# Core JAVA training - index

## Date:05/08/2024

1. Language And Applications
2. JAVA Features
    - Why Java is Independent?
    - Oops
    - Exception Handling
    - Multi threading
    - Web Application
    - Open Source
    - Security
    - Support  Networking
    - Memory Management
3. JDK,JRE,JVM
4. Basic Java Programming
5. Packages

## Date:06/08/2024
## Morning(11:00 am)

1. Nested Loops
2. One Dimensional Array
3. Two Dimensional Array
4. Logical Programming

## After Noon(3:30 pm)

1. SwitchCase
2. Scanner Class
3. Java.lang
    - Object Class Methods
4. Enum
5. Event Management Application

**Date:07/08/2024**
**Mrng(11:00 am)**

1.oops

- Encapsulation
  Programs
  Calculation
  Person
  Method Flow

**After Noon(3:30 pm)**

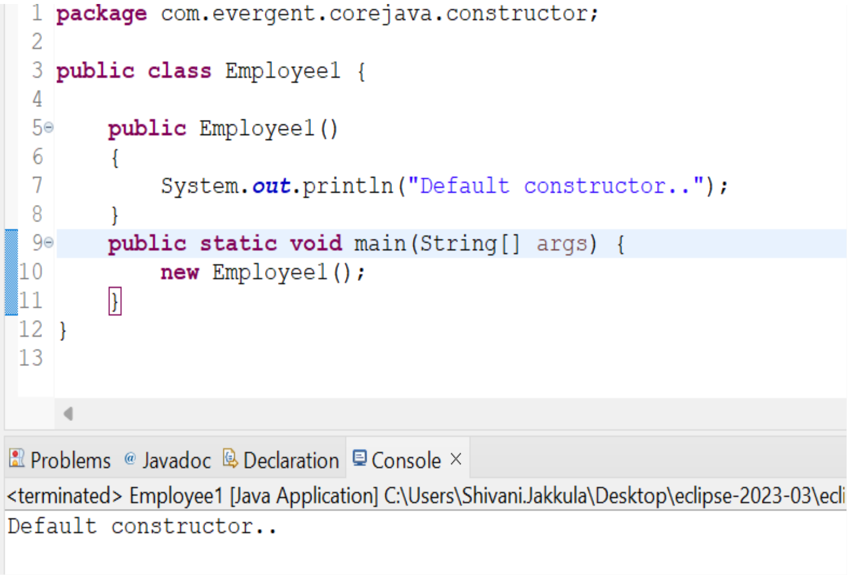
1. Inheritance
2. Polymorphism
   .Method overloading
   .Method Overriding
3. Abstraction
4. IS-A (Inheritance)
5. HAs- A (Object Creation).

**Date:08/08/24**

## Constructor

i. Class name and constructor name should be same
ii. There are 2 types of constructors
  a. Default Constructor
  b. Parameterized Constructor
iii. We can access constructor while creation of object
iv. Constructors are mainly for initializing
v. Constructor doesn’t have any return type not even void. If you declare as a void the compiler will consider as a method not a constructor
vi. Every class needs atleast 1 default constructor
vii. this, super This
  --> this is a keyword always refers to instance variables
viii. Always constructor are overloaded

## Program1:

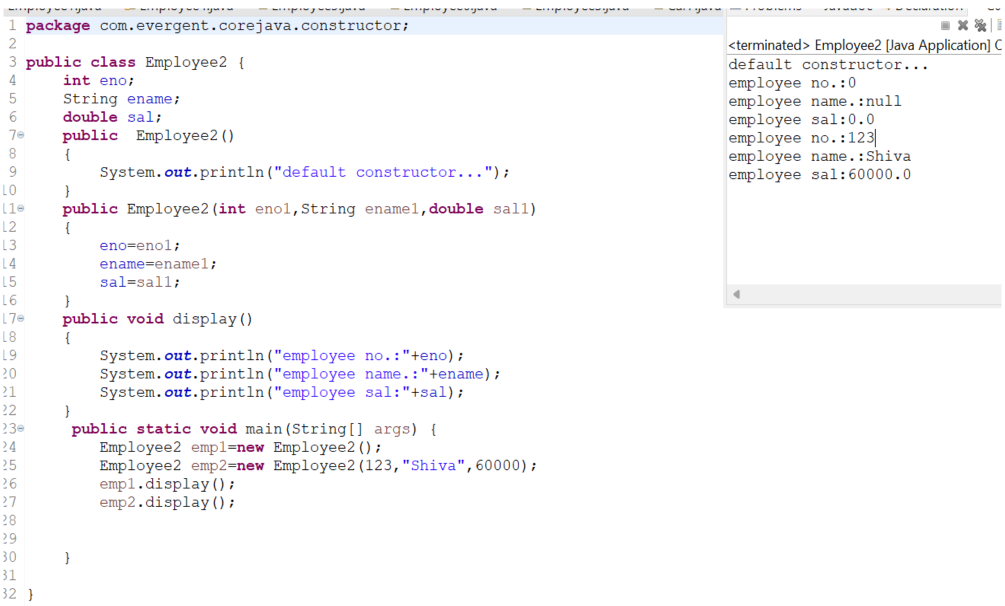


```java
package com.evergent.corejava.constructor;

public class Employee1 {

    public Employee1()
    {
        System.out.println("Default constructor..");
    }
    public static void main(String[] args) {
        new Employee1();
    }
}

```

<terminated> Employee1 [Java Application] C:\Users\Shivani.Jakkula\Desktop\eclipse-2023-03\ecli

```
Default constructor..
```

## Program2:

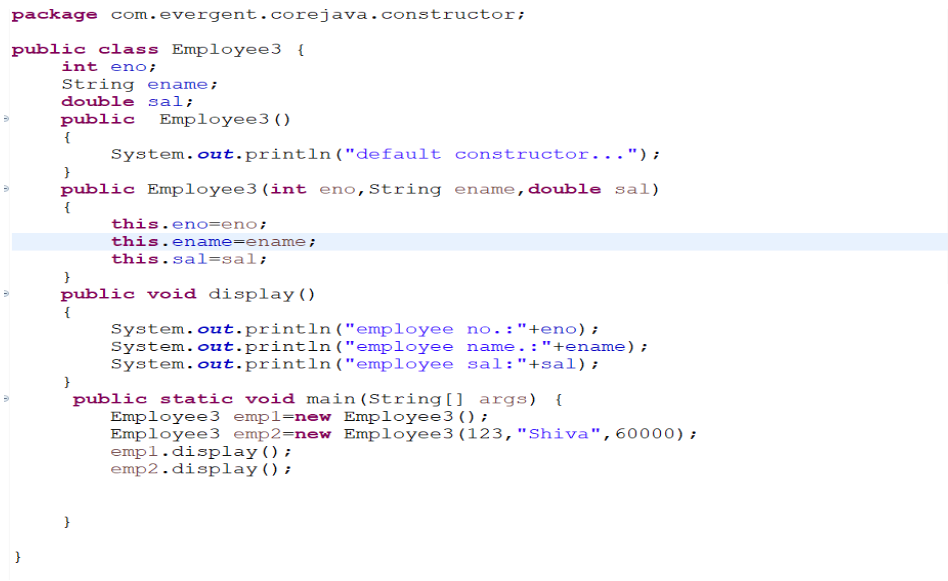


```java
package com.evergent.corejava.constructor;

public class Employee2 {
    int eno;
    String ename;
    double sal;
    public  Employee2()
    {
        System.out.println("default constructor...");
    }
    public Employee2(int eno1,String ename1,double sal1)
    {
        eno=eno1;
        ename=ename1;
        sal=sal1;
    }
    public void display()
    {
        System.out.println("employee no.:"+eno);
        System.out.println("employee name.:"+ename);
        System.out.println("employee sal:"+sal);
    }
     public static void main(String[] args) {
        Employee2 emp1=new Employee2();
        Employee2 emp2=new Employee2(123,"Shiva",60000);
        emp1.display();
        emp2.display();


    }

}
```

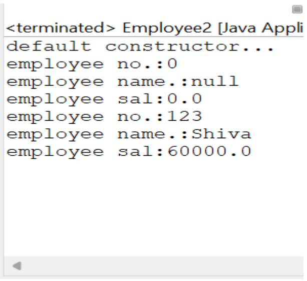
<terminated> Employee2 [Java Application] C

```
default constructor...
employee no.:0
employee name.:null
employee sal:0.0
employee no.:123
employee name.:Shiva
employee sal:60000.0
```

## Program3:



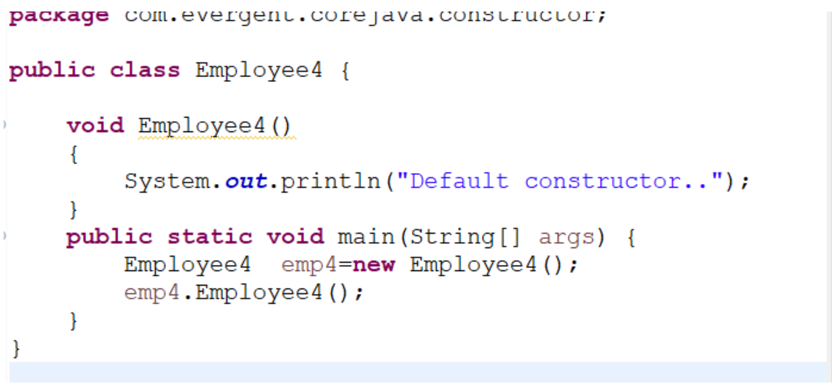
```java
package com.evergent.corejava.constructor;

public class Employee3 {
    int eno;
    String ename;
    double sal;
    public  Employee3()
    {
        System.out.println("default constructor...");
    }
    public Employee3(int eno,String ename,double sal)
    {
        this.eno=eno;
        this.ename=ename;
        this.sal=sal;
    }
    public void display()
    {
        System.out.println("employee no.:"+eno);
        System.out.println("employee name.:"+ename);
        System.out.println("employee sal:"+sal);
    }
     public static void main(String[] args) {
        Employee3 emp1=new Employee3();
        Employee3 emp2=new Employee3(123,"Shiva",60000);
        emp1.display();
        emp2.display();


    }

}
```



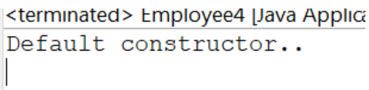
<terminated> Employee2 [Java Appli

```
default constructor...
employee no.:0
employee name.:null
employee sal:0.0
employee no.:123
employee name.:Shiva
employee sal:60000.0
```

## Program4:



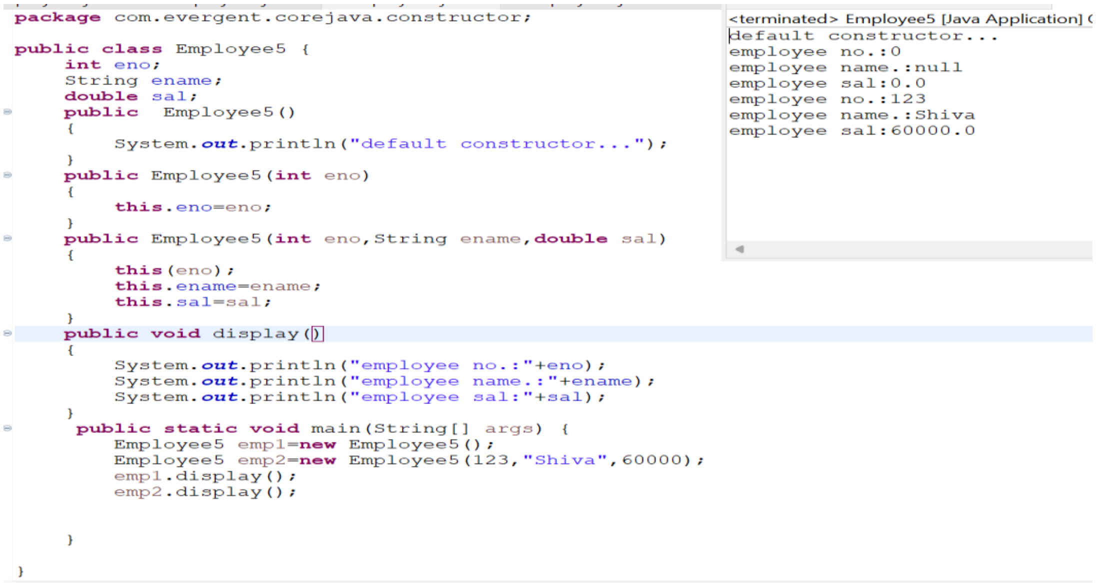
```java
package com.evergent.corejava.constructor;

public class Employee4 {

    void Employee4()
    {
        System.out.println("Default constructor..");
    }
    public static void main(String[] args) {
        Employee4  emp4=new Employee4();
        emp4.Employee4();
    }
}
```



<terminated> Employee4 [Java Applic

```
Default constructor..
```

## Program5:

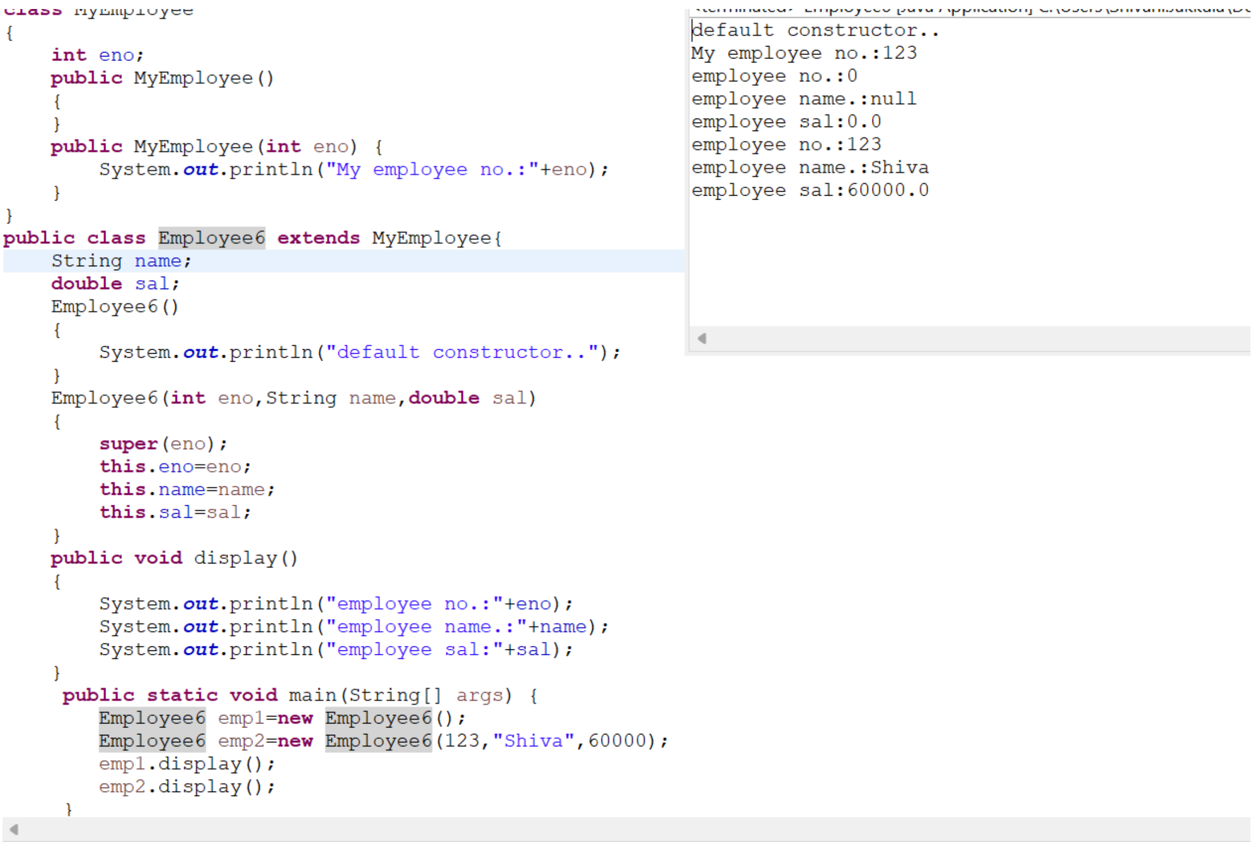


```
package com.evergent.corejava.constructor;

public class Employee5 {
    int eno;
    String ename;
    double sal;
    public  Employee5()
    {
        System.out.println("default constructor...");
    }
    public Employee5(int eno)
    {
        this.eno=eno;
    }
    public Employee5(int eno,String ename,double sal)
    {
        this(eno);
        this.ename=ename;
        this.sal=sal;
    }
    public void display()
    {
        System.out.println("employee no.:"+eno);
        System.out.println("employee name.:"+ename);
        System.out.println("employee sal:"+sal);
    }
     public static void main(String[] args) {
        Employee5 emp1=new Employee5();
        Employee5 emp2=new Employee5(123,"Shiva",60000);
        emp1.display();
        emp2.display();


    }
}
```

