Constructor in Inheritance

- Constructor are executed from Base class to derived class.
- we can call the Base class constructor explicitly through super() in derived class.
- we can call same class constructor using this()

Super this keywords In Java

this ; applied to variable, method, constructor.

- This is a keyword which acts as reference variable
- this reference variable contains address of current object.
- this is a instance reference variable and can't be accessed from static context.

use of this

① To access the variables

Syntax :

this.<variableName>

Ex: this.a
this.b

- This keyword in Java is used inside a method or Constructor of a class it is used to refer to a member of current object within the instance of method or constructor

(2) To access the methods

Syntax:

```
this.<methodName>();
```

Ex:

```
this.m1();
this.m2();
```

(3) To Access the overloaded constructor

Syntax:

```
this(Parameters);
```

Ex:

```
this();          ← Invoke default constructor
this(99);        ← Invoke 1-arg constructor
this(75,85);     ← Invoke 2-arg constructor.
```

- call to constructor by using this(); must be first line of constructor only.
- JVM never puts automatically this() keyword like super()
- If we write call to constructor explicitly by using this() the super() call to constructor will not be put by JVM.

```
class ThisExample {

int a;
    ThisExample()
     {
    sop(" From default Constructor");
     }

    ThisExample(int a) {

     this();
    sop("From 1-arg Constructor");
      this.a=a;

     }

void display()
   {
     sop(a);
   }
```

```
  psvm()
   {
     ThisExample te = new ThisExample(99);
       te.show();

     }
```

. when you have same name for local variables and class level then in the following.

(a) refer the local variable directly

(b) Refer the class level instance variable using this keyword.

Super :

. Super is a keyword and it will be used to refer the members of immediate super class.

. super keyword can't be accessed from static context

. super keyword can be used in three ways.

(1) To access immediate super class variable

Syntax:

```
super.<variableName>
```

Ex:

```
super.a
super.b
```

(2) To access the immediate super class methods

Syntax:

```
super.<methodName>();
```

Ex:

```
super.m1();
super.m2();
```

(3) To Access immidiate superclass constructor.

Syntax:

super (Parameters);

Ex:-

super(); ← invoke immediate superclass default constructor

super(99); ← " " " " 1 arg constructor

super(75, 95); ← " " " " 2 arg constructor.

- The keyword super is placed inside a subclass to call a method from a superclass.
- super can be applied to variable, constructor, method
- super call to constructor must be at first line of all constructors.

But not for super.variable or super.m1() ....

- Recursive call to this and super will be CTE (Compile time Error)
- when you have same name for local variable, class level variable and superclass variable, then do following

(a) refer the local variable directly
(b) refer the class level variable using this keyword
(c) refer the super class level variable using super keyword.

- Super keyword can be used to access both instance and static members.
- when you are writing any super statement inside the constructor then default super will not be inserted by the java compiler
- The first statement of any constructor can be either this or super, can't be both at a time.
- order of constructor execution is from super class to subclass
- Private members of super class can't be accessed using super keyword.

```
public class Person {

int age = 45;

    Person ()
    {
    Sop(" I am from Base constructor Default");

    }

    method1()
    {
    Sop("I am from Base method1 ");

    }

}

class Employee extends Person {

  int age = 29;

  int Eid ;

  ~~void~~ ~~Em~~ Employee ( )
      {
      Sop (" I am from Default ala ? Emp class");

      }

    Employee ( int id)
    {
        super();
    Sop (" I am from 1arg constructor");
     this.Eid = id ;

    }
```

```
void method2()
{
    SOP("value of age" + a);
    SOP("value of age" + this.a);
    SOP("value of age" + super.age);
    SOP("value of Eid" + this.Eid);
    SOP("value of Eid" + Eid);
}

psvm()
{
    Employee emp1 = new Employee();
    emp1.method1();
    Employee emp2 = new Employee(912);
    emp2.method2();
}
```