```
class Test
{
  1 static int i = 10;  7
  2 static
    {
        methodOne(); 8
        System.out.println("First static block");10
    }
  3 public static void main(String[] args)
    {
        methodOne(); 13
        System.out.println("Inside main method"); 15
    }
  4 public static void methodOne()
    {
        System.out.println(j); 9 , 14
    }
  5 static
    {
        System.out.println("Second static block");11
    }
  6 static int j = 20;  12
}
```

Output
0
First static block
Second static block
20
Inside main method

Steps followed by JVM

1. Identification of static members from top to bottom

i = 0 j = 0	RIWO Read Indirect Write only	Step 1 to 6

2. execution of static varaibles assignments and static block execution from top to bottom

i = 10 j = 20	R&W Read and Write only	Step 7 to 12

3. Execution of main method Step 13 to 15

```
class Test
{
  1 static int i = 10;  4
  2 static
    {       //direct read
        System.out.println(i); 5
    }
  3 public static void main(String[] args)
    {

    }
}
```

i = 0[RIWO]
i = 10[R&W]

Output
10

```
class Test
{
  1 static
    {       //direct read
        System.out.println(i);
    }
  2 public static void main(String[] args)
    {

    }
  3 static int i = 10;
}
```

i = 0 [RIWO]
Read Indirectly

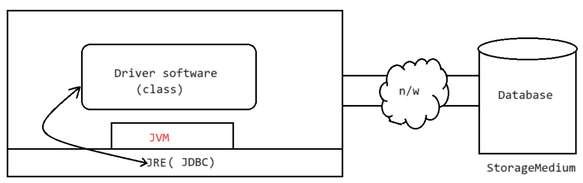
Output
CompiletimeError:Illegal forward reference

```
class Test
{
  1 static
    {
        methodOne(); 5
    }
  2 public static void main(String[] args)
    {
                8
    }
  3 public static void methodOne()
    {       //indirect read
        System.out.println(i); 6
    }
  4 static int i = 10;  7

}
```

i = 0 [RIWO]
Read Indirectly
i = 10[R&W]

Output
0



Capable of setting environment for java program execution

```
class Driver
{
    static
    {
      //Register the driver with DriverManager(Setting JDBC environment)
    }
}
```

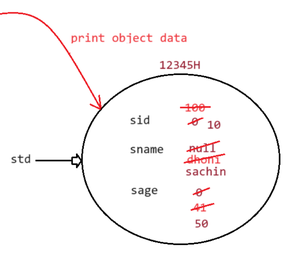
```
class Student
{
    //instance variable
    int sid;
    String sname;   3
    int sage;

    //constructor: shadowing -> resolved through "this"
    Student(int sid, String sname,int sage)
    {
        this.sid   = sid;
        this.sname = sname;                    5
        this.sage  = sage;
    }

    //instance block
    {
        sid = 100;
        sname = dhoni;   4
        sage = 41;
    }

    //instance method
    public void dispStdDetails()
    {
        System.out.println("SID   is :: "+sid);
        System.out.println("SNAME is :: "+sname);
        System.out.println("SAGE  is :: "+sage);
    }
}

class Test
{
    public static void main(String[] args) 1
    {
        Student std = new Student(10,"sachin",50);2
                      constructor
        std.dispStdDetails();
    }
}
```



1. new
2. load the class[only one time]
3. scan the instance variable allocate memory fill with default value.
4. scan for the instance block, if found load and execute.
5. make a call to constructor and give the values supplied by programmer
6. return the reference.

```
class Test
{
  3 int i = 10;  9
  4 {
        methodOne(); 10
        System.out.println("First instance block"); 12
    }
  5 Test()
    {
        System.out.println("Test class constructor"); 15
    }
    public static void main(String[] args)  1
    {
        Test t =new Test(); 2
        System.out.println("Inside main method"); 16
    }
  6 public void methodOne()
    {
        System.out.println(j); 11
    }
  7 {
        System.out.println("Second instance block"); 13
    }
  8 int j = 20;  14
}
```

Output
0
First instance block
Second instance block
Test class constructor
Inside main method

1. Identification of instance members from top to bottom.

i = 0 j = 0	RIWO

2. Execution of instance variable assignment and instance block execution from top to bottom

i = 10 j = 20	R&W

3. Execution of constructor