- **Name of Teacher:** Miss Nita N. Bargale
- **Class:** B.Sc. Computer Science (Entire)- III
- **Course Title:** Advanced Java programming

Constructor:-

- In java, a constructor is a block of codes similar to the method..
- It is a special type of method which is used to initialize the object.
- Constructors are automatically called when an object is created. At the time of calling constructor, memory for the object is allocated in the memory

- Rules for creating Java constructor
- Constructor name must be the same as its class name
- Constructors do not have a return type; not even void.
- A Java constructor cannot be abstract, static, final, and synchronized

- Types of Java constructors
- There are two types of constructors in Java:

1. Default constructor (no-arg constructor)
2. Parameterized constructor

**Default constructor (no-arg constructor**)

- Constructor with no arguments is known as no-arg constructor
- A constructor is called "Default Constructor" when it doesn't have any parameter.

Syntax:

```
class class_nm
{
class_nm()
{
---
}

}
```

```
class Test
 {
 Test()
{
// constructor
  body
 }
}
```

- Here, Test() is a constructor. It has the same name as that of the class and doesn't have a return type.

```
class Student3
{
int  id ;
String name ;
Student3()
{
 id=101;
 name=“ram”;
}
void display()
{
System.out.println(id+" "+name);
}
}
class  studentdemo
{
public static void main(String args[])
{
  Student3 s1=new Student3();
  s1.display();
}
}
```

- Parameterized constructor
- A constructor having an parameter list is known as a parameterized constructor.
- The parameter list can be specified in the parentheses in the same way as parameter list is specified in the method

```
class Rectangle
{
    int length;
    int breadth;
   //constructor to initialize length and breadth of rectangle
    Rectangle( int  l, int  b)
     {
         length = l;
           breadth= b;
  }
   //method to calculate area of rectangle
    int area()
    {
       return (length * breadth);
}
 }
```

```
//class to create rectangle objects and calculate area
class Paraconstructor
{
public static void main(String[] args)
    {
    Rectangle firstRect = new Rectangle(5,6);
     Rectangle secondRect = new Rectangle(7,8);
     System.out.println("Area of First Rectangle  : " +firstRect.area());
System.out.println("Area of Second Rectangle  : "+secondRect.area());
    }
}
```

- In this example, the class Rectangle contains a parameterized constructor. The specified values passed to this constructor are used to initialize instance variables **length** and breadth of an object. On the execution of the statement,
- **Rectangle firstRect = new Rectangle (5, 6);**
- the arguments 5 and 6 are passed to the parameters a and b of the Rectangle object's constructor which are then used to initialize length and breadth instance variables.
- Similarly it works for the other object secondRect.

- Difference between Constructor and Method

  1. The purpose of constructor is to initialize the object of a class while the purpose of a method is to perform a task by executing java code.
- 2. Constructors cannot be abstract, final, static and synchronised while methods can be.
- 3. Constructors do not have return types while methods do

- Constructor overloading
- Java Constructor overloading is a technique in which a class can have any number of constructors that differ in parameter list. The compiler differentiates these constructors by taking into account the number of parameters in the list and their type.
- Constructor overloading is similar to Java method overloading, we can also create two or more constructors with different parameters. This is called constructors overloading.

```
class Person
{
Person(String name)
{
System.out.println("Name of person = "+name);
}
Person(String name, String voterId)
{
System.out.println("Name of person = "+name );
System.out.println("Voter ID of " +name+ " = "+ voterId);
}
public static void main (String [] args)
{
Person person1 = new Person("Ravi");
Person person2 = new Person("Ram", "12345678");
```

N.N.Bargale