<terminated> Employee5 [Java Application] (

```
default constructor...
employee no.:0
employee name.:null
employee sal:0.0
employee no.:123
employee name.:Shiva
employee sal:60000.0
```

## Program 6:

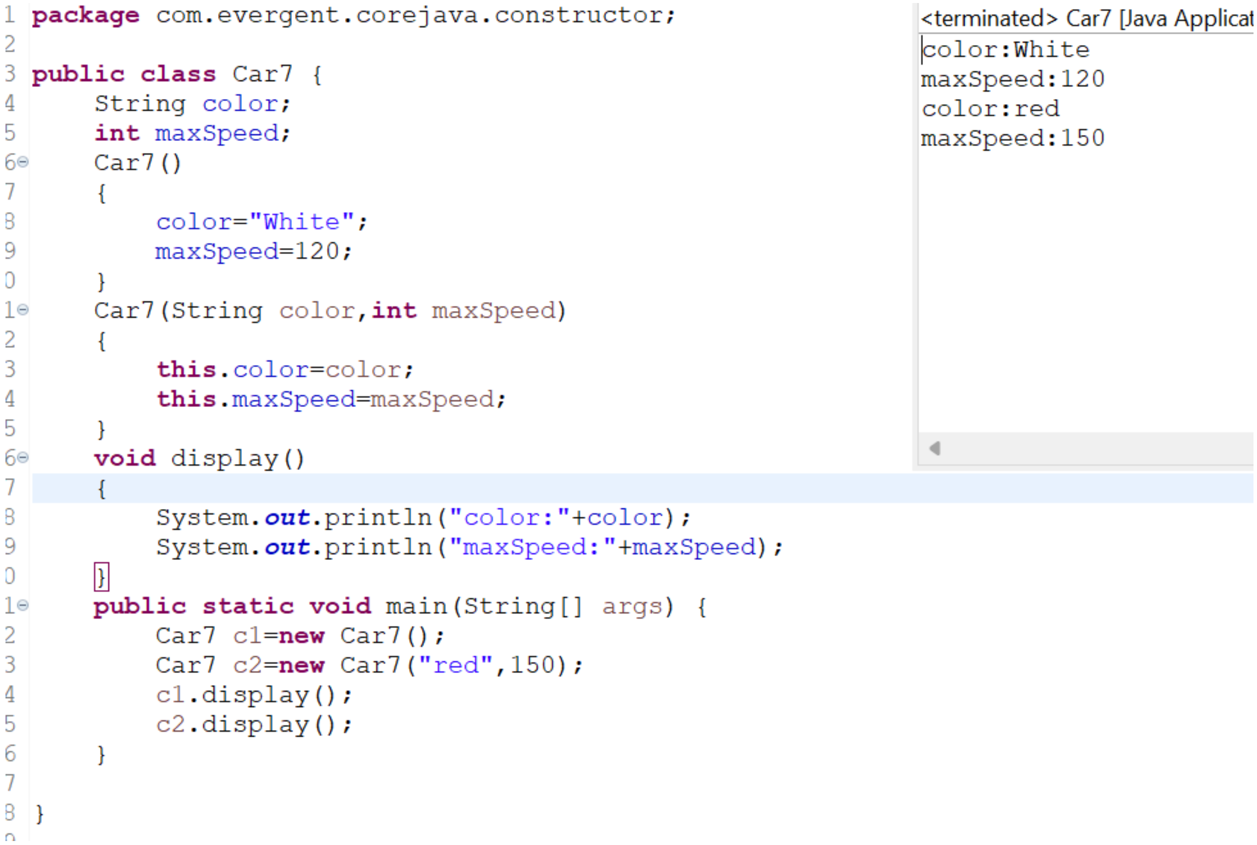


```
class MyEmployee
{
    int eno;
    public MyEmployee()
    {
    }
    public MyEmployee(int eno)  {
        System.out.println("My employee no.:"+eno);
    }
}
public class Employee6 extends MyEmployee{
    String name;
    double sal;
    Employee6()
    {
        System.out.println("default constructor..");
    }
    Employee6(int eno,String name,double sal)
    {
        super(eno);
        this.eno=eno;
        this.name=name;
        this.sal=sal;
    }
    public void display()
    {
        System.out.println("employee no.:"+eno);
        System.out.println("employee name.:"+name);
        System.out.println("employee sal:"+sal);
    }
     public static void main(String[] args)  {
        Employee6 emp1=new Employee6();
        Employee6 emp2=new Employee6(123,"Shiva",60000);
        emp1.display();
        emp2.display();
     }
```

<terminated> Employee6 [Java Application] C:\Users\Shivanisukkala\De

```
default constructor..
My employee no.:123
employee no.:0
employee name.:null
employee sal:0.0
employee no.:123
employee name.:Shiva
employee sal:60000.0
```

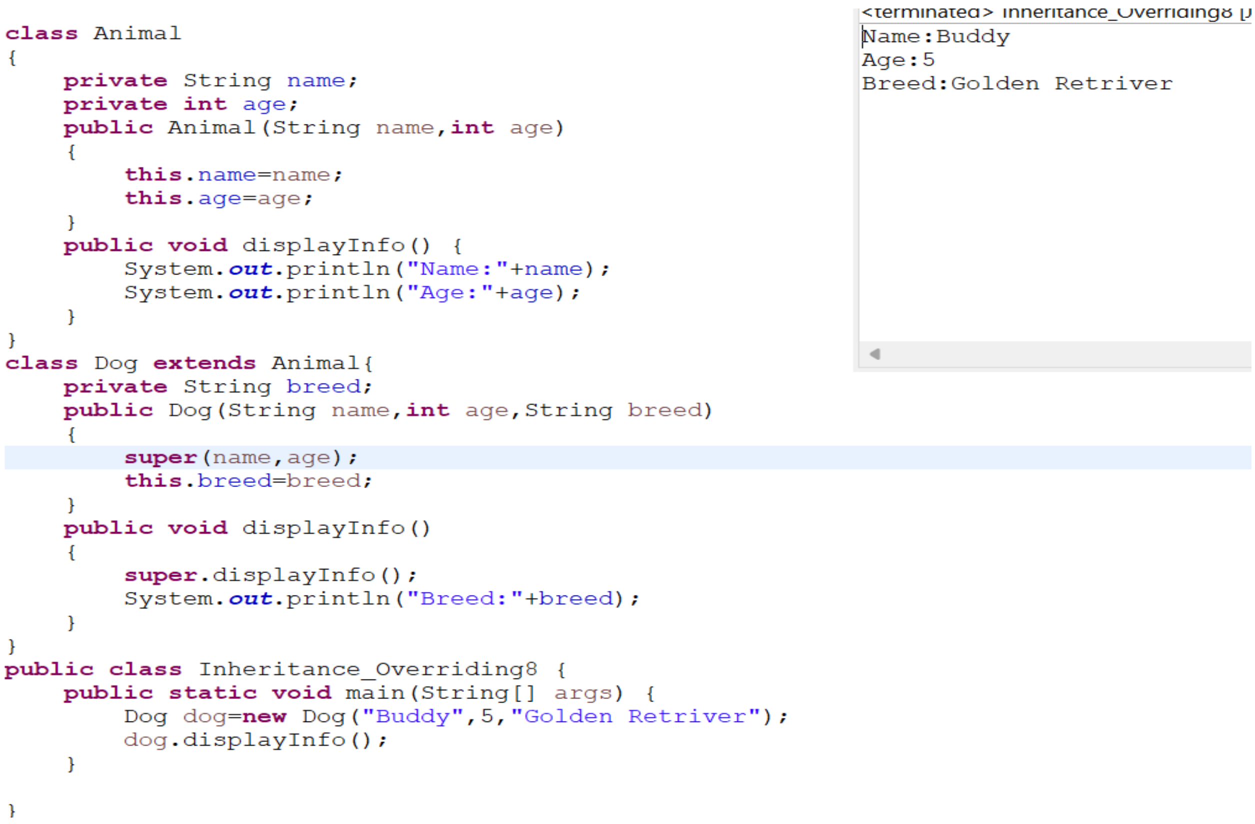
## Program 7:



```java
package com.evergent.corejava.constructor;

public class Car7 {
    String color;
    int maxSpeed;
    Car7()
    {
        color="White";
        maxSpeed=120;
    }
    Car7(String color,int maxSpeed)
    {
        this.color=color;
        this.maxSpeed=maxSpeed;
    }
    void display()
    {
        System.out.println("color:"+color);
        System.out.println("maxSpeed:"+maxSpeed);
    }
    public static void main(String[] args) {
        Car7 c1=new Car7();
        Car7 c2=new Car7("red",150);
        c1.display();
        c2.display();
    }

}
```

```
color:White
maxSpeed:120
color:red
maxSpeed:150
```

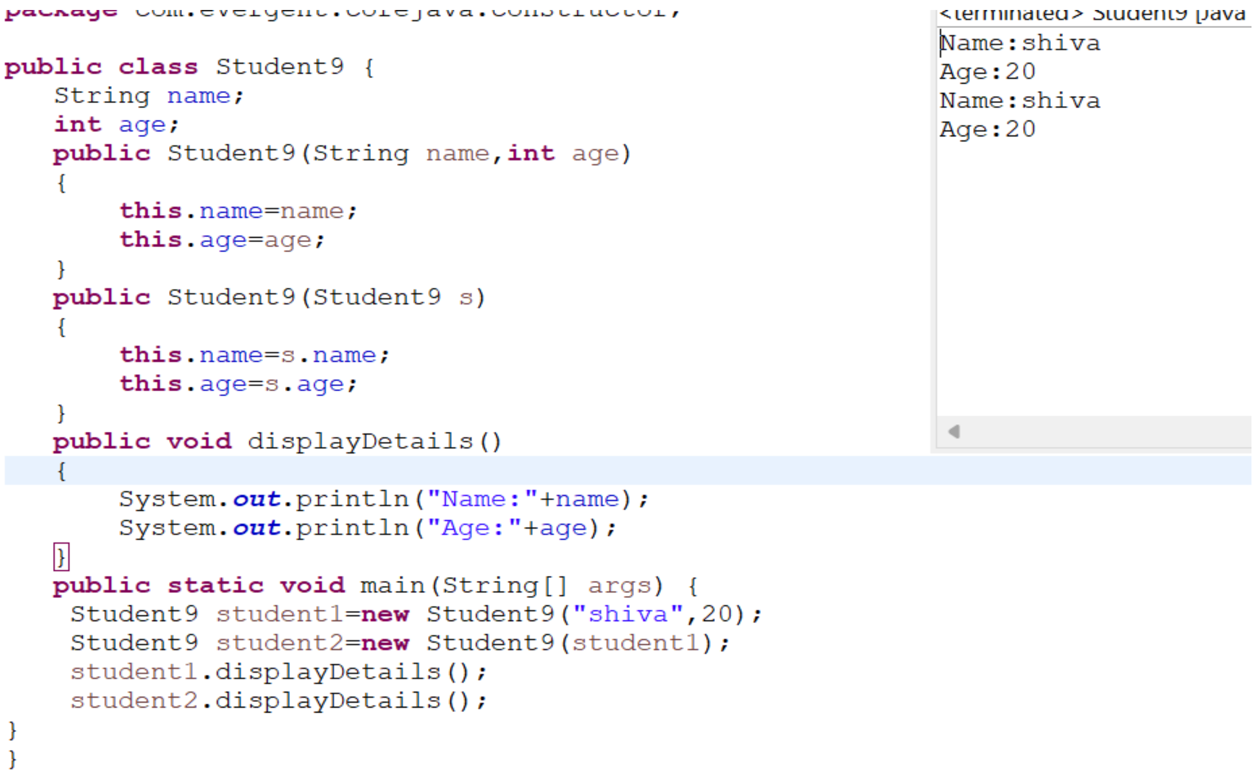
## Program8:



```
<terminated> Inheritance_Overriding8 [
Name:Buddy
Age:5
Breed:Golden Retriver
```

```
class Animal
{
    private String name;
    private int age;
    public Animal(String name,int age)
    {
        this.name=name;
        this.age=age;
    }
    public void displayInfo() {
        System.out.println("Name:"+name);
        System.out.println("Age:"+age);
    }
}
class Dog extends Animal{
    private String breed;
    public Dog(String name,int age,String breed)
    {
        super(name,age);
        this.breed=breed;
    }
    public void displayInfo()
    {
        super.displayInfo();
        System.out.println("Breed:"+breed);
    }
}
public class Inheritance_Overriding8 {
    public static void main(String[] args) {
        Dog dog=new Dog("Buddy",5,"Golden Retriver");
        dog.displayInfo();
    }

}
```

## Program9:



```java
package com.evergent.corejava.constructor;

public class Student9 {
   String name;
   int age;
   public Student9(String name,int age)
   {
       this.name=name;
       this.age=age;
   }
   public Student9(Student9 s)
   {
       this.name=s.name;
       this.age=s.age;
   }
   public void displayDetails()
   {
       System.out.println("Name:"+name);
       System.out.println("Age:"+age);
   }
   public static void main(String[] args) {
    Student9 student1=new Student9("shiva",20);
    Student9 student2=new Student9(student1);
    student1.displayDetails();
    student2.displayDetails();
}
}
```

<terminated> Student9 [Java

```
Name:shiva
Age:20
Name:shiva
Age:20
```

## Date:09/08/2024 - Day5

### 1. Static

a. Static is a keyword

b. We can declare variables and methods as static

c. We can access static variables and static methods directly through calssname.methodname and classname.variablename respectively.

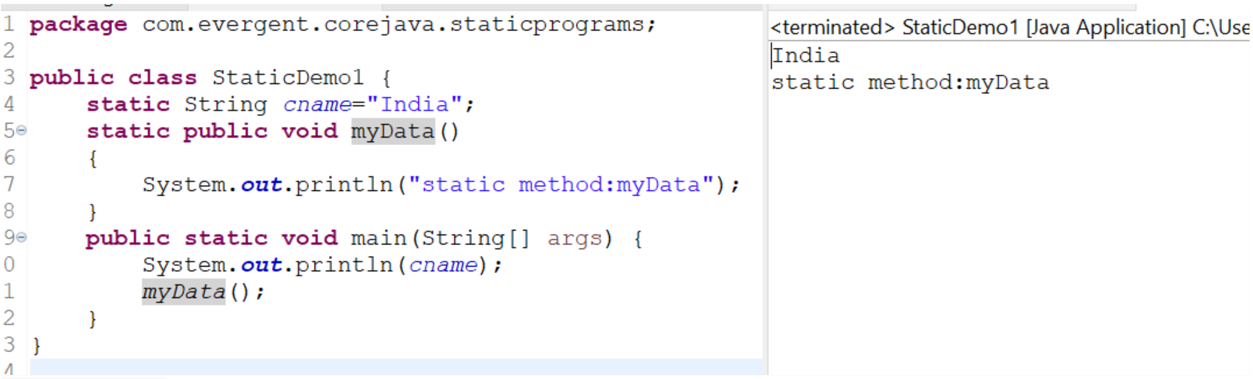
d. Static methods can access static methods and static variables only.

e. Static methods cannot access non static methods and non static variables.

f. Non static methods can access static methods and static variables.

g. Static block- whenever class is loaded inside the JVM at that time static block is initiated.

