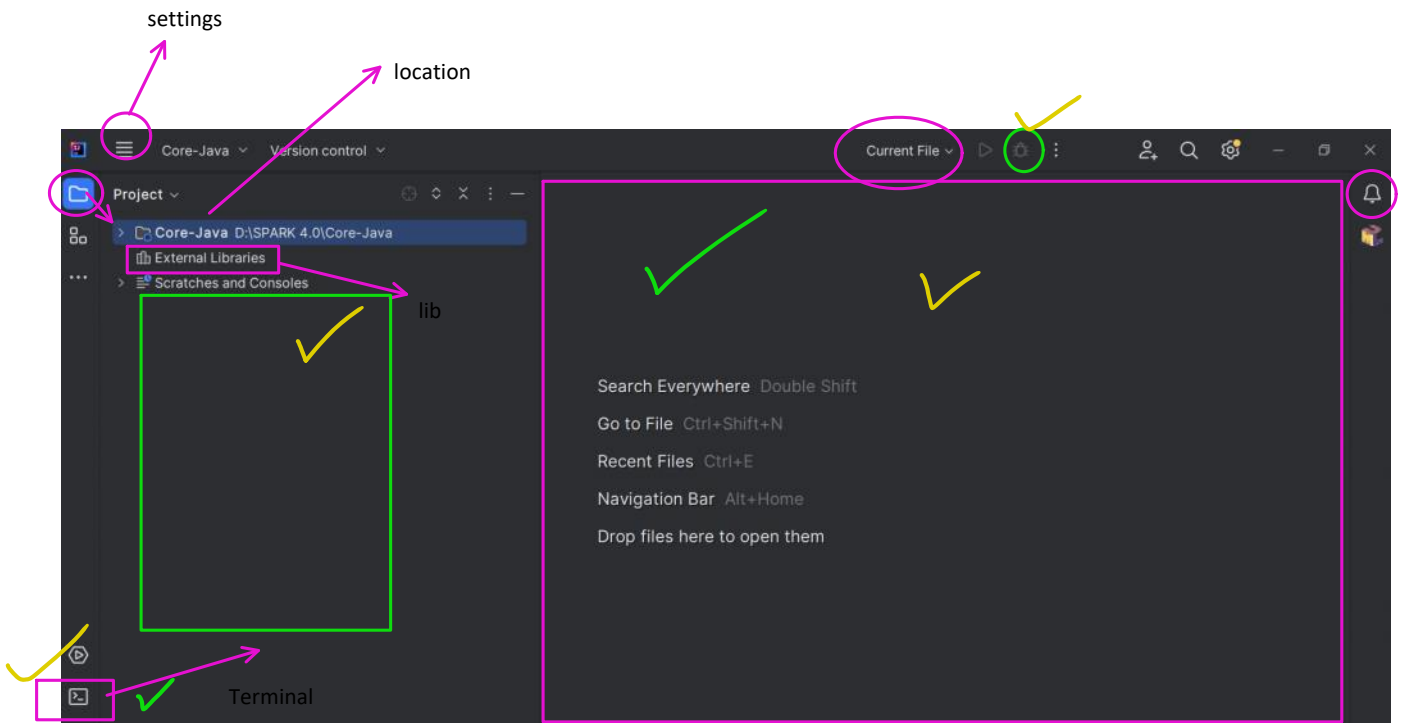


OOPs

07 June 2025 09:15

1. Download IntelliJ



Why this is IDE ?

integrated development environment :

