Instance Block OR Non static Block OR Instance Initializer :

* It is a special block in java which will be automatically executed at the time of creating the object.
Example :

```
{
   //instance block
}
```

* Every time we will create an object in java, instance block will be executed.

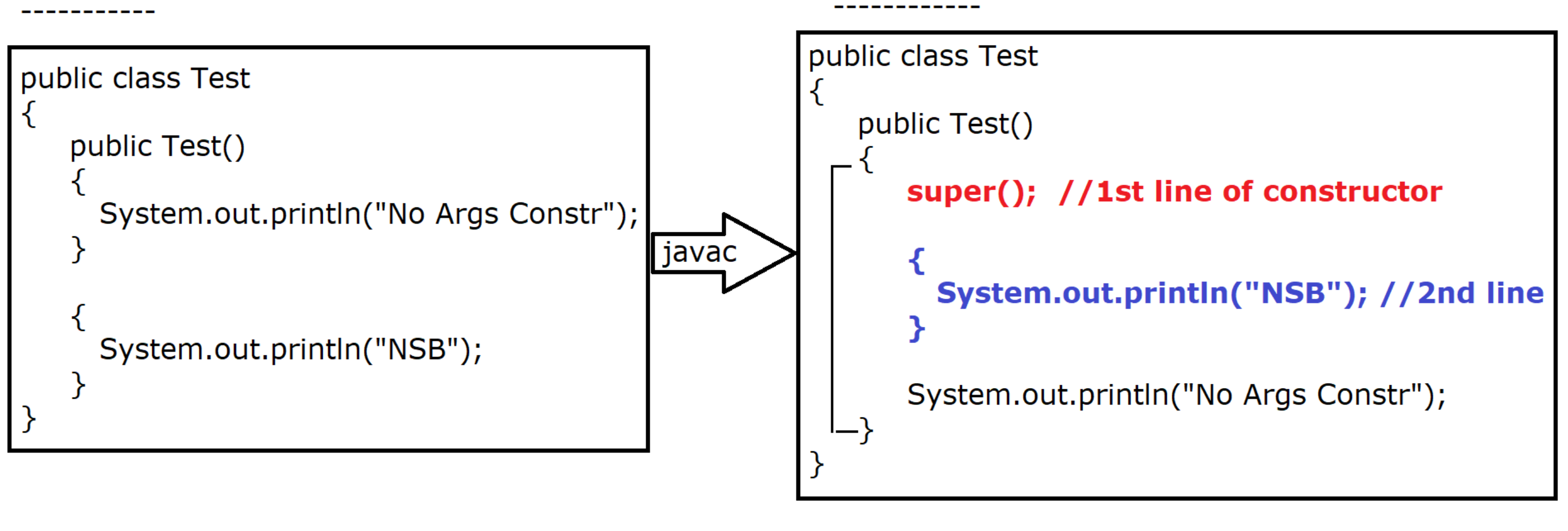
```java
package com.ravi.instance_block_demo;

class Test
{
   {
      System.out.println("Instance OR Non static block");
   }
}
public class InstanceBlockDemo1
{
   public static void main(String[] args)
   {
      System.out.println("Main Method");
      new Test();
      new Test();
   }

}
```

* A non static block is executed before the constructor body because java compiler will automatically place the non static block in the 2nd line of the constructor, IF AND ONLY IF, THE FIRST LINE OF CONSTRUCTOR CONTAINS super()

```java
package com.ravi.instance_block_demo;

class Demo
{
   public Demo()
   {
      System.out.println("No Argument Constructor!!!");
   }

   {
      System.out.println("Non static block");
   }
}
public class InstanceBlockDemo2 {

   public static void main(String[] args)
   {
      new Demo();
      System.out.println("...........");
      new Demo();

   }

}
```

Different cases to explain the above paragraph :

Case 1 :