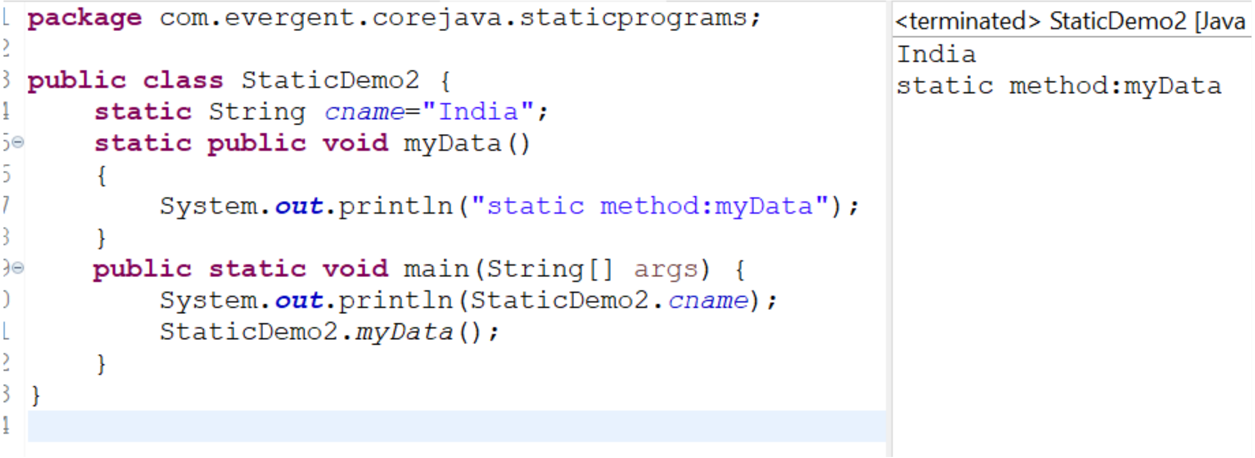
**Program1:**



```java
package com.evergent.corejava.staticprograms;

public class StaticDemo1 {
    static String cname="India";
    static public void myData()
    {
        System.out.println("static method:myData");
    }
    public static void main(String[] args) {
        System.out.println(cname);
        myData();
    }
}
```

```
<terminated> StaticDemo1 [Java Application] C:\Use
India
static method:myData
```

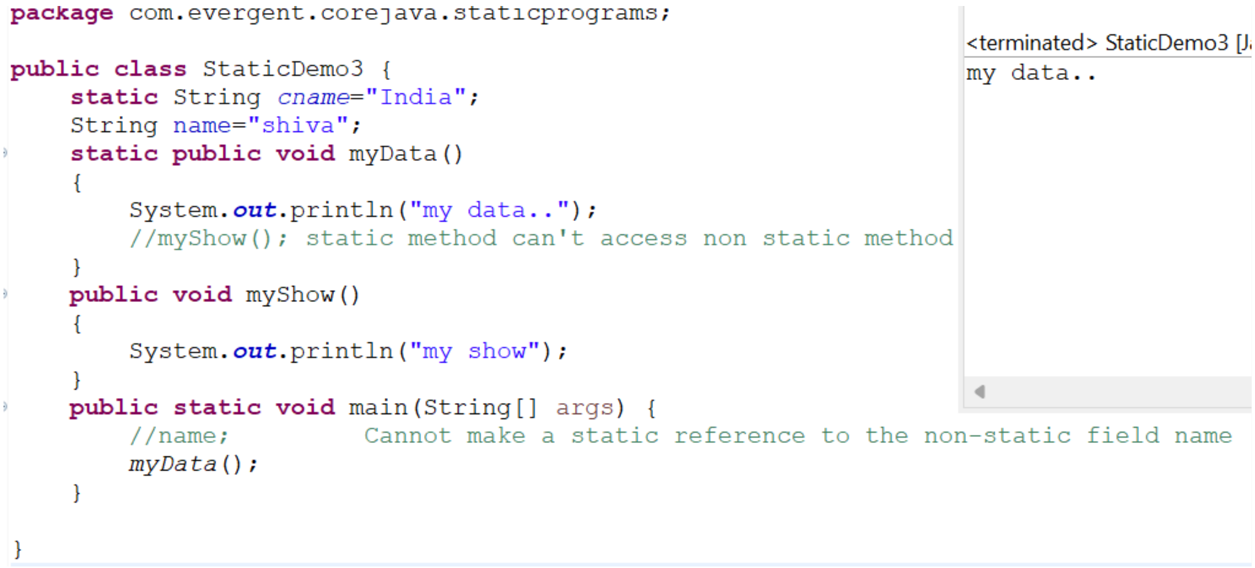
**Program2:**



```java
package com.evergent.corejava.staticprograms;

public class StaticDemo2 {
    static String cname="India";
    static public void myData()
    {
        System.out.println("static method:myData");
    }
    public static void main(String[] args) {
        System.out.println(StaticDemo2.cname);
        StaticDemo2.myData();
    }
}
```

```
<terminated> StaticDemo2 [Java
India
static method:myData
```

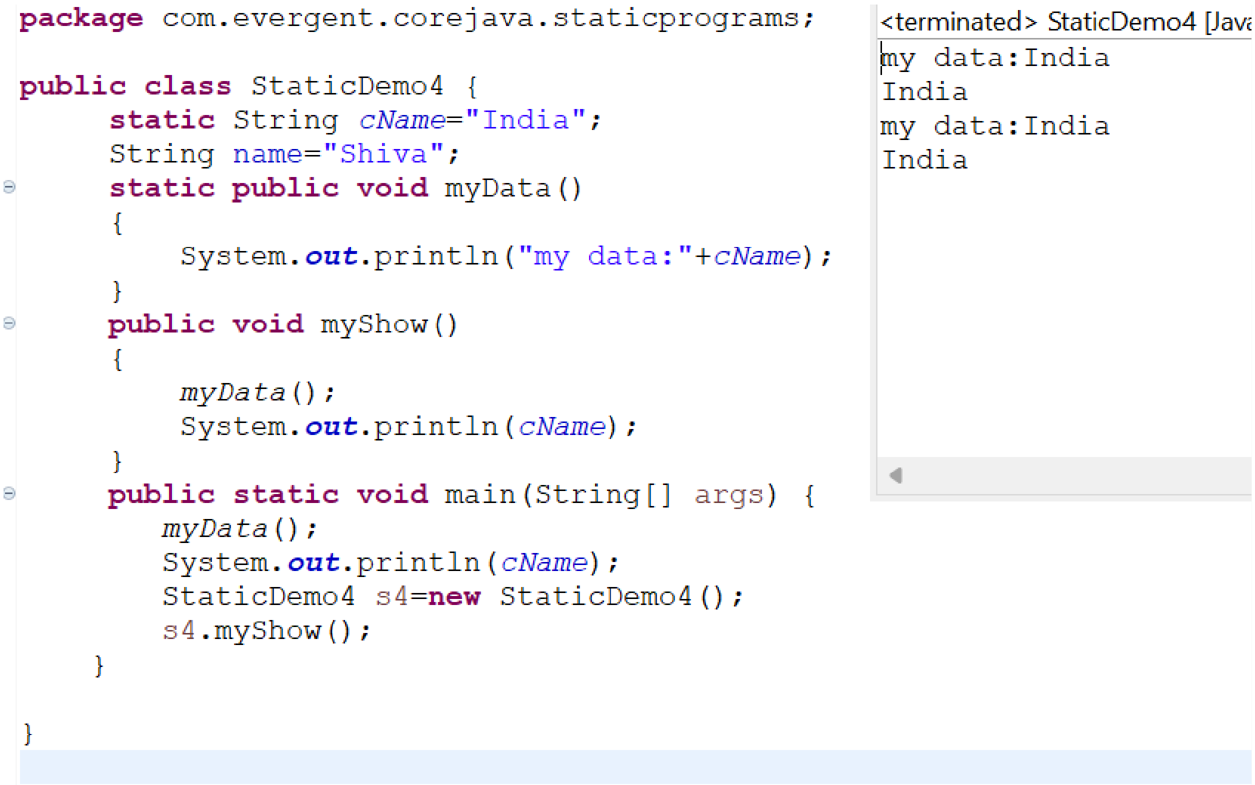
**Program3:**



```
package com.evergent.corejava.staticprograms;

public class StaticDemo3 {
    static String cname="India";
    String name="shiva";
    static public void myData()
    {
        System.out.println("my data..");
        //myShow(); static method can't access non static method
    }
    public void myShow()
    {
        System.out.println("my show");
    }
    public static void main(String[] args) {
        //name;         Cannot make a static reference to the non-static field name
        myData();
    }

}
```

```
<terminated> StaticDemo3 [J
my data..
```

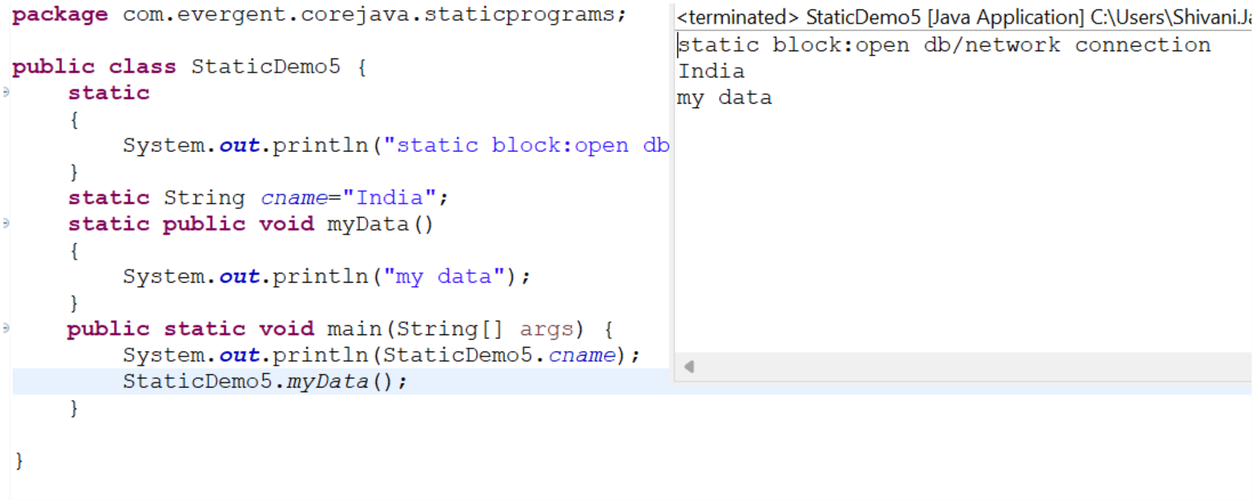
## Program4:



```
package com.evergent.corejava.staticprograms;

public class StaticDemo4 {
      static String cName="India";
      String name="Shiva";
      static public void myData()
      {
           System.out.println("my data:"+cName);
      }
      public void myShow()
      {
           myData();
           System.out.println(cName);
      }
      public static void main(String[] args) {
          myData();
          System.out.println(cName);
          StaticDemo4 s4=new StaticDemo4();
          s4.myShow();
     }

}
```

```
<terminated> StaticDemo4 [Java
my data:India
India
my data:India
India
```

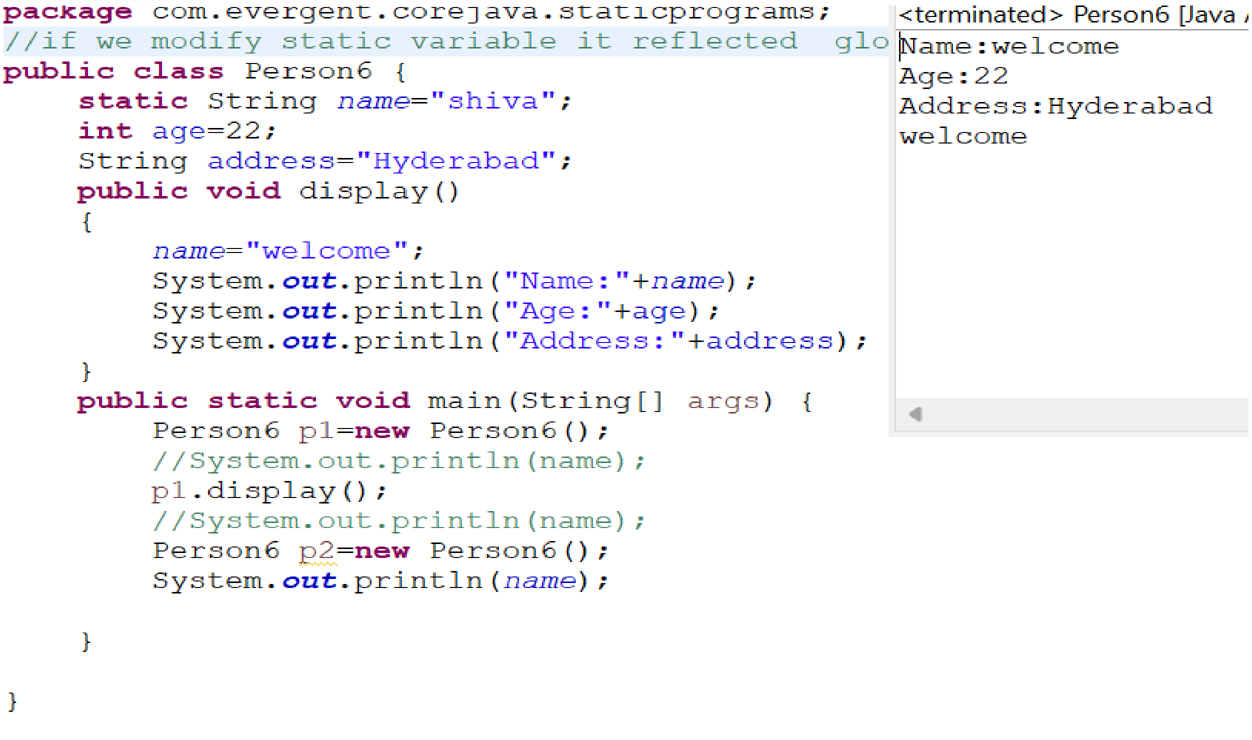
## Program5:



```
package com.evergent.corejava.staticprograms;

public class StaticDemo5 {
    static
    {
        System.out.println("static block:open db
    }
    static String cname="India";
    static public void myData()
    {
        System.out.println("my data");
    }
    public static void main(String[] args) {
        System.out.println(StaticDemo5.cname);
        StaticDemo5.myData();
    }

}
```

```
<terminated> StaticDemo5 [Java Application] C:\Users\Shivani.Ja
static block:open db/network connection
India
my data
```

## Program6:



```
package com.evergent.corejava.staticprograms;
//if we modify static variable it reflected  glo
public class Person6 {
    static String name="shiva";
    int age=22;
    String address="Hyderabad";
    public void display()
    {
        name="welcome";
        System.out.println("Name:"+name);
        System.out.println("Age:"+age);
        System.out.println("Address:"+address);
    }
    public static void main(String[] args)  {
        Person6 p1=new Person6();
        //System.out.println(name);
        p1.display();
        //System.out.println(name);
        Person6 p2=new Person6();
        System.out.println(name);

    }

}
```

```
<terminated> Person6 [Java
Name:welcome
Age:22
Address:Hyderabad
welcome
```

## 2. Final

a. Final is a Keyword.