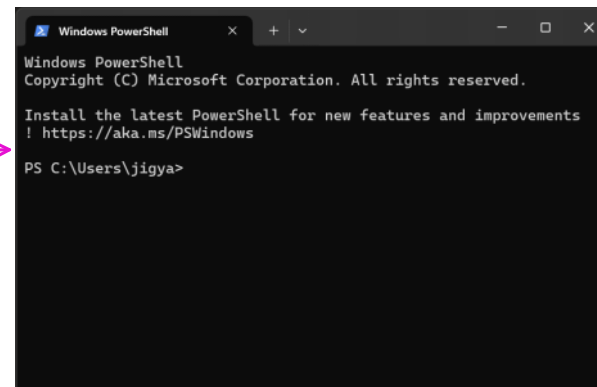
```

class Student
{
    int rollNo=123; //non-static
    String name="rahul";
    String address="delhi";
    static int id=333;

    //ye ho app
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
        //System.out.println(rollNo); // no
        System.out.println(id);

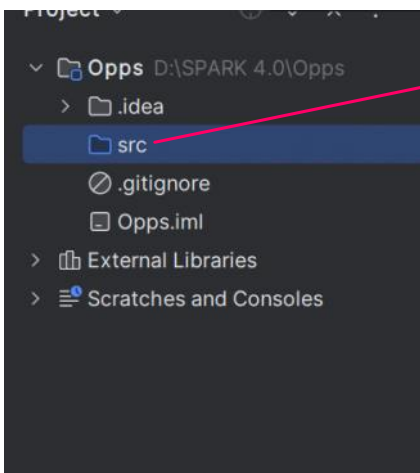
        //create object
        Student s=new Student();
        System.out.println(s.rollNo);
        System.out.println(s.name);
    }
}

```



Development
Debug
Rapid Development

result



Java code

What is OOps ?

OOPs is computer programming model that organizes software around data or object rather than function and logic.

```

public class Car {

    //fields , attributes, properties
    no usages
    String color;
    no usages
    String brand;
    no usages
    int speed;

    //behaviour(methods) //actions
    no usages
    public void run()
    {
        System.out.println("Car us running");
    }
}

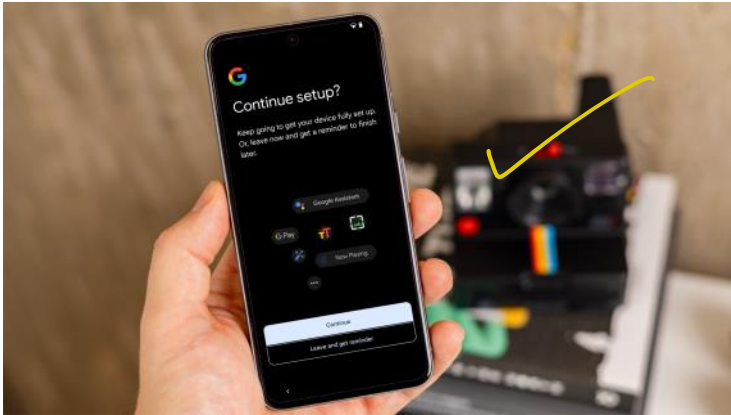
```

To interact with this class we need object

What is Constructor ?

Note: A constructor is special method in java used to create and initialize object

Note:



Eg: When we buy a phone, setup window comes in picture which help us to set lang, location, time, wifi setting

Rule:

1. Has same name as class
2. No return type
3. Called by JVM automatically
4. Setting default values of instance variable and creating object

Type of constructor

1. Default : by default used by jvm
2. Palettized : When I want to restrict object creation
3. Overloaded

Note : if we are not creating any constructor then java make default constructor

String name;
int duration;

String name;
int duration;



Course c= new Course();



Course c2= new Course();

Note: we if are not creating any constructor then automatically default const will be there and called by JVM

If we create para constructor then jvm will not create default one

When we use para cons ?

1. When we want to restrict object creation based on parameter
2. To set instance variable value at time of object creation

