
2.Method overriding polymorphism.

1.Method overloading polymorphism:

- ➢ Class-same
- ➢ Method-same
- ➢ Argument-differ
- ➢ In a same class method name is same and the argument is different is called method overloading
- ➢ The argument is depends on

- data types
- data types count
- data type order

2.Method overriding polymorphism:

- ➢ Class name-differ(using extends)
- ➢ Method-same
- ➢ Argument- same
- ➢ In a different class , the method name should be same and argument name should be same is called overriding.

## Application of oops:

- Now we have a basic idea of what object oriented programming means ,now let’s look at some of the application of oops.
    - Client server system
    - Object oriented database
    - Automation system
    - Real time system
    - Parallel programming

### Features of oops in java:

- Higher priority is focused on data rather than functions
- Objects communicate with each other through functions
- An object is a group of data and method.
- New data and methods can be easily added whenever needs.
- A bottom up approach is adopted in programming design.

### Conclusion of oops:

- Object-Oriented Programming (OOP) is of paramount importance in Java and software development as a whole.
- By embracing OOP principles such as modularity, encapsulation, inheritance, polymorphism, and abstraction, developers can create modular, maintainable, and scalable Java applications.
- OOP empowers developers to design and implement software systems more effectively, ensuring code reusability, extensibility, and collaboration.
- Understanding the significance of OOP in Java is key to mastering the language and building robust and efficient applications.