Tutorial Link https://course.testpad.chitkara.edu.in/tutorials/this() and super() in java/5b1bb1887becc0459dea56c0

TUTORIAL

this() and super() in java

this() and super() both are special in terms of constructors.

- this() is used to call another constructor of same class. It can be used when we need to use some default values for missing values.
- super() is used to call a constructor of super class. It is required when we need to pass some arguments to base class and complete object instantiation.

In some cases, you might want to define several constructors where each of the constructors takes some subset of the arguments used to create an object. One constructor takes all the arguments and supplies the code that constructs the object, and the other constructors take a partial list of arguments, supply default values, and invoke the main constructor. As an example, consider the example shown below:

```
class AreaObject
{
        double x, y, width, height;
        public AreaObject(double x, double y, double width,
double height)
        {
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
        }
        public AreaObject(double x, double y)
        {
                this(x,y,10,10);
        }
        public AreaObject()
        {
                this(1,1);
        }
}
```

The first AreaObject() constructor takes the arguments x, y, width, and height and assigns them to the appropriate field variables. The second AreaObject() constructor takes just the AreaObject's coordinates as arguments and invokes the first constructor, passing 10 as the default value for the AreaObject's height and width. The third constructor has no arguments and invokes the second constructor with the default value of 1 for the AreaObject's x and y coordinates.

The this(x,y,10,10) and this(1,1) used in the second and third constructors is a special notation provided by Java to enable you to invoke a constructor of the same class from another constructor of that class. It is referred to as a constructor call statement. In order to use a constructor call statement, there must be a constructor whose argument list matches those supplied with this().

In other cases, instead of invoking the constructor of the same class as the object being constructed, you invoke the constructor of its superclass. To do this, you use the superclass constructor call statement, super(). To see how super() works consider the MyAreaObject class declared by the following code: -

```
class MyAreaObject extends AreaObject
{
        int length;
        public MyAreaObject(double x, double y, double width,
double height, int length)
        {
                super(x,y,width,height);
                this.length = length;
        }
        public MyAreaObject(int length)
        {
                super(10,10,100,100);
                this.length = length;
        }
        public MyAreaObject()
        {
                super();
                this.length = 20;
        }
}
```

The MyAreaObject class extends AreaObject with the capability to specify the AreaObject's length. The first constructor invokes super(x,y,width,height) to invoke the superclass (AreaObject) constructor to initialize the AreaObject's location and dimensions. The second constructor also uses the superclass constructor and passes it the default values of 10 for the AreaObject's x and y coordinates and 100 for its width and height. The third constructor simply invokes super() to invoke the parameterless AreaObject() constructor of its superclass and instantiate length to 20.

So, this comes in two variants in Java. First is this keyword which refers to current object and second is this() as a self constructor calling mechanism. Similarly super can be used in two ways, First to refer a variable of super class and secondly, to call a constructor of base class.