## <u>2.5: “super” keyword</u>

- ✓ Super is a special keyword that directs the compiler to invoke the superclass members. It is used to refer to the parent class of the class in which the keyword is used.
- ✓ **super keyword is used for the following three purposes**:
  1. To invoke superclass constructor.
  2. To invoke superclass members variables.
  3. To invoke superclass methods.

### 1. <u>Invoking a superclass constructor:</u>

- ✓ **super** as a standalone statement(ie. super()) represents a call to a constructor of the superclass.
- ✓ A subclass can call a constructor method defined by its superclass by use of the following form of **super**:

```
super();
or
super(parameter-list);
```

- ✓ Here, parameter-list specifies any parameters needed by the constructor in the superclass.
- ✓ **super( )** must always be the first statement executed inside a subclass constructor.
- ✓ The compiler implicitly calls the base class’s no-parameter constructor or default constructor.
- ✓ If the superclass has parameterized constructor and the subclass constructor does not call superclass constructor explicitly, then the Java compiler reports an error.

### 2. <u>Invoking a superclass members (variables and methods):</u>

**(i) Accessing the instance member variables of the superclass: Syntax:**

```
super.membervariable;
```

**(ii) Accessing the methods of the superclass:Syntax:**

```
super.methodName();
```

This call is particularly necessary while calling a method of the super class that is overridden in the subclass.

- ✓ If a parent class contains a finalize() method, it must be called explicitly by the derived class’s finalize() method.

```
super.finalize();
```

**Example:**

```
class A        // super class
{
   int i;
   A(String str)      //superclass constructor
   {
      System.out.println(" Welcome to "+str);
   }
   void show()        //superclass method
   {
      System.out.println(" Thank You!");
   }
}
class B extends A
{
   int i;      // hides the superclass variable 'i'.
   B(int a, int b)     // subclass constructor
   {
      super("Java Programming");      // invoking superclass constructor
      super.i=a;      //accessing superclass member variable
      i=b;
   }
   // Method overriding
   @Override
   void show()
   {
      System.out.println(" i in superclass : "+super.i);
      System.out.println(" i in subclass : "+i);
      super.show();      // invoking superclass method
   }
}
public class UseSuper {
   public static void main(String[] args) {
      B objB=new B(1,2);      // subclass object construction
      objB.show();       // call to  subclass method show()
   }
}
```

**Output:**

```
Welcome to Java Programming
                i in superclass : 1
                i in subclass : 2
                Thank You!
```

**Program Explanation:**

In the above program, we have created the base class named **A** that contains a instance variable **'i'** and a method **show().** Class A contains a parameterized constructor that receives string as a parameter and prints that string. Class **B** is a subclass of **A** which contains a instance variable 'i' ( hides the superclass variable 'i') and overrides the superclass method **show().** The subclass defines the constructor with two parameters a **and b.** The subclass constructor invokes the superclass constructor **super(String)** by passing the string "Java Programming" and assigns the value **a** to the superclass variable**(super.i=a)** and **b** to the subclass variable. The show() method of subclass prints the values of 'i' form both superclass and subclass & invokes the superclass method as **super.show().**

In the main class, object for subclass **B** is created and the object is used to invoke **show()** method of subclass.