protected :
---
* It is an access modifier which is less restrictive than default. The members which are declared with protected access modifier can be accessible from same package as well as outside of the package but by using Inheritance concept.

```
class Test
{
   int x = 100;

   {
      x = 200;
   }

   public Test()
   {
      x = 300;
   }

   public void set(int x)
   {
      this.x = x;
   }

}

Test t1 = new Test();
```

1) new keyword
   x = 0

2) super()
   Object(){}

3) Instance Variable is initialized OR not at decl.

4) Instance variable is initialized with NSB or not

Note : Instance Variable declaration cum initialization and Non static block initialization, both are having same priority, Will be executed according to the order.

5) Instance Variable is initialized in the constructor body

6) Can initialize the NSV inside Method body also but not recommended.

JVM Architecture :
---
* The entire JVM Architecture is divided into 3 sections :

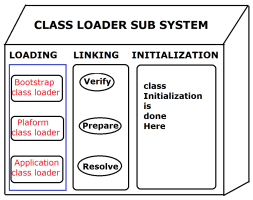
   1) Class Loader Sub System
   2) Runtime Data Areas (Memory Areas)
   3) Execution Engine

Class Loader Sub System :
---
* It is used to load the .class file into JVM memory so the execution of the program will be started.

* In order the load the required .class file into JVM memory, Class loader sub system, internally uses an algorithm i.e "**Delegation Hierarchy Algorithm**"

* Class Loader sub system internally performs the following 3 task

   a) LOADING
   b) LINKING
   c) INITIALIZATION



LOADING :

In order to load the required .class file, JVM makes a request to class loader sub system. The class loader sub system follows delegation hierarchy algorithm to load the required .class files from different areas.

To load the required .class from different area, we have 3 different kinds of class loaders.

1) Bootstrap/Primordial class Loader

2) Platform/Extension class Loader

3) Application/System class Loader

Bootstrap /Primordial class loader :
---
* It is responsible to load all the predifined .class files OR Java API (Application Programming interface) level .class file into JVM memory.

* When we install java software then java software people has provided a **jar file (java level zip file which contains number of .class files)** called **jrt-fs.jar**

* It loads the .class file from the following path :
   C:\\ Program files \ java \ JDK \ lib \ jrt-fs.jar

* It has the highest priority because it is the super class of all the class loaders

Platform/Extension class loader :
---
* It is used to load the required .class file into JVM which is given by any 3rd party in **jar** file format

* It loads the .class file from the following path
   C:\\Program files \java \ JDK \ jre \ lib \ ext \ 3rdParty.jar

* It is the super class of Application class loader so it has highest priority than Application class loader but lower priority than Bootstrap class loader.