=> Inheritance :-

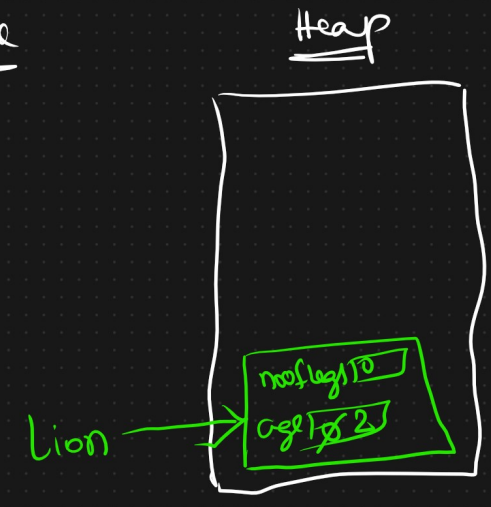
=> super & this

this() & this

this => current running object

super => Parent

```
class Animal1
{
    int age;

    Animal1()
    {
        age=2;
        System.out.println("Zero Param Animal Cons");
    }
    Animal1(int age)
    {
        this.age=age;
        System.out.println(" Param Animal Cons");
    }
}
class Lion extends Animal1
{

    int noOfLegs;

    Lion()
    {
        super();
        System.out.println("Zero Param Lion Cons");
    }

    Lion(int noOfLegs)
    {
        this.noOfLegs=noOfLegs;
        System.out.println(" Param Lion Cons");
    }
     void disp()
     {
         System.out.println(age);
         System.out.println(noOfLegs);
     }
}
public class LaunchInh5
{

    public static void main(String[] args)
    {

        Lion lion=new Lion();
        lion.disp();
    }

}
```



Thread t1 = new Thread();

class Demo extends Thread

{ ≡ }

Demo d = new Demo();

```
class Animal1
{
    int age;

    Animal1()
    {
        age=2;
        System.out.println("Zero Param Animal Cons");
    }
    Animal1(int age)
    {
        this.age=age;
        System.out.println(" Param Animal Cons");
    }
}
class Lion extends Animal1
{
    //int age;
    int noOfLegs;

    Lion()
    {
        System.out.println("Zero Param Lion Cons");
    }
    Lion(int noOfLegs)
    {
        //super();
        this.noOfLegs=noOfLegs;
        System.out.println(" Param Lion Cons");
    }
     void disp()
     {
         System.out.println(age);
         System.out.println(noOfLegs);
     }
}
public class LaunchInh5
{

    public static void main(String[] args)
    {

        Lion lion=new Lion();
        lion.disp();

        Lion lion2=new Lion(4);
        lion2.disp();
    }

}
```

Zero para Param

```java
class Animal1
{
    int age;

    Animal1()
    {
        age=2;
        System.out.println("Zero Param Animal Cons");
    }
    Animal1(int age)
    {
        this.age=age;
        System.out.println(" Param Animal Cons");
    }
}
class Lion extends Animal1
{
    int noOfLegs;
    Lion()
    {
        //super();
        this(4);
        System.out.println("Zero Param Lion Cons");
    }
    Lion(int noOfLegs)
    {
        //super();
        super(6);
        this.noOfLegs=noOfLegs;
        System.out.println(" Param Lion Cons");
    }
     void disp()
     {
         System.out.println(age);
         System.out.println(noOfLegs);
     }
}
public class LaunchInh5
{

    public static void main(String[] args)
    {
        Lion lion=new Lion();
        lion.disp();

        Lion lion2=new Lion(4);
        lion2.disp();
    }

}
```

Console

Param Animal cons

Param Lion cons

6

4

Heap

age 6

noOfLegs 0 4

Lion

```java
    int age;

    Animal1()
    {
        age=2;
        System.out.println("Zero Param Animal Cons");
    }
    Animal1(int age)
    {
        this.age=age;
        System.out.println(" Param Animal Cons");
    }
}
class Lion extends Animal1
{
    int noOfLegs;
    Lion()
    {
        super(6);
        System.out.println("Zero Param Lion Cons");
    }
    Lion(int noOfLegs)
    {
        this();
        this.noOfLegs=noOfLegs;
        System.out.println(" Param Lion Cons");
    }
     void disp()
     {
         System.out.println(age);
         System.out.println(noOfLegs);
     }
}
public class LaunchInh5
{

    public static void main(String[] args)
    {
        Lion lion2=new Lion(4);
        lion2.disp();
    }
```

Param Animal cons

Zero Param Lion cons

Param Lion cons

4

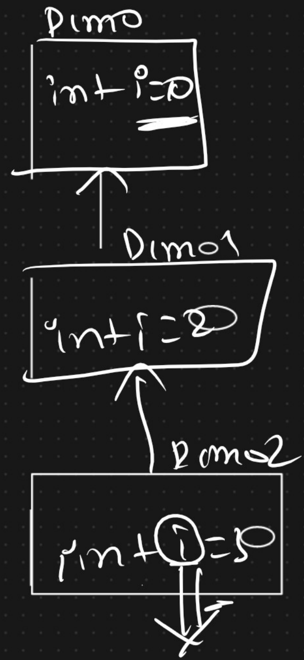
4

Heap area

noOfLegs 0 4

age 0 6

Lion2



i ✓
super.i
✗ super.super.i ✗

✓ Constructor execution → Inheritance

{ super() this()

super this }

⇒ Packages :-

⇓

Folder ⇒ Files Calculator Addition

sub

⇒ New Project ⇒ 4 ⇒ add

↳ 4 = files Multi

4 ⇒ sub

4 = fw

4 ⇒ mul

4 = 2)

4 ⇒ div

4 =

⇓

Package :-

=> Package :- domainName . comp . —

calcul => — com.telusko.calc.add ⇓ 4 files
— add
sub
— mul

4 Access Modifiers || Specifiers

- public
- protected
- default
- private

} class, var, method, constructor

default
void disp()
{
}

default
int a;

private int age;

public void disp()
{
}

protected void disp()
{
}

① int age;

	Within same class ?	with same Package d/f class	outside package d/f class but IS-A ✓	outside Package d/f class no relationship
Public	✓	✓	✓	✓
protected	✓	✓	✓	✗
default	✓	✓	✗	✗
private	✓	✗	✗	✗

=>

Visibility increasing ↑

public

protected

default

private

↓ visibility decreases