Object oriented programming

This is a programming practice or a convention in which we are going to declare any real time entity as an object.

The major features of an OOP is that it provides.

1. **<u>Communication (Relationships)</u>**
    a) Has a relationship - (Aggregation)
    - ◇ Driven and driver class

    b) Is a relationship - (Inheritance)
    - ◇ **super** keyword
    - ◇ Types of inheritance
    - ◇ Multiple Inheritance
    - ◇ Non Primitive Typecasting
    - ◇ **instanceof** keyword
    - ◇ Real time example

2. **<u>Versatility (Polymorphism)</u>**
    a) Static / Compile - Time polymorphism
    - – Overloading
        - ◆ Method Overloading
        - ◆ Constructor Overloading

    b) Dynamic / Runtime Polymorphism)
    - – Overriding
    - – @Override Annotation
    - – Method Shadowing

3. **<u>Privacy (Abstraction)</u>**
    a) **abstract** keyword
    b) Abstract class
    c) Concrete class and concrete Methods
    d) Abstract class and Abstract methods
    e) Overriding an Abstract methods
    f) Interface
    g) Multiple inheritance in interface

4. **<u>Security (Encapsulation)</u>**

a) Access Modifiers
b) Getter-Setter methods

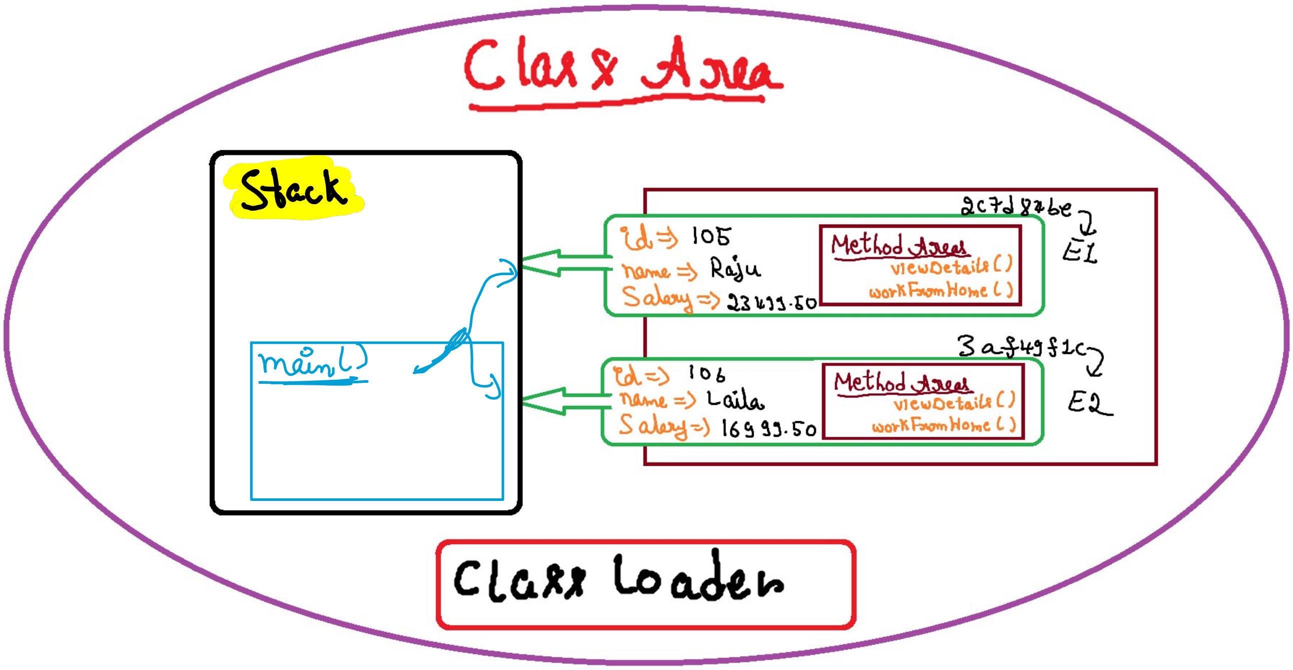
Example for OOP program

```
//* Back end class

class Employee //Object
{

//  ? this is the common data shared by multiple
    public static int EmpCount = 105;

//  ? the global non static variables -> Data Members -> Properties/States
    public int id;
    public String name;
    public double salary;

//  & The job of constructor is to initialize the variables

//  ? No argument constructor
    public Employee()
    {
        this.id = EmpCount++;
    }
    //! initialize to default

//  ? parametrized constructor
    public Employee( String name, double salary)
    //! initialize to user-defined values
    {

    //^ this call statement is calling other constructors by using the args
        this();

    //^ this keyword is used to call the members of current class
        this.name=name;
        this.salary=salary;
    }

//  ?The non static Methods perform the operation required - Behaviours
    public void workFromHome()
    {
        System.out.println(this.name +" is working");
    }

    public void viewDetails()
    {
        System.out.println("+--------------------------+");
        System.out.println("| Name        : "+this.name+"\t\t |");
        System.out.println("| Employee ID : "+this.id+"\t\t\t |");
        System.out.println("| Salary      : "+this.salary+"Rs\t |");
        System.out.println("+--------------------------+");
    }
}
```

```
public class A1 //^Execution class
{
    public static void main(String[] args)
    {
        //To create an object we use a Class Variable containing instance
    //^ ClassName  [Var_name] ; Here we are using the ClassName as Datatype
        Employee   E1    ;
    /*
        ^ By calling the constructor we will have a copy of the class created
        ^ which contains the data of employee class and gets created in heap
        ^ area.
    */
        E1 =  new Employee( "Raju", 23499.50);
        // ? the new keyword is allocating the memory
        // ? the constructor is generating an instance

        System.out.println( E1 );
        E1.viewDetails();
        E1.workFromHome();


        System.out.println("<--------------------------------------------->");

        Employee E2= new Employee("Laila", 16999.50);
        System.out.println(E2);
        E2.viewDetails();
        E2.workFromHome();
    }

}
```