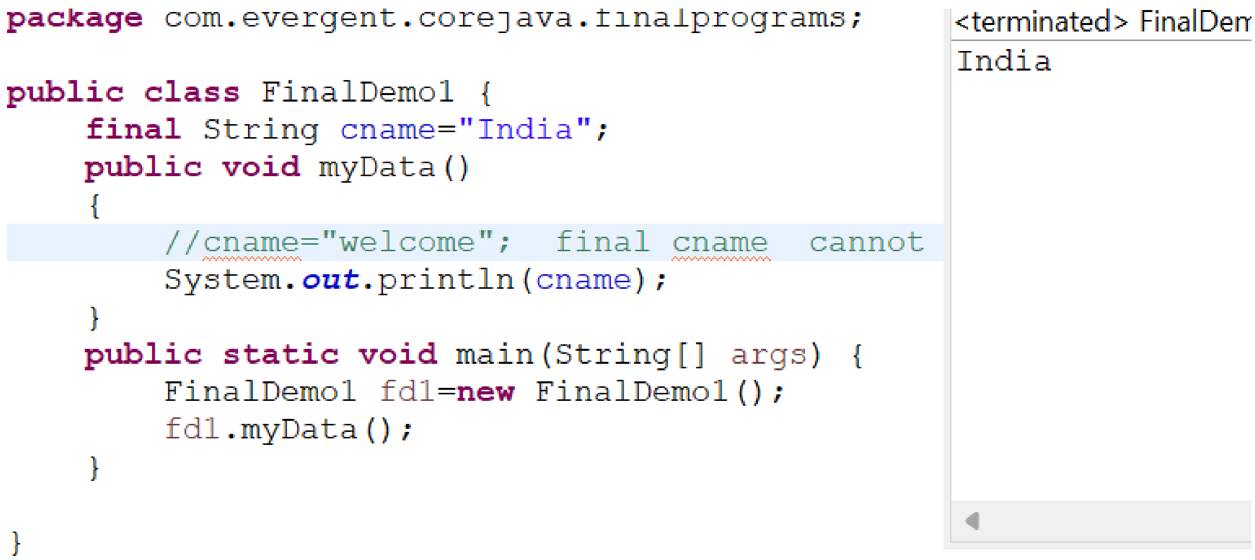
b. We can declare a variable, method, or a class as final.

c. Final variable cannot be modified.

d. Final Method cannot be overrided.

e. Final class cannot be inherited.

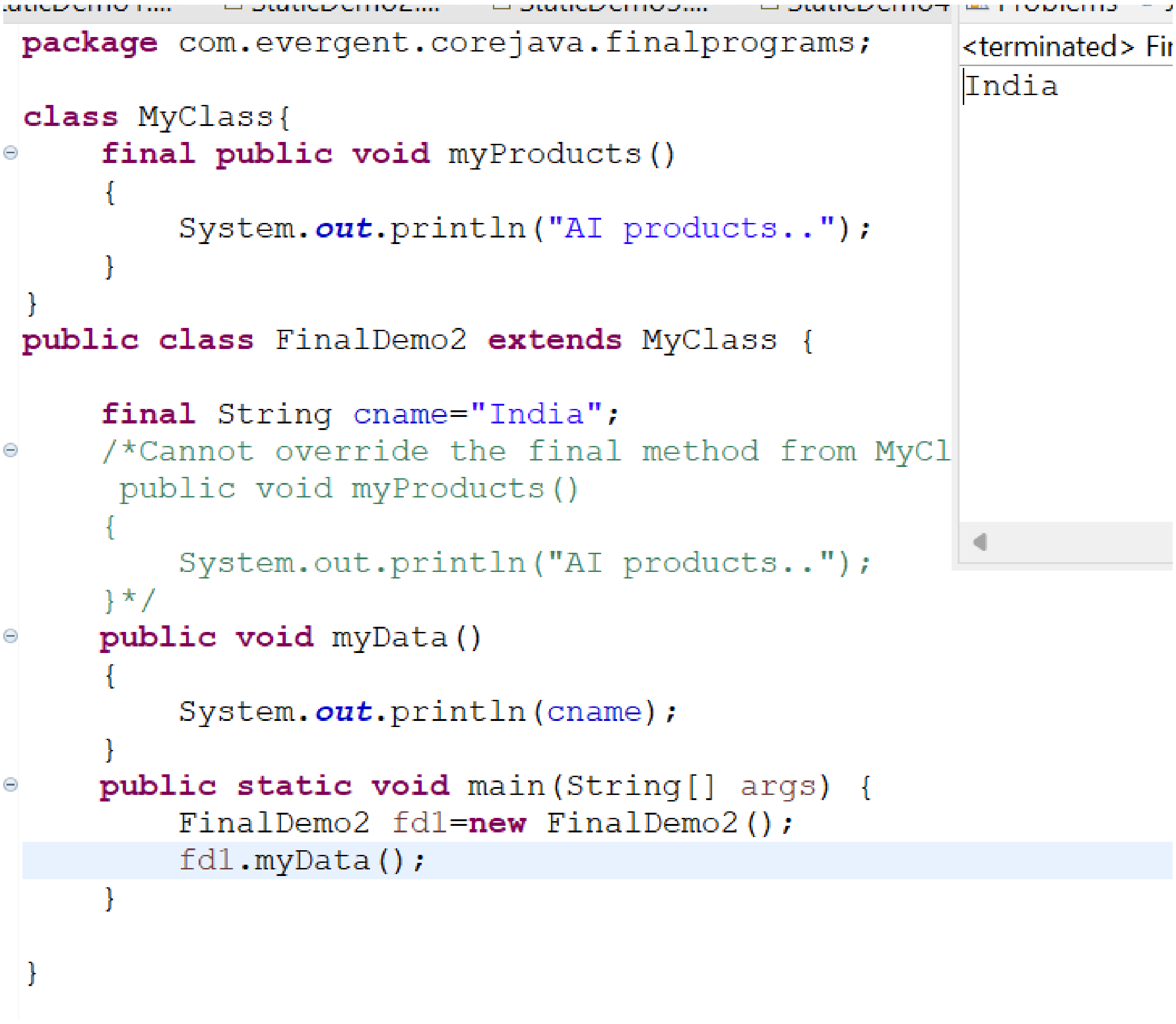
**Program1:**



```
package com.evergent.corejava.finalprograms;

public class FinalDemo1 {
    final String cname="India";
    public void myData()
    {
        //cname="welcome";  final cname  cannot
        System.out.println(cname);
    }
    public static void main(String[] args) {
        FinalDemo1 fd1=new FinalDemo1();
        fd1.myData();
    }

}
```

```
India
```

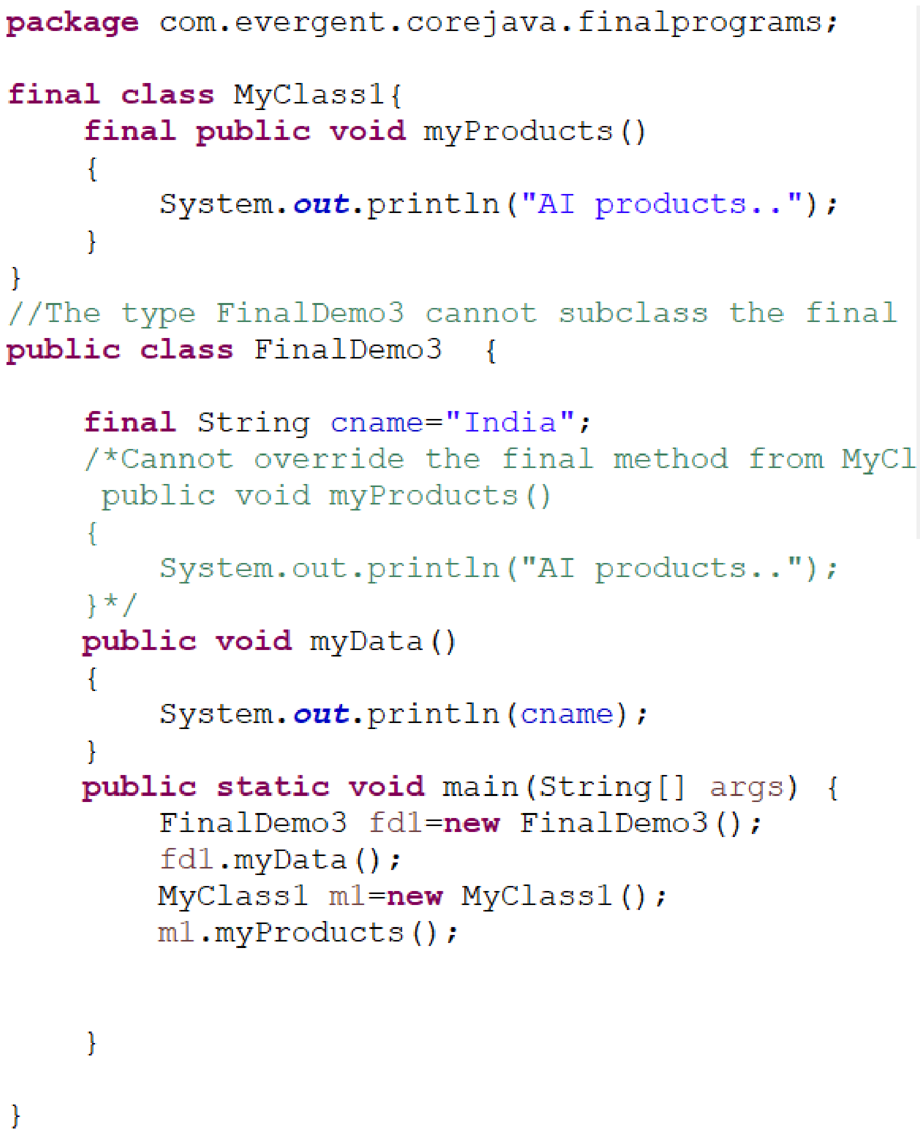
**Program2:**



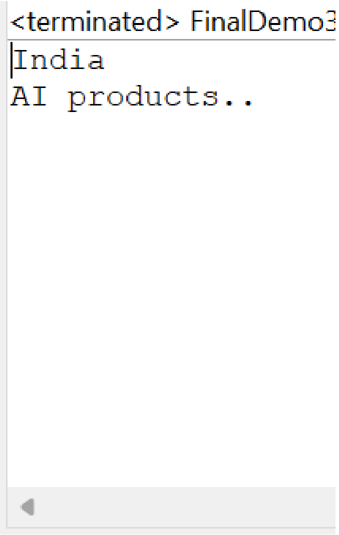
```
package com.evergent.corejava.finalprograms;

class MyClass{
    final public void myProducts()
    {
        System.out.println("AI products..");
    }
}
public class FinalDemo2 extends MyClass {

    final String cname="India";
    /*Cannot override the final method from MyCl
     public void myProducts()
    {
        System.out.println("AI products..");
    }*/
    public void myData()
    {
        System.out.println(cname);
    }
    public static void main(String[] args) {
        FinalDemo2 fd1=new FinalDemo2();
        fd1.myData();
    }

}
```

```
India
```

**Program3:**



```
package com.evergent.corejava.finalprograms;

final class MyClass1{
    final public void myProducts()
    {
        System.out.println("AI products..");
    }
}
//The type FinalDemo3 cannot subclass the final
public class FinalDemo3  {

    final String cname="India";
    /*Cannot override the final method from MyCl
     public void myProducts()
    {
        System.out.println("AI products..");
    }*/
    public void myData()
    {
        System.out.println(cname);
    }
    public static void main(String[] args) {
        FinalDemo3 fd1=new FinalDemo3();
        fd1.myData();
        MyClass1 m1=new MyClass1();
        m1.myProducts();


    }

}
```



