

## =>variables declared inside the method =>Memory will be given inside stack area =>Once the control enters inside the method

Local variables

memory will be given.
=>Once the control leaves the method memory will be taken out. =>No default value will be given to the local variables

```
class LoanApp
    //static variable
    static float rateOfInterest = 9.5f;
 public class TestApp
    public static void main(String[] args)
        System.out.println(LoanApp.rateOfInterest);
static variable
```

class Student

- a. Memory will be given in the Method-Area.
- b. Memory will given at the time of loading .class
- c. Default value will be given if user won't specify any value. d. Memory will be taken out at the time of unloading
- the .class file. e. Static variables can be accessed in 2 ways
- a. using ClassName.❖
- b. using reference of the object.

```
//instance variable
    String name;
    int age;
    //constructor
    Student(String name ,int age){
       this.name = name;
       this.age = age;
    //instance method
    public void disp(){
      System.out.println("Name is :: "+name);
       System.out.println("Age is :: "+age);
public class TestApp
   public static void main(String[] args)
      Student std1 = new Student("sachin",51);
      std1.disp();
```

## Student std2 = new Student("kohli",37); std2.disp();

## D:\Decode Java1.0Batch>javac TestApp.java D:\Decode Java1.0Batch>java TestApp Name is :: sachin Age is :: 51

Age is :: 37

Name is :: kohli

StackArea	Instance Area
local wantable	
local variable	instance variable
main()	
main()	
LoanApp.rateOfInterest	
	<pre>local variable  main()</pre>

MethodArea	StackArea	HeapArea
Static variables	Local variables	instance variable
Student.class	Student() <del>name = kohli</del>	(default value)
TestApp.class	<pre>mame = sachin age = 51 Student() main() std1 std2</pre>	name = null age = 0 51  this  name = null age = 0 37  this