Type equation here.

```
Output : [Running] cd "c:\Programming\OOP\" && javac A1.java && java A1

Employee@2c7b84de
+----------------------------+
| Name        : Raju         |
| Employee ID : 105          |
| Salary      : 23499.5Rs    |
+----------------------------+
Raju is working
<--------------------------------------------->
Employee@3af49f1c
+----------------------------+
| Name        : Laila        |
| Employee ID : 106          |
| Salary      : 16999.5Rs    |
+----------------------------+
Laila is working

[Done] exited with code=0 in 2.315 seconds
```



```
public class A2
{
    //~ We can create Global objects also but it has its own rule

    //& A non static member
    public String variable ="Object's data member";

    static//! static anonymous block
    {
        System.out.println(  new A2().variable   );
        System.out.println( "------------------" );
    }

    public static void main(String[] args)
    {
        //! Main method
        System.out.println(  new A2().variable   );
        System.out.println( "------------------" );

        A2.test();
    }

    public static int test()
    {
        //! Any static method
        System.out.println(  new A2().variable   );
        System.out.println( "------------------" );

        return 0;
    }
}
```

```
[Running] cd "c:\Programming\OOP\" && javac A2.java && java A2
```

```
Object's data member
------------------
Object's data member
------------------
Object's data member
------------------

[Done] exited with code=0 in 1.965 seconds
```

```
public class A3
{
    String var = "Instance Variable";

    A3 obj1 ;
    //! Non static Object

    static A3 obj2 = new A3();
    //! Global Static Object

    public static void main(String[] args)
    {
        A3 obj = new A3();
        System.out.println(obj );
        System.out.println(obj.obj1 +"-->"+(obj.obj1=new A3()));
        System.out.println(obj.var);
        System.out.println(obj.obj1.var+"-->"+(obj.obj1.var = "First"));
        System.out.println(obj.obj1.obj2);//static through object
        System.out.println(A3.obj2);
        System.out.println(A3.obj2.var);
    }
}
```

```
[Running] cd "c:\Programming\OOP\" && javac A3.java && java A3

A3@76ed5528
null-->A3@6b884d57
Instance Variable
Instance Variable-->First
A3@38af3868
A3@38af3868
Instance Variable

[Done] exited with code=0 in 1.862 seconds
```

```
class Connection
{
    String db;
    void connect()
    {
        System.out.println(db +" database is connected to program");
    }
}
public class A4
{
```

```java
    //! global static object - reduces the unnecessary recreation of object
    public static Connection c;
    static
    {
        c= new Connection();

        c.db="Java";
    }
    public static void main(String[] args)
    {
        c.connect();
        System.out.println(c);
        test();
    }
    public static void test()
    {
        System.out.println(c);
        c.connect();
    }
}
```

```
[Running] cd "c:\Programming\OOP\" && javac A4.java && java A4

Java database is connected to program
Connection@a09ee92
Connection@a09ee92
Java database is connected to program

[Done] exited with code=0 in 2.135 seconds
```