<terminated> FinalDemo3

```
India
AI products..
```

**12/08/24 :**

**Strings:**

**-Why string is immutable?**

a. String is a final class

b. Strings are immutable

c. Strings having methods

d. All methods are non-synchronized

**StringBuffer:**

a. String buffer is a final class

b. String buffer is mutable

c. String buffer having methods

**d.** All methods are synchronized

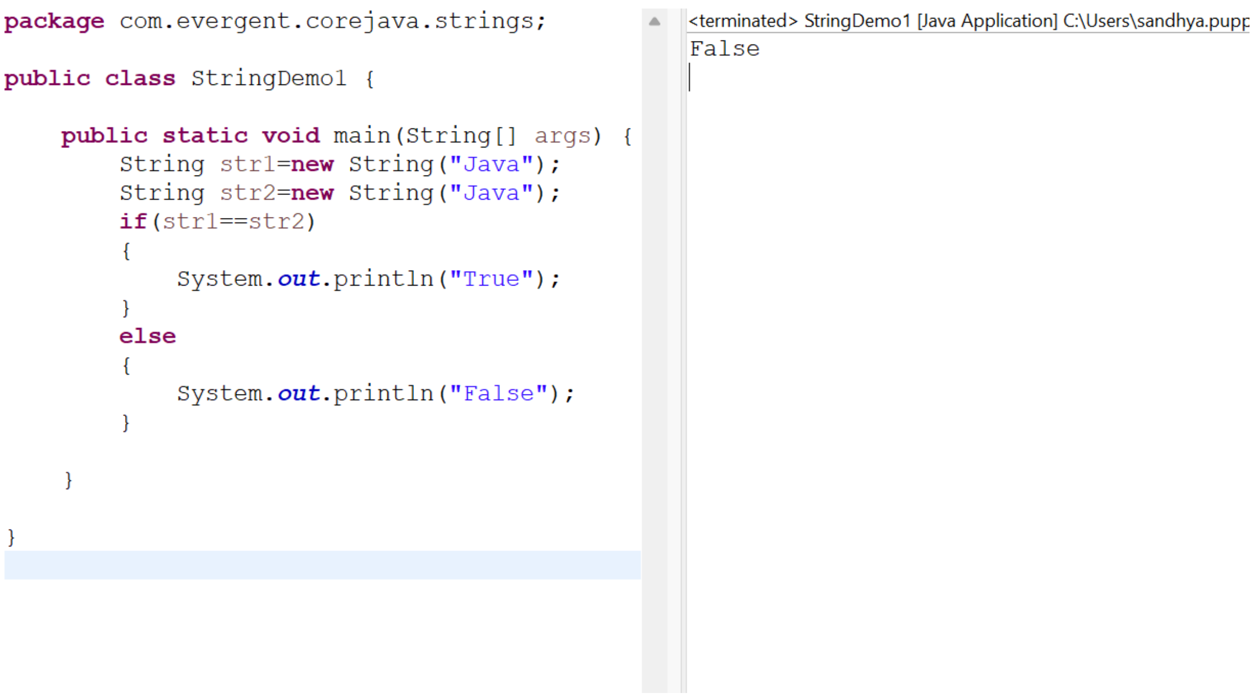
**String Builder:**

a. String builder is a final class

b. String builder is mutable

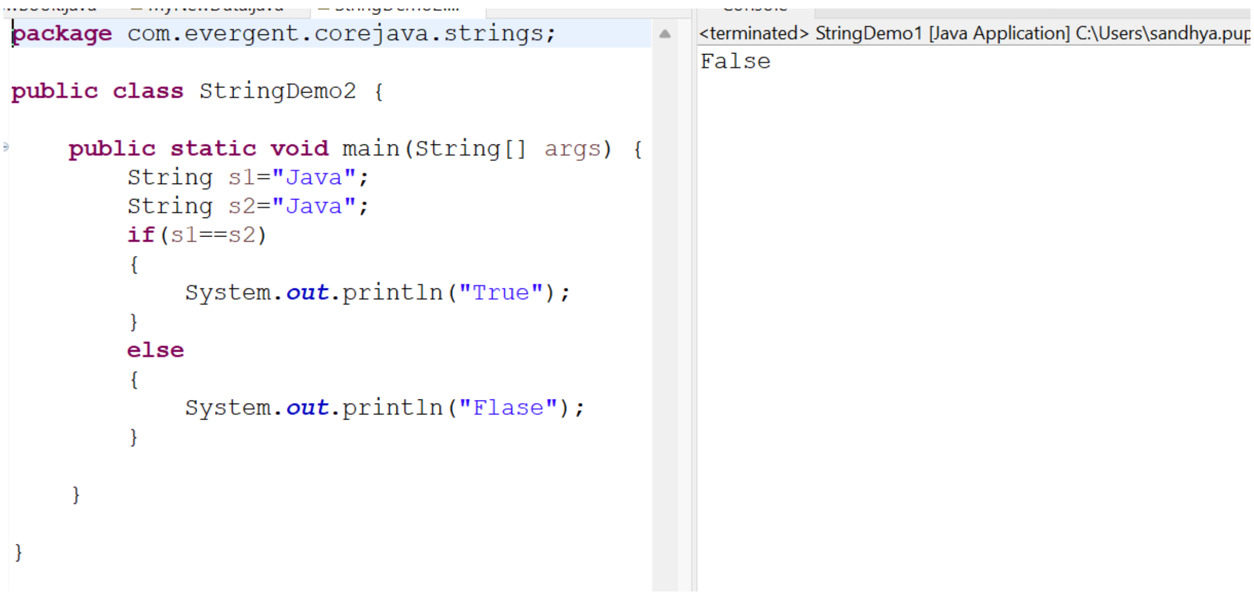
c. String builder having methods

d. All methods are non-synchronized



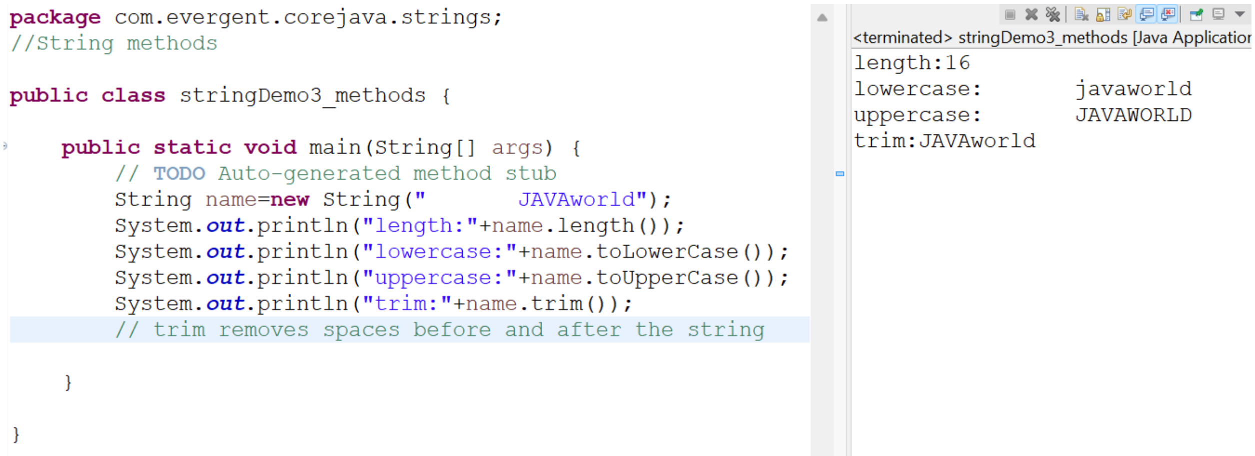
```java
package com.evergent.corejava.strings;

public class StringDemo1 {

    public static void main(String[] args) {
        String str1=new String("Java");
        String str2=new String("Java");
        if(str1==str2)
        {
            System.out.println("True");
        }
        else
        {
            System.out.println("False");
        }

    }

}
```

```
<terminated> StringDemo1 [Java Application] C:\Users\sandhya.pupp
False
```



```
package com.evergent.corejava.strings;

public class StringDemo2 {

    public static void main(String[] args) {
        String s1="Java";
        String s2="Java";
        if(s1==s2)
        {
            System.out.println("True");
        }
        else
        {
            System.out.println("Flase");
        }

    }

}
```

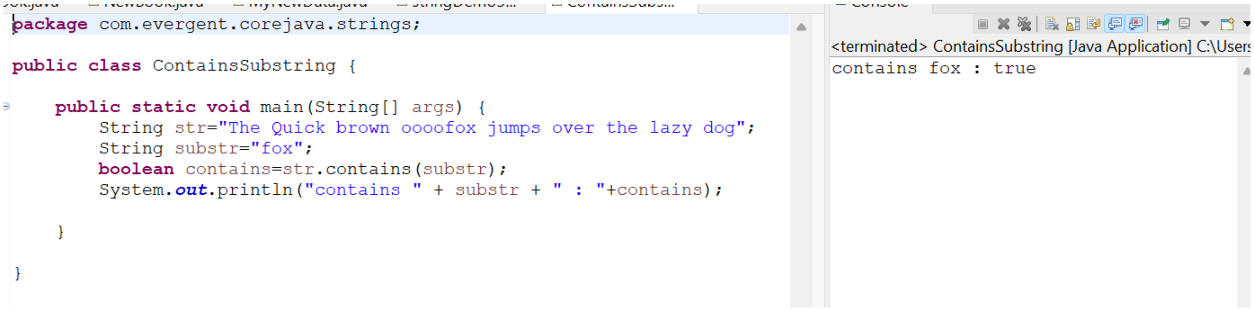
```
<terminated> StringDemo1 [Java Application] C:\Users\sandhya.pup
False
```



```
package com.evergent.corejava.strings;
//String methods

public class stringDemo3_methods {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        String name=new String("       JAVAworld");
        System.out.println("length:"+name.length());
        System.out.println("lowercase:"+name.toLowerCase());
        System.out.println("uppercase:"+name.toUpperCase());
        System.out.println("trim:"+name.trim());
        // trim removes spaces before and after the string

    }

}
```

```
<terminated> stringDemo3_methods [Java Application
length:16
lowercase:       javaworld
uppercase:       JAVAWORLD
trim:JAVAworld
```

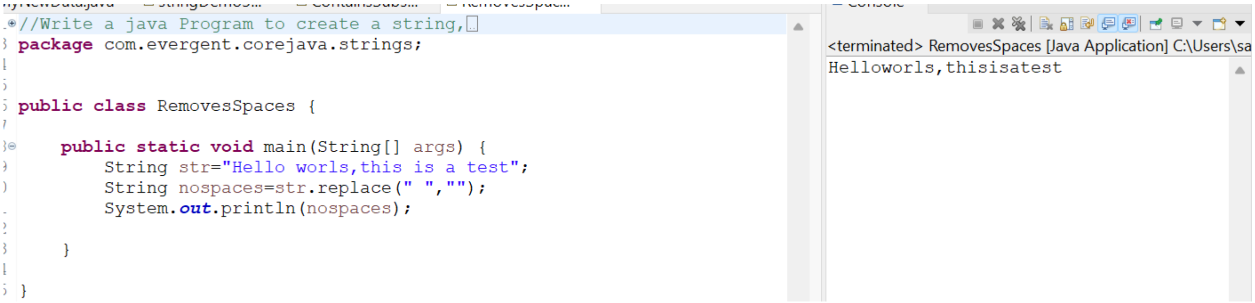
1.create a java program that creates a string and checks if it contains  specific subString and then prints out the result



```
package com.evergent.corejava.strings;

public class ContainsSubstring {

    public static void main(String[] args) {
        String str="The Quick brown oooofox jumps over the lazy dog";
        String substr="fox";
        boolean contains=str.contains(substr);
        System.out.println("contains " + substr + " : "+contains);

    }

}
```

```
<terminated> ContainsSubstring [Java Application] C:\Users
contains fox : true
```

2. Write a java prgm to create a String ,remove all spaces from the string and then print out the result

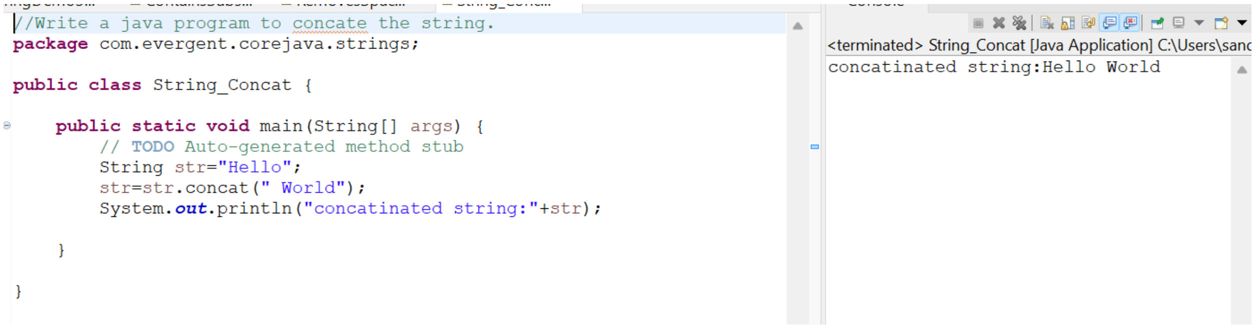


```
//Write a java Program to create a string,
package com.evergent.corejava.strings;

public class RemovesSpaces {

    public static void main(String[] args) {
        String str="Hello worls,this is a test";
        String nospaces=str.replace(" ","");
        System.out.println(nospaces);

    }

}
```

```
<terminated> RemovesSpaces [Java Application] C:\Users\sa
Helloworls,thisisatest
```

String concatination:

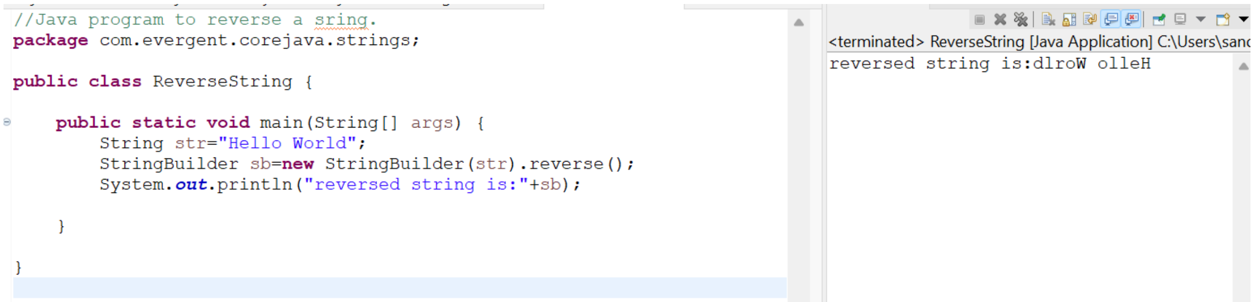
Strings can be concatenated using + operator (or) concat.



```
//Write a java program to concate the string.
package com.evergent.corejava.strings;

public class String_Concat {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        String str="Hello";
        str=str.concat(" World");
        System.out.println("concatinated string:"+str);

    }

}
```

```
<terminated> String_Concat [Java Application] C:\Users\sanc
concatinated string:Hello World
```

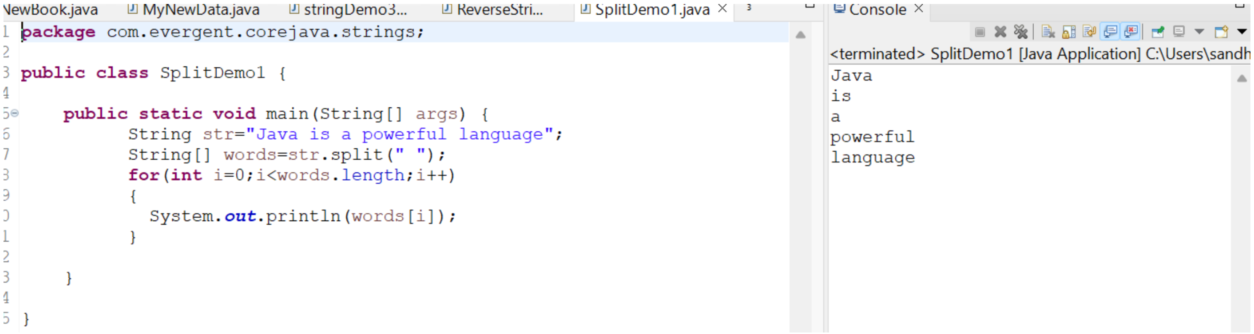
Reverse of a String:



```
//Java program to reverse a sring.
package com.evergent.corejava.strings;

public class ReverseString {

    public static void main(String[] args) {
        String str="Hello World";
        StringBuilder sb=new StringBuilder(str).reverse();
        System.out.println("reversed string is:"+sb);

    }

}
```

```
<terminated> ReverseString [Java Application] C:\Users\sanc
reversed string is:dlroW olleH
```

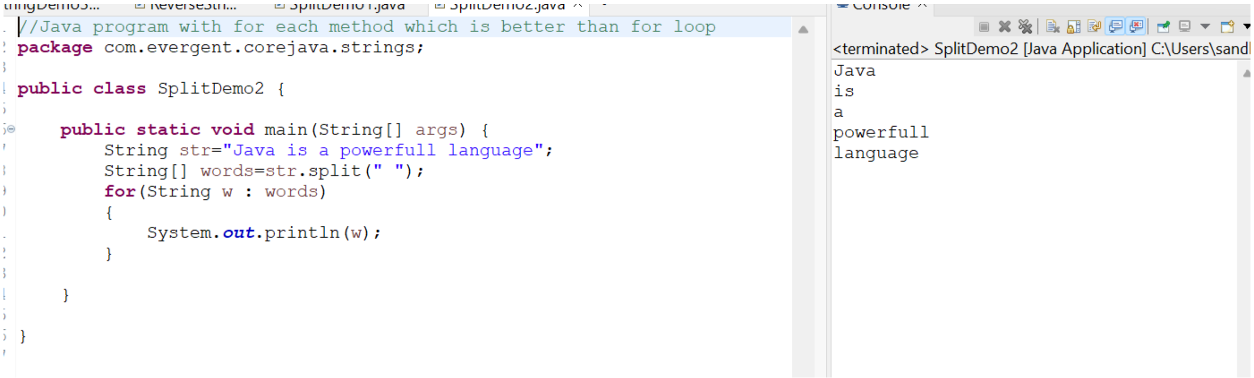
Splitting of any sentence (or)  text on spaces



```
package com.evergent.corejava.strings;

public class SplitDemo1 {

    public static void main(String[] args) {
            String str="Java is a powerful language";
            String[] words=str.split(" ");
            for(int i=0;i<words.length;i++)
            {
              System.out.println(words[i]);
            }

    }
}
```

```
<terminated> SplitDemo1 [Java Application] C:\Users\sandh
Java
is
a
powerful
language
```

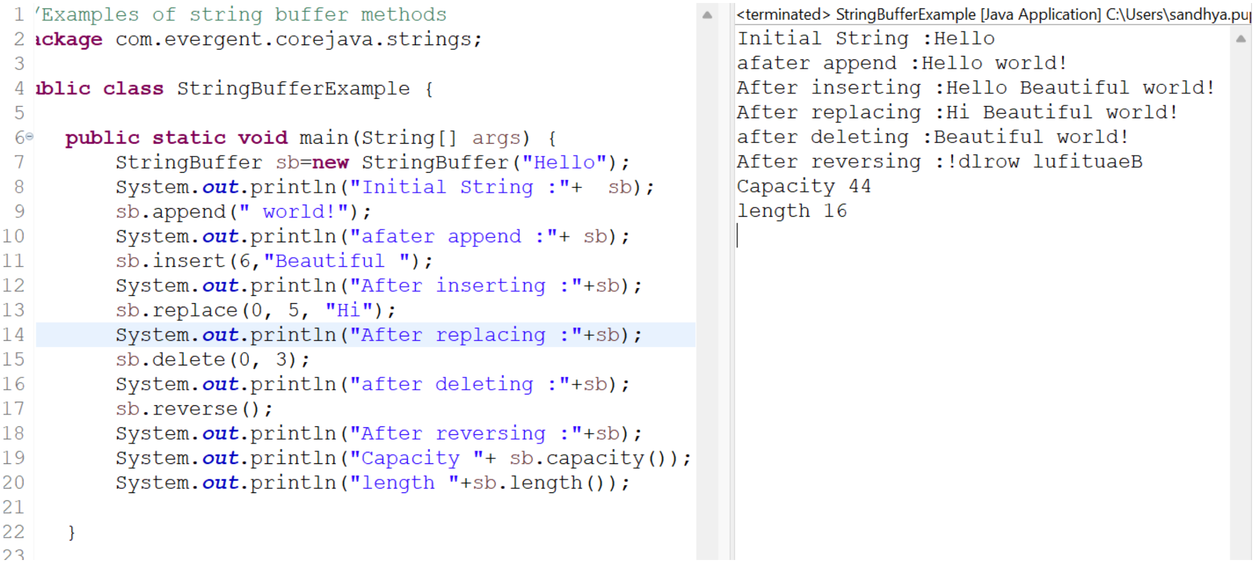
Using for each loop:



```
//Java program with for each method which is better than for loop
package com.evergent.corejava.strings;

public class SplitDemo2 {

    public static void main(String[] args) {
        String str="Java is a powerfull language";
        String[] words=str.split(" ");
        for(String w : words)
        {
            System.out.println(w);
        }

    }

}
```

```
<terminated> SplitDemo2 [Java Application] C:\Users\sandl
Java
is
a
powerfull
language
```