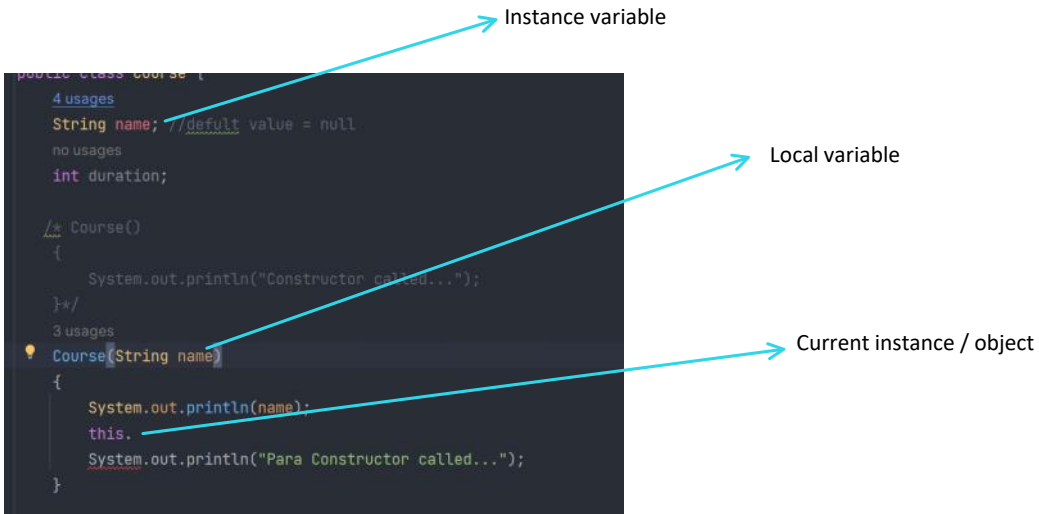
Homework:

create a class name Student with 4 fields

Add para cons and method

Create object with different para

Course c=new Course()
c.name //instance



```
public class Course {
    4 usages
    String name; //default value = null
    no usages
    int duration;

    /* Course()
    {
        System.out.println("Constructor called...");
    }*/
    3 usages
    Course(String name)
    {
        System.out.println(name);
        this.
        System.out.println("Para Constructor called...");
    }
}
```

What is this ?

The this is the keyword which refer to the current object

Why we use it ?

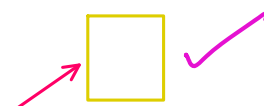
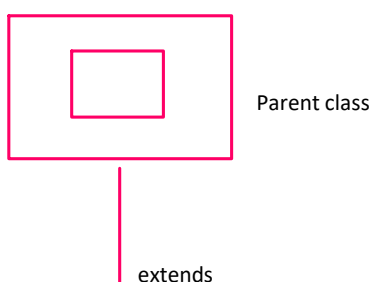
1. Reference to current object / and to make difference between local and instance variable
2. Call current class constructor
3. this as method argument

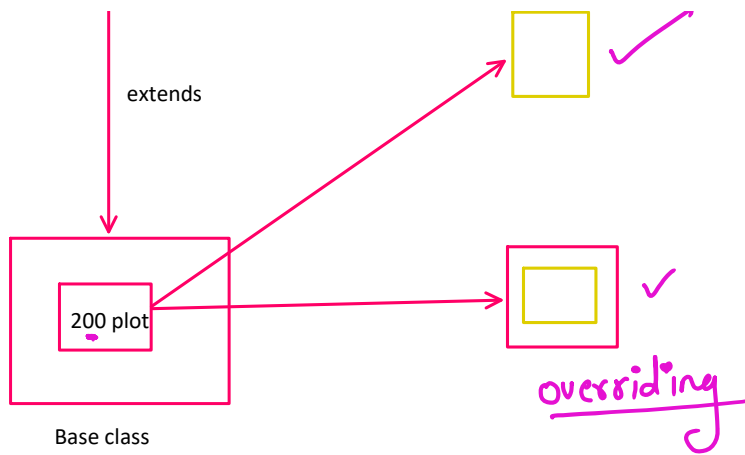
There are typically Three principles in OPPs

- ✓ 1. Encapsulation : process of binding data and methods in single unit is called encapsulation
2. Inheritance :
3. Polymorphism :

Abstraction is not considered as sperate principle in OOP, but is considered as concept that is utilized in all the pillars mentioned above.

- ✓ 4. Abstraction : hiding internal details and showing necessary info is called abstraction
hiding : private
Show : method





Type of inheritance

1. Single - one child one parent
2. Multilevel - child ---> parent ----> grand parent
3. Hierarchical -- one parent , multiple child
4. Multiple -> Not supported

Super keyword

1. Refer to parent object
2. Call parent constructor

Polymorphism in java

Polymorphism : Many Forms



Lock --- one press
Trun off -- long press

1 button multiple kam kr raha ha

Two Types

1. Compile time = Method overloading
2. Run time = Method Overriding

Method signature : name + Para

Note: Method overloading means declare multiple method with same name bit having different method

signature

In method overloading we have 3 rules

1. Method name must be same
2. List of parameter must be different
3. Return type is not considered In method overloading