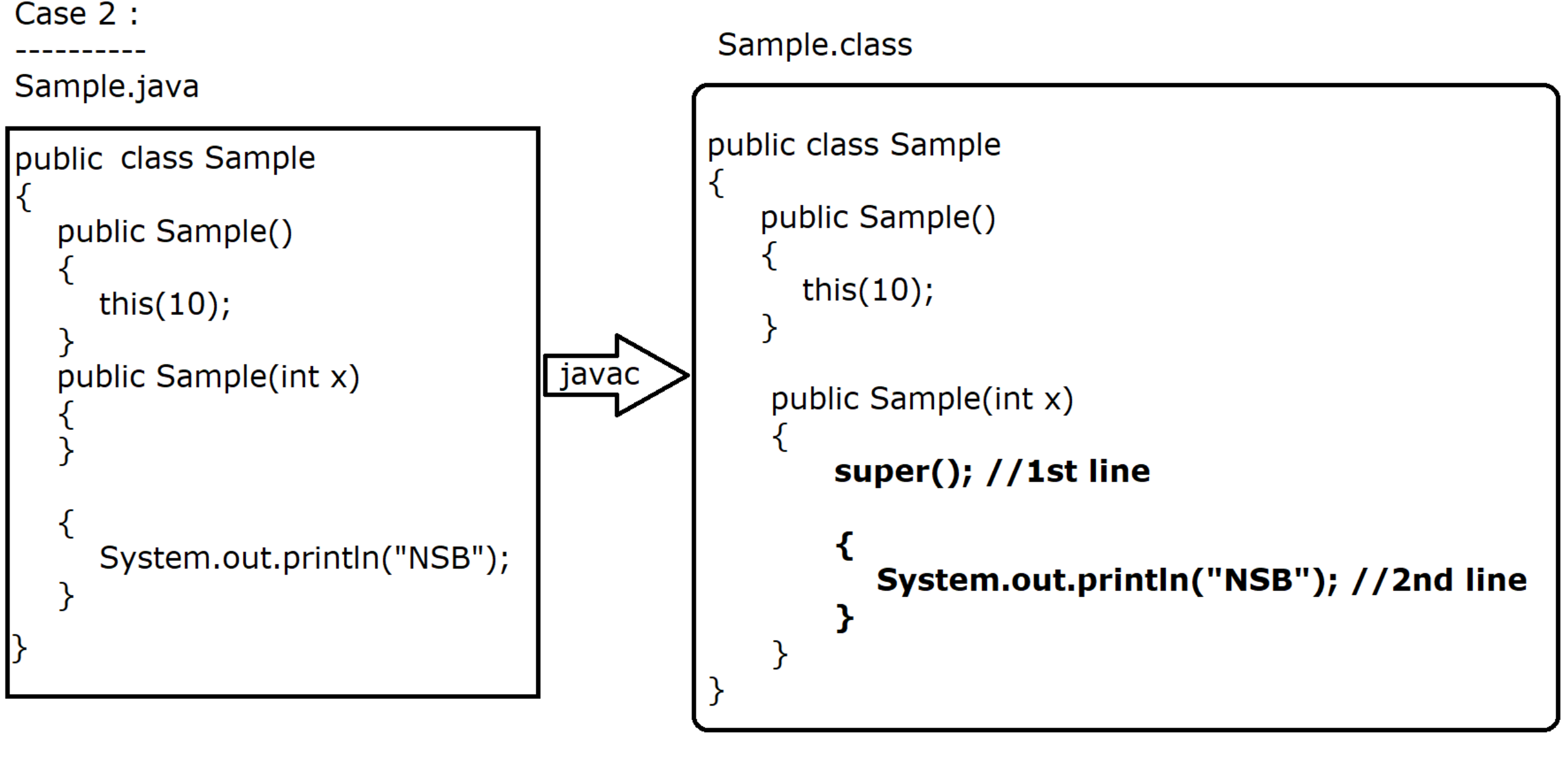
Test.java

```
public class Test
{
   public Test()
   {
     System.out.println("No Args Constr");
   }

   {
     System.out.println("NSB");
   }
}
```

Test.class

```
public class Test
{
   public Test()
   {
      super();  //1st line of constructor

      {
        System.out.println("NSB"); //2nd line
      }

      System.out.println("No Args Constr");
   }
}
```

Case 2 :



Sample.java

```
public  class Sample
{
   public Sample()
   {
      this(10);
   }
   public Sample(int x)
   {
   }

   {
      System.out.println("NSB");
   }
}
```

Sample.class

```
public class Sample
{
   public Sample()
   {
      this(10);
   }

   public Sample(int x)
   {
      super(); //1st line

      {
        System.out.println("NSB"); //2nd line
      }
   }
}
```

```java
package com.ravi.instance_block_demo;

class Addition
{
   public Addition()
   {
      this(100);
      System.out.println("No Argument Constructor!!!");
   }

   public Addition(int x)
   {
      super();
      System.out.println("Parameterized Constructor :"+x);
   }

   {
      System.out.println("Non static block");
   }
}

public class InstanceBlockDemo3
{
   public static void main(String[] args)
   {
      new Addition();

   }
}
```

Note : Non static block will be added to the parameterized constructor only

* The main purpose of non static block to initialize the non static variable so, It is also known as Instance initializer as well as it is also used to provide common message for all the Objects.

```java
package com.ravi.instance_block_demo;

class Sample
{
   int x;

   {
      x = 100;
      System.out.println("Object creation is in progress");
   }

   public Sample()
   {
      System.out.println("x value is :"+x);
   }
}
public class InstanceBlockDemo4 {

   public static void main(String[] args)
   {
      new Sample();
      System.out.println("...............");
      new Sample();

   }

}
```

* If we have multiple non static block then it would be executed according to the order. [top to bottom]

```java
package com.ravi.instance_block_demo;

class Foo
{
   int x;

   public Foo()
   {
      System.out.println("x value is :"+x);
   }

   {
      x = 100;
      System.out.println("x value is :"+x);
   }

   {
      x = 200;
      System.out.println("x value is :"+x);
   }

   {
      x = 300;
      System.out.println("x value is :"+x);
   }

}
public class InstanceBlockDemo5 {

   public static void main(String[] args)
   {
      new Foo();

   }

}
```

* Any initializer must be executed normally so we cannot write return statement inside non static block.

```java
package com.ravi.instance_block_demo;

public class InstanceBlockDemo6
{
   {
      return; //error
   }

   public static void main(String[] args)
   {

   }

}
```

* If we write non static block inside the constructor body then it will be executed as it is, Compiler will not perform any action.

```java
package com.ravi.instance_block_demo;

class Student
{
   public Student()
   {
      System.out.println("No Argument Constructor!!!");

      {
         System.out.println("Non static block");
      }
   }
}

public class InstanceBlockDemo7
{
   public static void main(String[] args)
   {
      new Student();
   }

}
```

```
Important Points :
------------------
1) The first line of any constructor is reserved for
   either super() OR this()

2) If user has not provided either super() OR this()
   then java compiler will automatically add super()
   to the first line of constructor

3) If a contsructor first line contains super() then
   2nd line is reserved for Non static block (If available)
------------------------------------------------------------
```