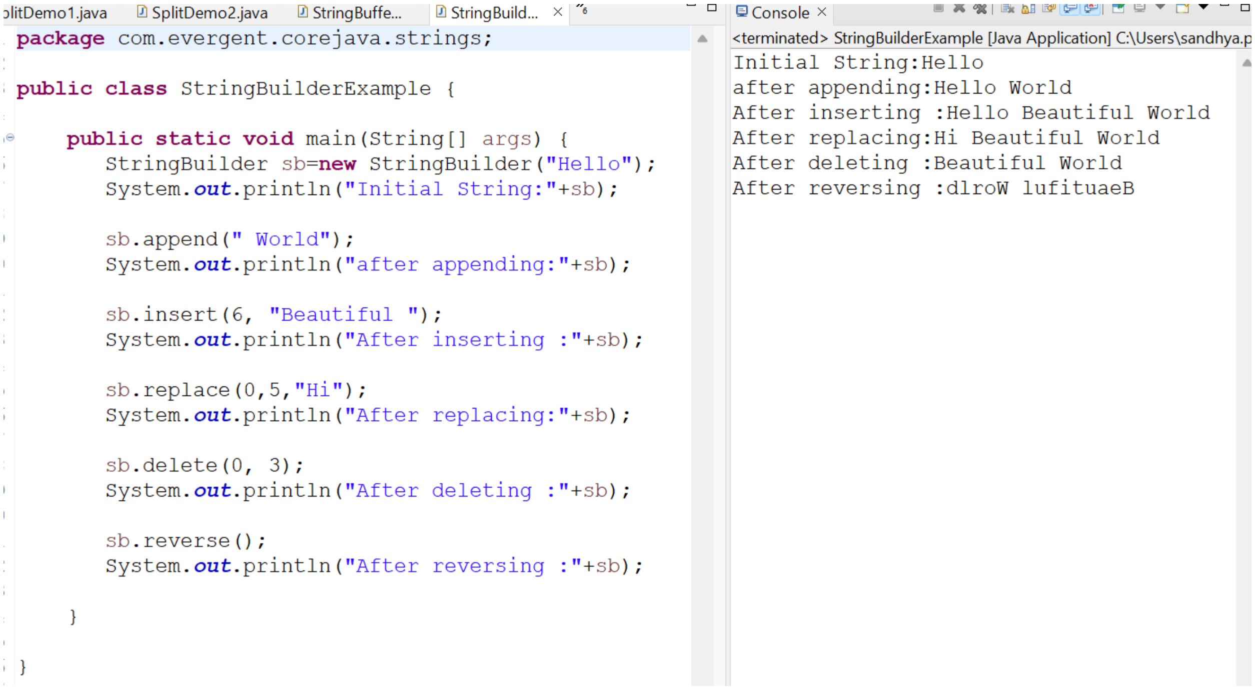
String Buffer Examples:



```
'Examples of string buffer methods
ackage com.evergent.corejava.strings;

ublic class StringBufferExample {

    public static void main(String[] args) {
        StringBuffer sb=new StringBuffer("Hello");
        System.out.println("Initial String :"+  sb);
        sb.append(" world!");
        System.out.println("afater append :"+ sb);
        sb.insert(6,"Beautiful ");
        System.out.println("After inserting :"+sb);
        sb.replace(0, 5, "Hi");
        System.out.println("After replacing :"+sb);
        sb.delete(0, 3);
        System.out.println("after deleting :"+sb);
        sb.reverse();
        System.out.println("After reversing :"+sb);
        System.out.println("Capacity "+ sb.capacity());
        System.out.println("length "+sb.length());

    }
```

```
<terminated> StringBufferExample [Java Application] C:\Users\sandhya.pu
Initial String :Hello
afater append :Hello world!
After inserting :Hello Beautiful world!
After replacing :Hi Beautiful world!
after deleting :Beautiful world!
After reversing :!dlrow lufituaeB
Capacity 44
length 16
```

StringBuilder Examples


plitDemo1.java
SplitDemo2.java
StringBuffe...
StringBuild...
Console
<terminated> StringBuilderExample [Java Application] C:\Users\sandhya.p

```java
package com.evergent.corejava.strings;

public class StringBuilderExample {

    public static void main(String[] args) {
       StringBuilder sb=new StringBuilder("Hello");
       System.out.println("Initial String:"+sb);

       sb.append(" World");
       System.out.println("after appending:"+sb);

       sb.insert(6, "Beautiful ");
       System.out.println("After inserting :"+sb);

       sb.replace(0,5,"Hi");
       System.out.println("After replacing:"+sb);

       sb.delete(0, 3);
       System.out.println("After deleting :"+sb);

       sb.reverse();
       System.out.println("After reversing :"+sb);

    }

}
```

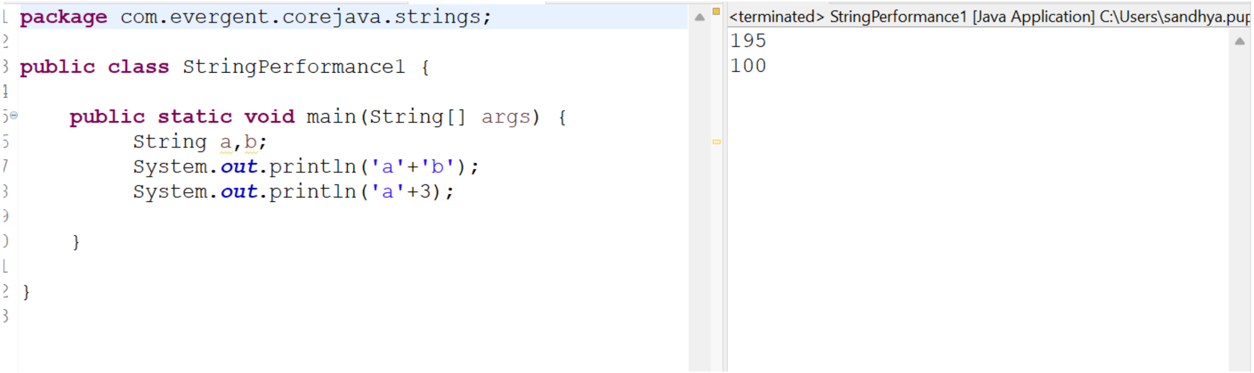
```
Initial String:Hello
after appending:Hello World
After inserting :Hello Beautiful World
After replacing:Hi Beautiful World
After deleting :Beautiful World
After reversing :dlroW lufituaeB
```

**String class important points:**

1. In java a string is a sequence of characters ,often used to represent text.
2. Strings are objects in java and are instances of the string class,which is part of the java java.lang package
3. Key features of strings in java:
   A. Immutable:once a string object is created ,it cannot be changed .
4. Java optimizes memory usage by storing strings in special area of memory as string pool
5. If two strings have the same value and are created without using new keyword they will refer to same object in the stringpool.
6. We can create  a string in java in multiple ways:
   a. Using string literals :str=”hello world”;
   b. Using the new keyword
      String str=new String(“hello, world”);

**String Performance Examples:**

1)

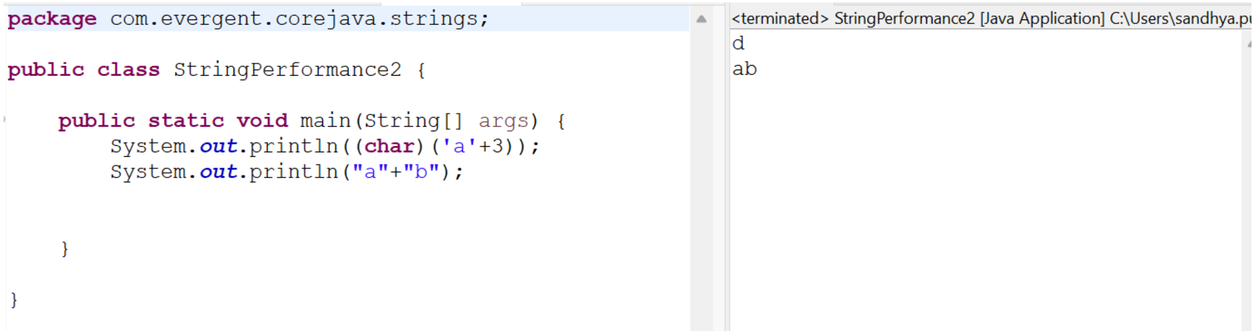


```java
package com.evergent.corejava.strings;

public class StringPerformance1 {

    public static void main(String[] args) {
        String a,b;
        System.out.println('a'+'b');
        System.out.println('a'+3);

    }

}
```

<terminated> StringPerformance1 [Java Application] C:\Users\sandhya.pup

```
195
100
```

2)

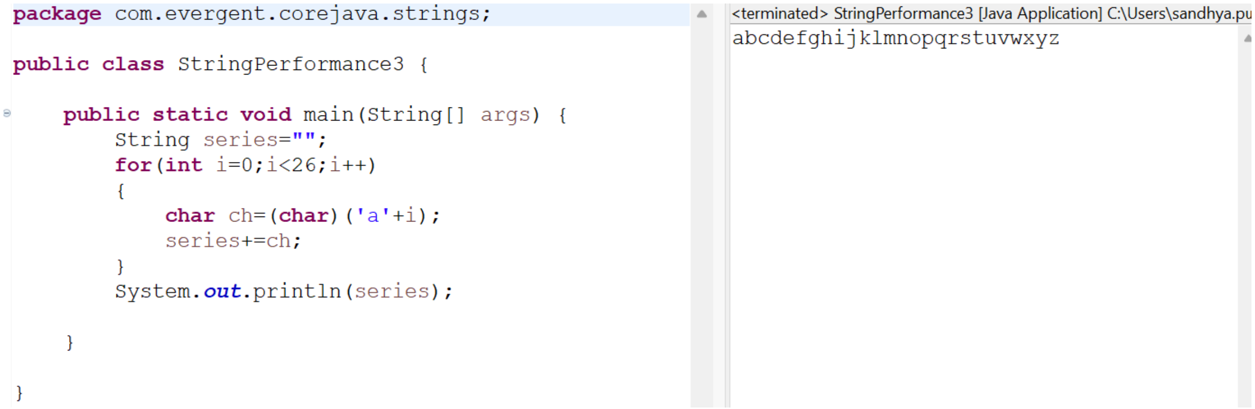


```java
package com.evergent.corejava.strings;

public class StringPerformance2 {

    public static void main(String[] args) {
        System.out.println((char)('a'+3));
        System.out.println("a"+"b");


    }

}
```

<terminated> StringPerformance2 [Java Application] C:\Users\sandhya.pu

```
d
ab
```

3)

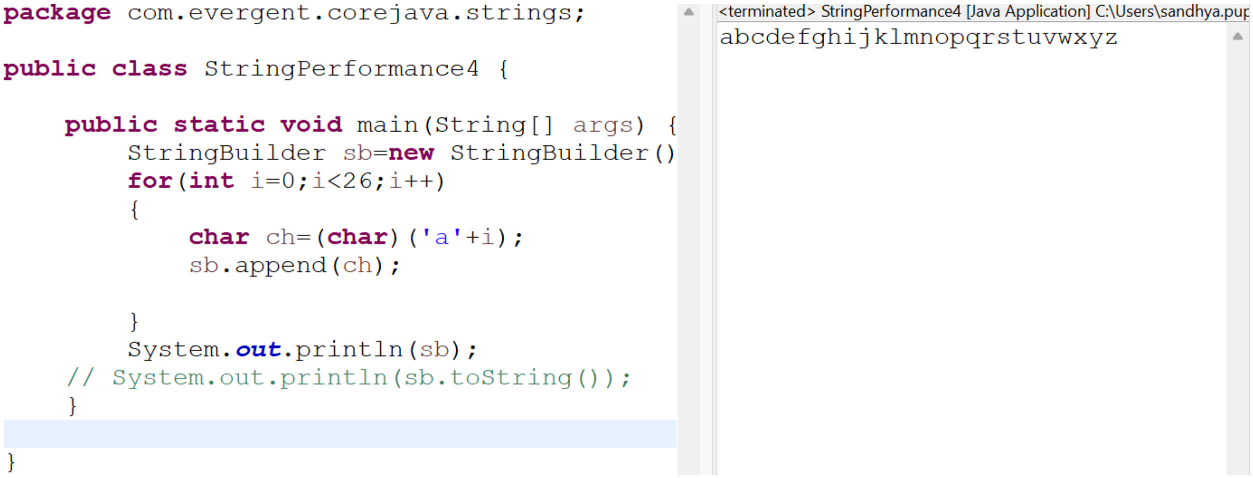


```java
package com.evergent.corejava.strings;

public class StringPerformance3 {

    public static void main(String[] args) {
        String series="";
        for(int i=0;i<26;i++)
        {
            char ch=(char)('a'+i);
            series+=ch;
        }
        System.out.println(series);

    }

}
```

<terminated> StringPerformance3 [Java Application] C:\Users\sandhya.pu

```
abcdefghijklmnopqrstuvwxyz
```

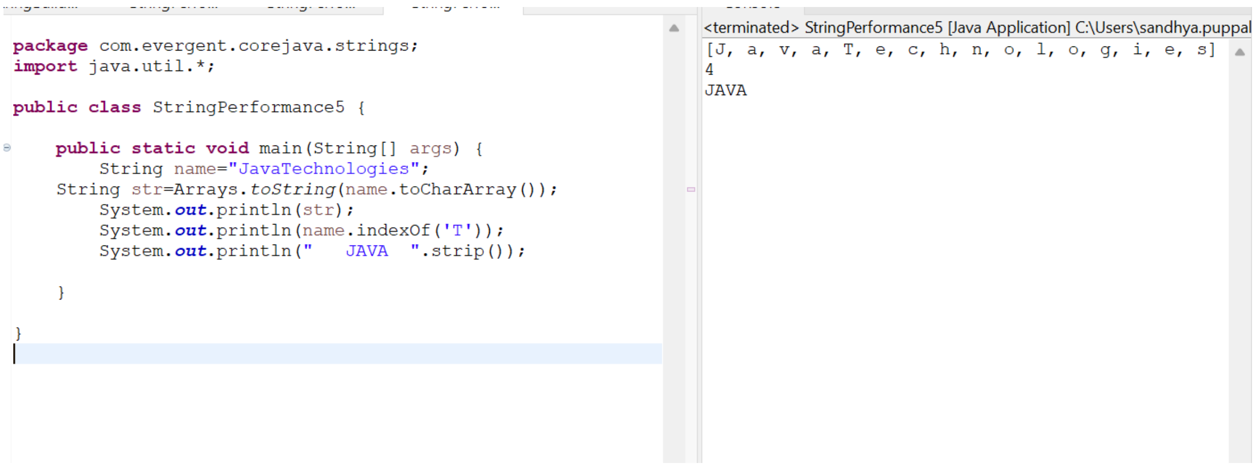
4)



```
package com.evergent.corejava.strings;

public class StringPerformance4 {

    public static void main(String[] args) {
        StringBuilder sb=new StringBuilder()
        for(int i=0;i<26;i++)
        {
            char ch=(char)('a'+i);
            sb.append(ch);

        }
        System.out.println(sb);
    // System.out.println(sb.toString());
    }

}
```

```
<terminated> StringPerformance4 [Java Application] C:\Users\sandhya.pup
abcdefghijklmnopqrstuvwxyz
```

5)



```
package com.evergent.corejava.strings;
import java.util.*;

public class StringPerformance5 {

    public static void main(String[] args) {
        String name="JavaTechnologies";
    String str=Arrays.toString(name.toCharArray());
        System.out.println(str);
        System.out.println(name.indexOf('T'));
        System.out.println("    JAVA   ".strip());

    }

}
```

```
<terminated> StringPerformance5 [Java Application] C:\Users\sandhya.puppal
[J, a, v, a, T, e, c, h, n, o, l, o, g, i, e, s]
4
JAVA
```

## Interface

1) Interface is a Keyword.