Overriding

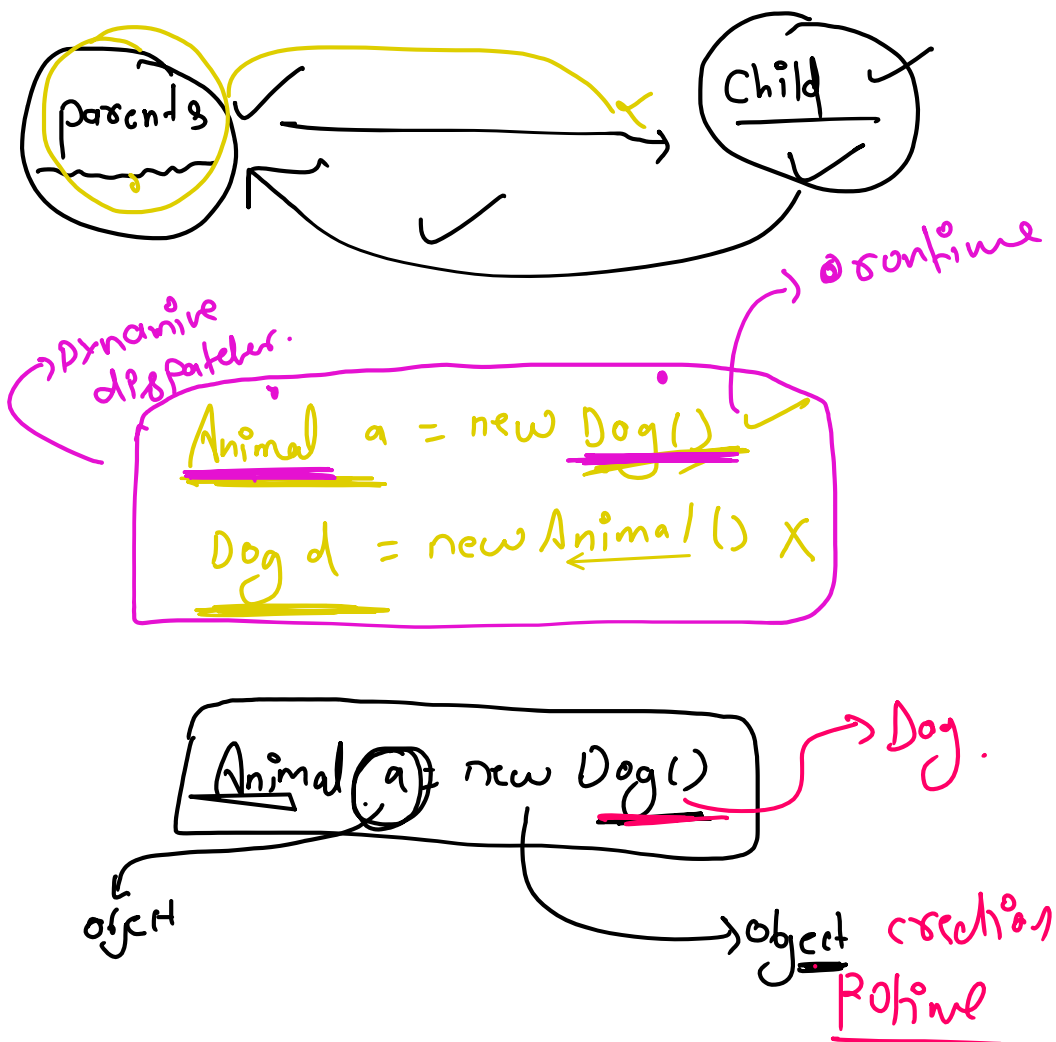
Method overriding means change parent imp

Rules

1. Method name must be same
2. List of para must be same
3. Return type must be same
4. Private, final, static methods cannot be overridden
5. There must be IS-A relationship between classes (Inheritance)

✓
✓
Calculator cal=new Calculator();

cal.add(19,19) ----> at compile time



Compile time : method resolution should take care by jvm at **compile time**

Runtime : method resolution should take care by jvm at **runtime**