Super

- The super keyword in Java is a reference variable which is used to refer immediate parent class object.
- It is used to call superclass methods, and to access the superclass constructor.
- The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.
- To understand the super keyword, you should have a basic understanding of Inheritance and Polymorphism.

Uses of Super

- Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.

1. super can be used to refer 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 – using with variable

- super is used to refer immediate parent class instance variable.
- We can use super keyword to access the data member or field of parent class. It is used if parent class and child class have same fields.
- If a class has a variable with name same as the variable of the superclass then, you can differentiate the superclass variable from the subclass variable using the super keyword.
- general form:
- super.member
- Here, member can be either a method or an instance variable.

Super – using with variable

```
• class Computer{
        String color="white";
  }
  class Dell extends Computer{
        String color="black";
        void displayColor(){
              System.out.println(color);          //prints color of Dell class
              System.out.println(super.color);//color of computer class
        }
• }
  class TestSuper1{
  public static void main(String args[]){
        Dell d=new Dell();
        d.displayColor();
    }
• }
```

Super – using with methods

- This is used when we want to call parent class method. So whenever a parent and child class have same named methods then to resolve ambiguity we use super keyword.
- class Base{
-         void message(){
-           System.out.println(“Now at base class method ");
-         }
- }

Super – using with methods

- class Child extends Base{

          void message(){

                super.message();

                System.out.println(“Now at child class method");

      }

- }

  class SuperMethod{

          public static void main(String args[]){

                   Faculty s=new Faculty();

-                    s.message();

-          }

- }

Super as a Constructor

- The super keyword can also be used to access the parent class constructor by adding ‘()’ after it, i.e. super().
- One more important thing is that ‘super()’ can call both parametric as well as non-parametric constructors depending upon the situation.
- Calling a constructor of a super-class from the constructor of a sub-class:

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

- Must occur as the very first instructor in the sub-class constructor:

```
class SuperClass { … }
        class SubClass extends SuperClass {
                SubClass(…) {
                super(…);
                }
        …
}
```

Example: Super Constructor

```
•   class Computer{
•           Computer(){
                 System.out.println(“Computer class Constructor");
            }
    }
•       class Dell extends Computer{
             Dell(){
                 super();
                 System.out.println(“Dell class Constructor");
             }
•       }
      class TestSuper1{
           public static void main(String args[]){
•                Dell d=new Dell();
•       }
•   }
```

Difference between super and super()

Super	Super()
The super keyword in Java is a reference variable that is used to refer parent class objects.	The super() in Java is a reference variable that is used to refer parent class constructors.
super can be used to call parent class’ variables and methods.	super() can be used to call parent class’ constructors only.
The variables and methods to be called through super keyword can be done at any time,	Call to super() must be first statement in Derived(Student) Class constructor.
If one does not explicitly invoke a superclass variables or methods, by using super keyword, then nothing happens	If a constructor does not explicitly invoke a superclass constructor by using super(), the Java compiler automatically inserts a call to the no-argument constructor of the superclass.