2) We can declare methods signatures only,but not implemnetation.

3) By default ,all interface methods are abstract.

4) If any class implements interface the class should be override all interface methods otherwise the class will be showing compile time error.

5) We cannot create object to interface but we can create reference to interface.

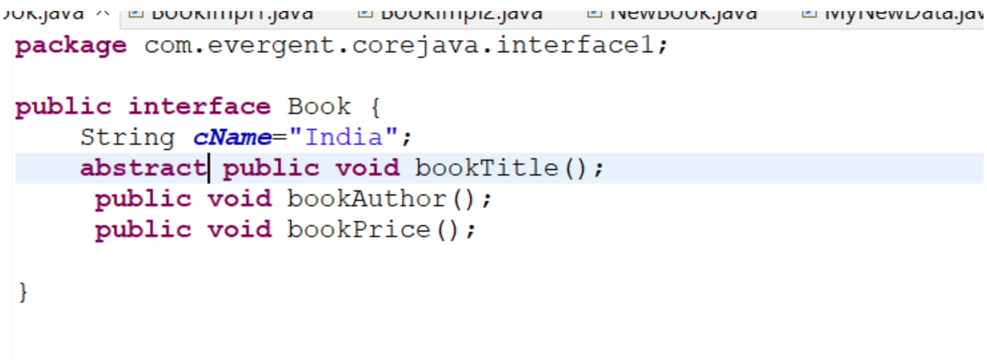
6) We can declare variables inside interface all are public static final.

7) Java will support multiple inheritance through interface.

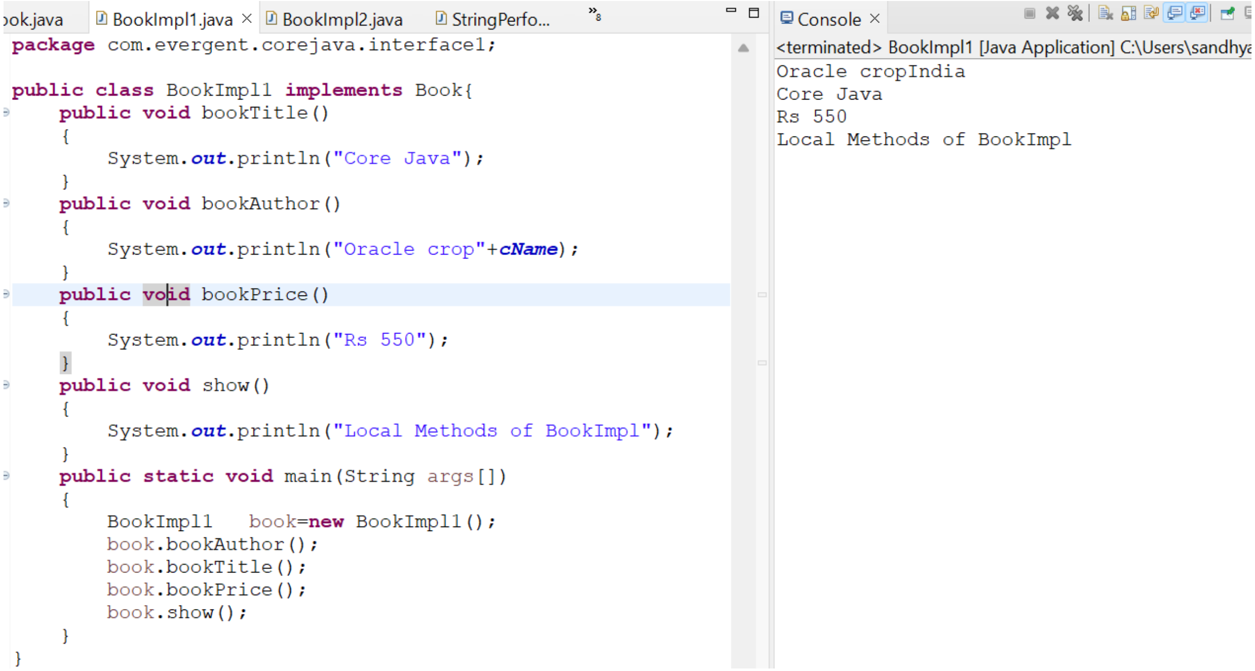
8)  One class can implements more than one interface.

9) One interface can extend other interface.

Program for first 4 and 6th points



```
package com.evergent.corejava.interface1;

public interface Book {
     String cName="India";
     abstract public void bookTitle();
      public void bookAuthor();
      public void bookPrice();

}
```

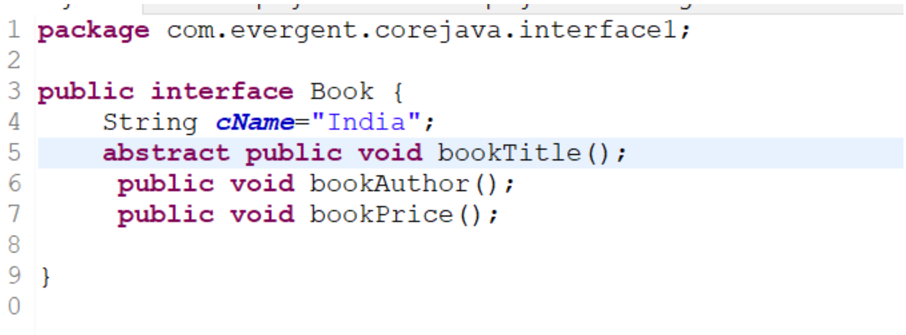


```
package com.evergent.corejava.interface1;

public class BookImpl1 implements Book{
    public void bookTitle()
    {
        System.out.println("Core Java");
    }
    public void bookAuthor()
    {
        System.out.println("Oracle crop"+cName);
    }
    public void bookPrice()
    {
        System.out.println("Rs 550");
    }
    public void show()
    {
        System.out.println("Local Methods of BookImpl");
    }
    public static void main(String args[])
    {
        BookImpl1   book=new BookImpl1();
        book.bookAuthor();
        book.bookTitle();
        book.bookPrice();
        book.show();
    }
}
```

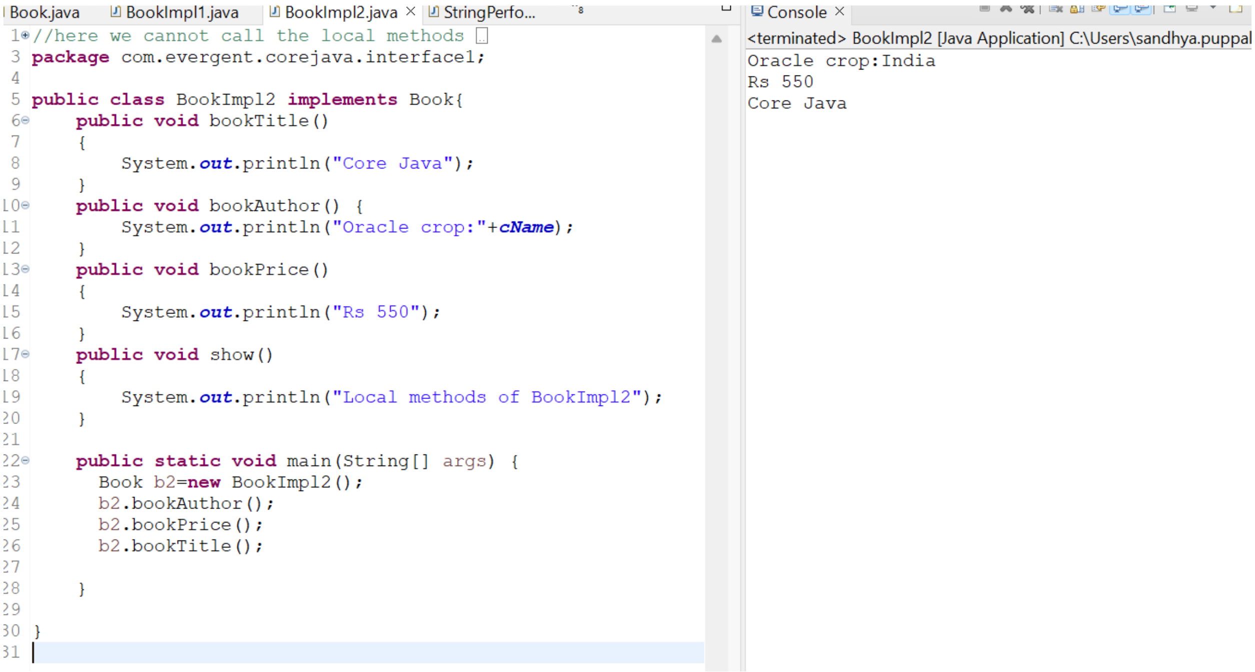
```
<terminated> BookImpl1 [Java Application] C:\Users\sandhya
Oracle cropIndia
Core Java
Rs 550
Local Methods of BookImpl
```

2)  program for 5th point

We cannot create object to interface but we can create reference to interface.



```java
package com.evergent.corejava.interface1;

public interface Book {
     String cName="India";
     abstract public void bookTitle();
      public void bookAuthor();
      public void bookPrice();

}
```

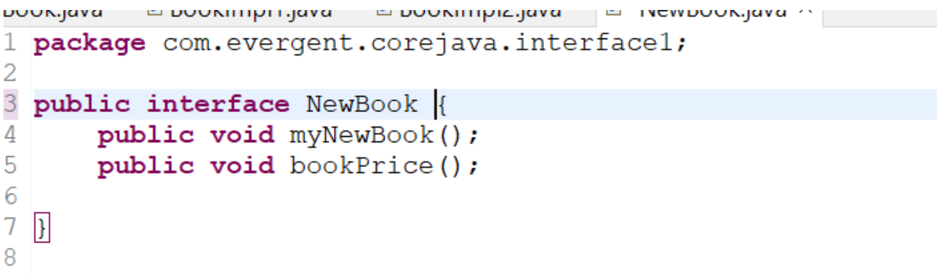


```java
//here we cannot call the local methods 
package com.evergent.corejava.interface1;

public class BookImpl2 implements Book{
    public void bookTitle()
    {
        System.out.println("Core Java");
    }
    public void bookAuthor() {
        System.out.println("Oracle crop:"+cName);
    }
    public void bookPrice()
    {
        System.out.println("Rs 550");
    }
    public void show()
    {
        System.out.println("Local methods of BookImpl2");
    }

    public static void main(String[] args) {
      Book b2=new BookImpl2();
      b2.bookAuthor();
      b2.bookPrice();
      b2.bookTitle();

    }

}
```

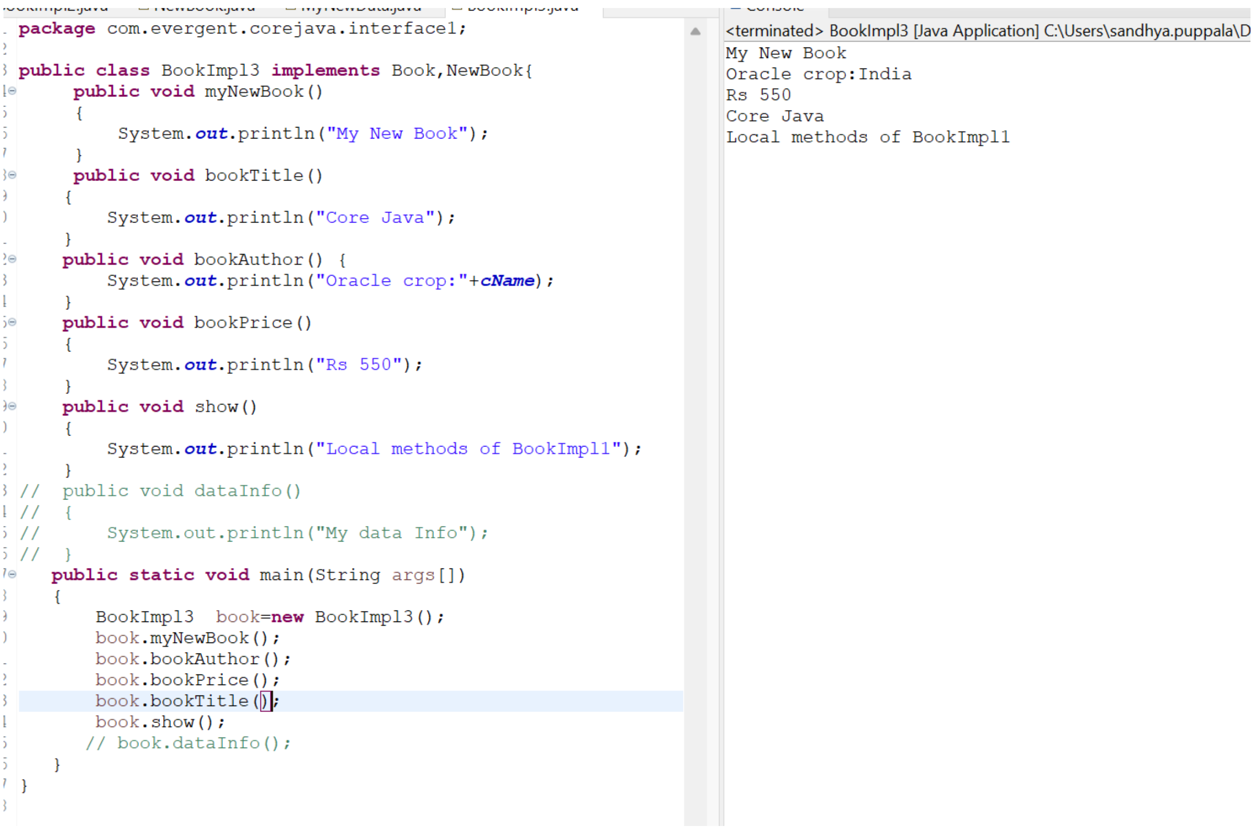
```
<terminated> BookImpl2 [Java Application] C:\Users\sandhya.puppal
Oracle crop:India
Rs 550
Core Java
```

3) Program for 7th point

Java will support multiple inheritance through interface.



```java
package com.evergent.corejava.interface1;

public interface NewBook {
     public void myNewBook();
     public void bookPrice();

}
```

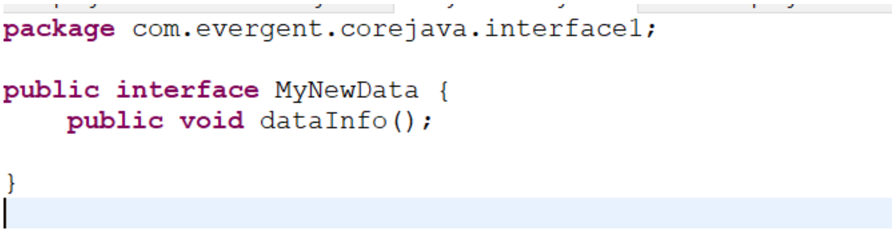


```
package com.evergent.corejava.interface1;

public class BookImpl3 implements Book,NewBook{
     public void myNewBook()
     {
         System.out.println("My New Book");
     }
     public void bookTitle()
    {
        System.out.println("Core Java");
    }
    public void bookAuthor() {
        System.out.println("Oracle crop:"+cName);
    }
    public void bookPrice()
    {
        System.out.println("Rs 550");
    }
    public void show()
    {
        System.out.println("Local methods of BookImpl1");
    }
//  public void dataInfo()
//  {
//      System.out.println("My data Info");
//  }
   public static void main(String args[])
   {
       BookImpl3  book=new BookImpl3();
       book.myNewBook();
       book.bookAuthor();
       book.bookPrice();
       book.bookTitle();
       book.show();
      // book.dataInfo();
   }
}
```

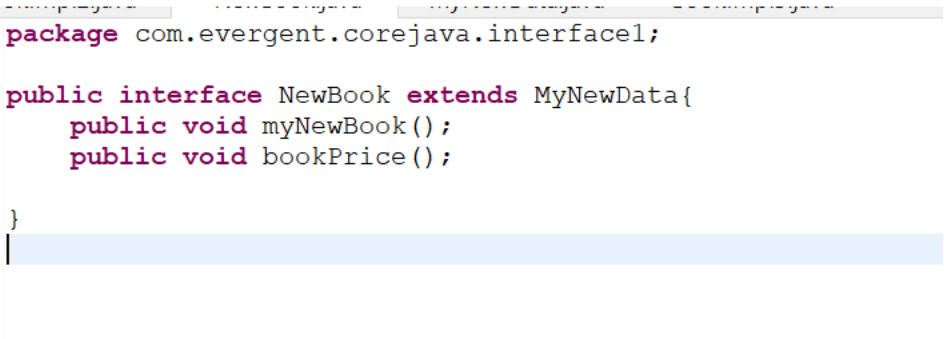
```
<terminated> BookImpl3 [Java Application] C:\Users\sandhya.puppala\D
My New Book
Oracle crop:India
Rs 550
Core Java
Local methods of BookImpl1
```

4) Program for 9th point.

