Super Keyword

Definition:

The super keyword in Java is a reference variable which is used for referring to an immediate parent class object.

Need for *super* keyword:

The most common use of the ***super*** keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

Usage of super keyword:

1. ***super*** can be used to refer to the immediate parent class instance variable.
2. ***super*** can be used to invoke immediate parent class method.
3. ***super*** can be used to invoke immediate parent class constructor.

Super keyword as:

1. Parent class variable:

```
class Base
{
    String color="white";
}
class Derived extends Base
{
```

```
    String color="black";
    void printColor()
    {
        System.out.println(color);
        System.out.println(super.color);
    }
}
class Main
{
    public static void main(String args[])
    {
        Derived d=new Derived();
        d.printColor();
    }
}
```

OUTPUT

Black
White

Explanation:

super.color accesses the parent class color variable.

2. Parent class method:

```
class Base
{
    void eat()
    {
        System.out.println("eating...");
    }
}
class Derived extends Base
{
    void eat()
    {
```

```
        System.out.println("eating bread...");
    }
    void bark()
    {
        System.out.println("barking...");
    }
    void work()
    {
        super.eat();
        bark();
    }
}
class Main
{
    public static void main(String args[])
    {
        Derived d=new Derived();
        d.work();
    }
}
```

OUTPUT:

eating…
barking…

Explanation:

super.eat() accesses the parent class eat method.

3. Parent class constructor:

```
    class Base
    {
        Base()
        {
            System.out.println("Base class is invoked");
        }
```

```
}
class Derived extends Base
{
    Derived()
    {
        super();
        System.out.println("Derived is invoked");
    }
}
class Main
{
    public static void main(String args[])
    {
        Derived d=new Derived();
    }
}
```

OUTPUT:

Base is invoked
Derived is invoked

Explanation:

super() accesses the parent class constructor.

Important points to remember:

1. Call to super() must be the first statement in the derived class constructor.
2. If a constructor does not explicitly invoke a superclass constructor, the Java compiler automatically inserts a call to the no-argument constructor of the superclass. If the superclass does not have a no-argument constructor, you will get a compile-time error.

3. If a subclass constructor invokes a constructor of its superclass, either explicitly or implicitly, you might think that a whole chain of constructors called, all the way back to the constructor of Object. This, in fact, is the case. It is called *constructor chaining*..

Real time example:

```
class Person{
    int id;
    String name;
    Person(int id,String name){
        this.id=id;
        this.name=name;
    }
}
class Emp extends Person{
    float salary;
    Emp(int id,String name,float salary){
        super(id,name);//reusing parent constructor
        this.salary=salary;
    }
    void display(){
        System.out.println(id+" "+name+" "+salary);}
}
class TestSuper5{
```

```
    public static void main(String[] args){
        Emp e1=new Emp(1,"ankit",45000f);
        e1.display();
    }
}
```

OUTPUT:

1 ankit 45000

Difference between ***super*** and ***this*** keyword:

	super()	**this()**
Definition	super() - refers to an immediate parent class instance.	this() - refers to the current class instance.
Invoke	Can be used to invoke immediate parent class method.	Can be used to invoke current class method.
Constructor	super() acts as immediate parent class constructor and should be first line in child class constructor.	this() acts as a current class constructor and can be used in parameterized constructors.

Override	When invoking a superclass version of an overridden method the super keyword is used.	When invoking a current version of an overridden method the this keyword is used.

Practice problems:

- Hackerrank:Method overriding