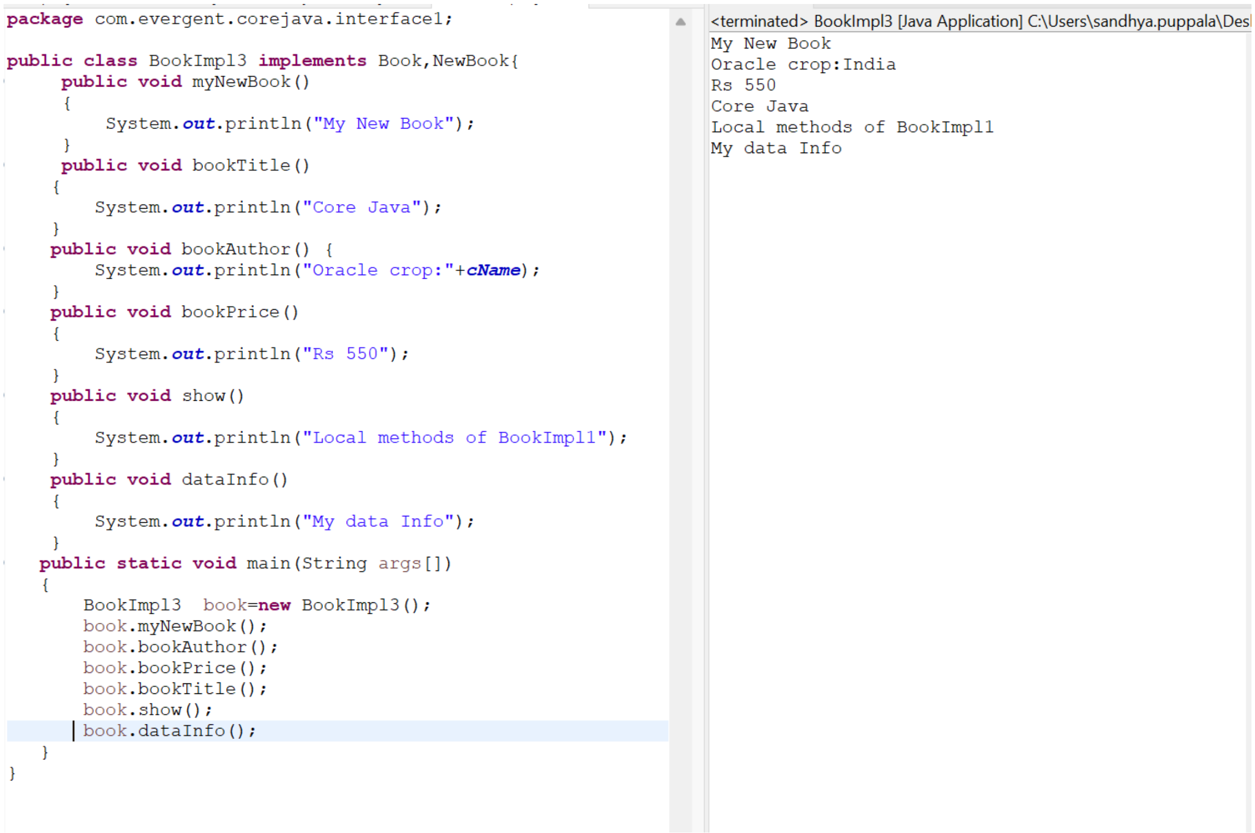
One interface can extend other interface.



```
package com.evergent.corejava.interface1;

public interface MyNewData {
     public void dataInfo();

}
```



```
package com.evergent.corejava.interface1;

public interface NewBook extends MyNewData{
     public void myNewBook();
     public void bookPrice();

}
```



```java
package com.evergent.corejava.interface1;

public class BookImpl3 implements Book,NewBook{
     public void myNewBook()
     {
         System.out.println("My New Book");
     }
     public void bookTitle()
    {
        System.out.println("Core Java");
    }
    public void bookAuthor() {
        System.out.println("Oracle crop:"+cName);
    }
    public void bookPrice()
    {
        System.out.println("Rs 550");
    }
    public void show()
    {
        System.out.println("Local methods of BookImpl1");
    }
    public void dataInfo()
    {
        System.out.println("My data Info");
    }
   public static void main(String args[])
   {
       BookImpl3  book=new BookImpl3();
       book.myNewBook();
       book.bookAuthor();
       book.bookPrice();
       book.bookTitle();
       book.show();
       book.dataInfo();
   }
}
```

```
<terminated> BookImpl3 [Java Application] C:\Users\sandhya.puppala\Des
My New Book
Oracle crop:India
Rs 550
Core Java
Local methods of BookImpl1
My data Info
```

## Abstarct Class

1) Abstarct is Keyword.
2) Abstarct class having abstarct methods and concrete(implemented) methods.

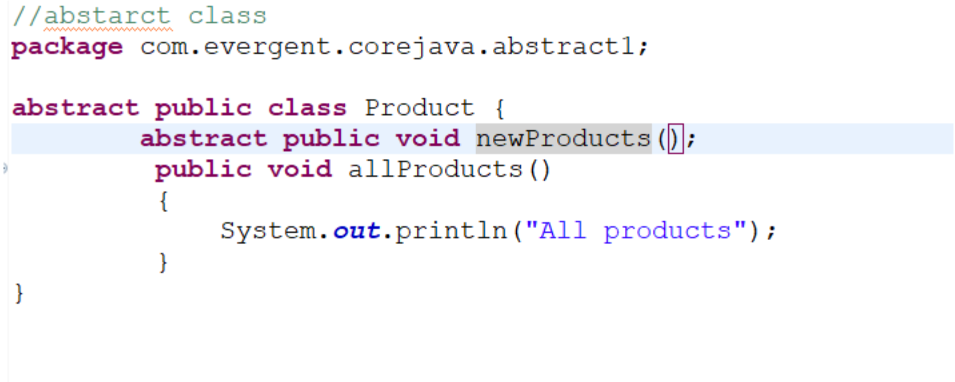
3) If any class having one abstarct method,the class should be declare as abstarct keyword,otherwise the class will be showing complile time error.

4) If any class extends abstarct class,the class should be override all abstarct methods,Otherwise the class will be showing compile time error.

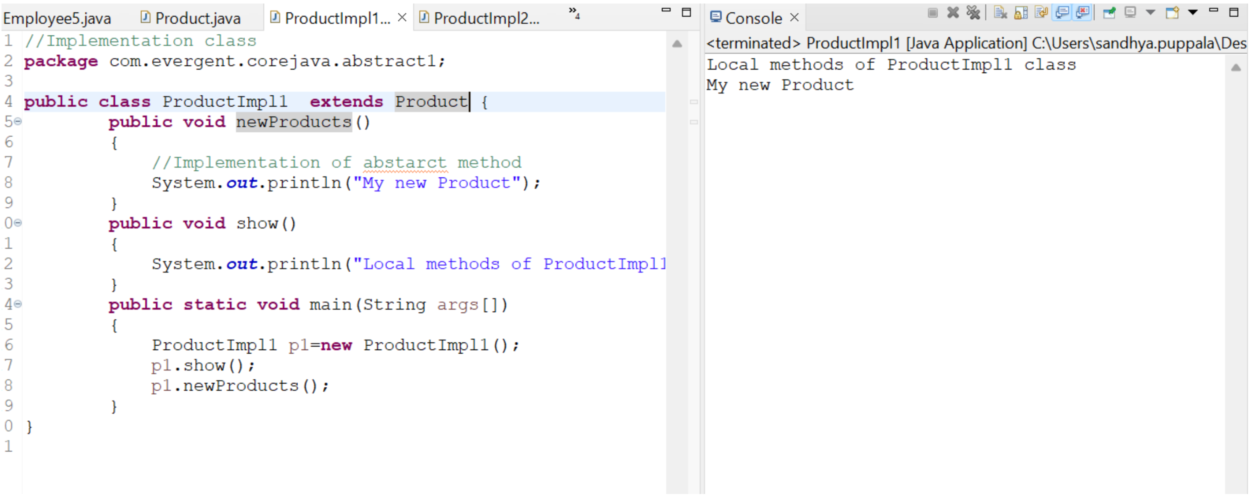
5) We cannot create object to abstrcat to Abstract class but we can create reference to Abstract class.

6) When we extend one abstarct either we should over ride the abstarct method (or) class should be declared as Abstract.
7) Uses of Abstraction is,not implemented methods can be implemented methods should be used straight away.
8) We can declare Abstract class also with zero abstract methods,we cannot create object.

**Program 1**



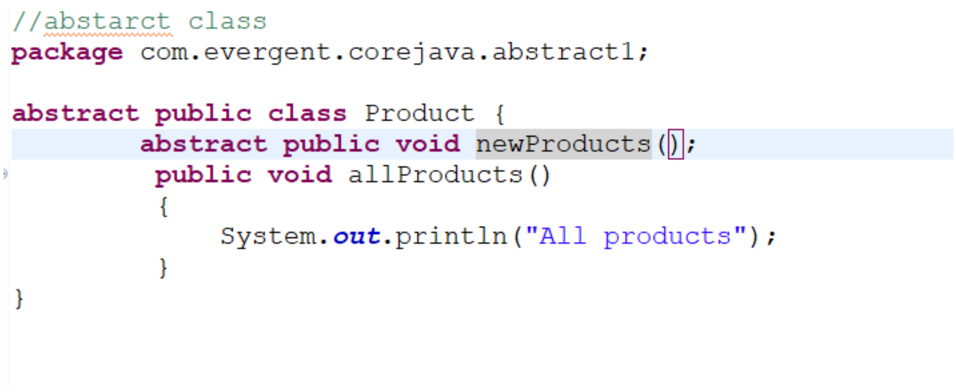
```
//abstarct class
package com.evergent.corejava.abstract1;

abstract public class Product {
        abstract public void newProducts();
         public void allProducts()
         {
             System.out.println("All products");
         }
}
```



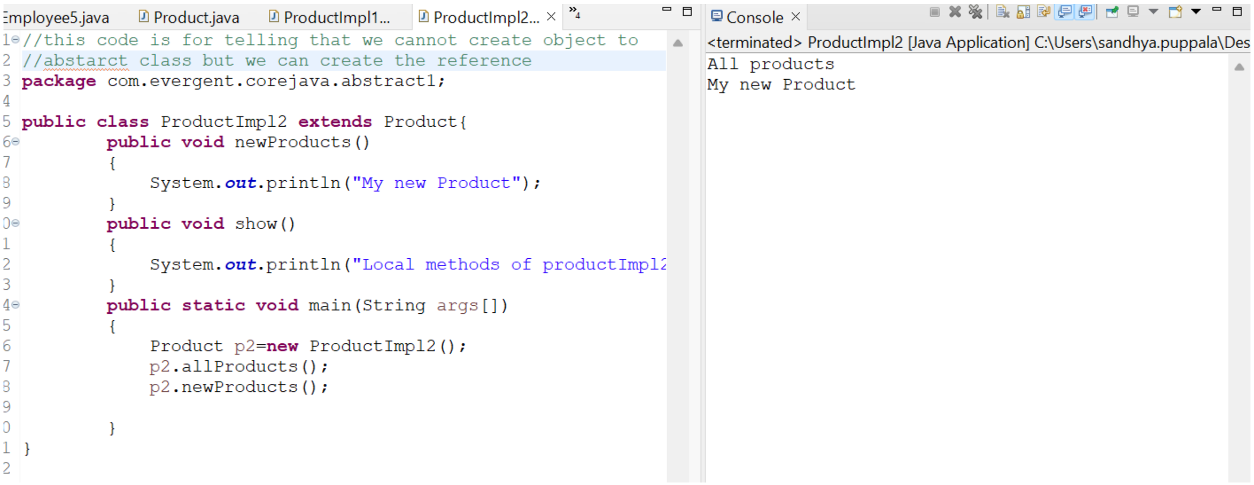
```
//Implementation class
package com.evergent.corejava.abstract1;

public class ProductImpl1  extends Product {
        public void newProducts()
        {
            //Implementation of abstarct method
            System.out.println("My new Product");
        }
        public void show()
        {
            System.out.println("Local methods of ProductImpl1
        }
        public static void main(String args[])
        {
            ProductImpl1 p1=new ProductImpl1();
            p1.show();
            p1.newProducts();
        }
}
```

```
<terminated> ProductImpl1 [Java Application] C:\Users\sandhya.puppala\Des
Local methods of ProductImpl1 class
My new Product
```

## Program 2)



```
//abstarct class
package com.evergent.corejava.abstract1;

abstract public class Product {
          abstract public void newProducts();
           public void allProducts()
           {
                System.out.println("All products");
           }
}
```



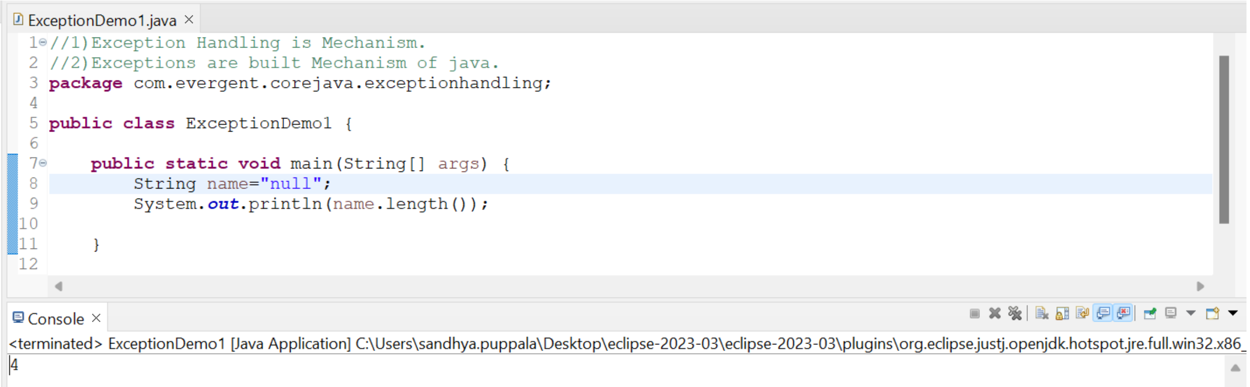
```
//this code is for telling that we cannot create object to
//abstarct class but we can create the reference
package com.evergent.corejava.abstract1;

public class ProductImpl2 extends Product{
        public void newProducts()
        {
            System.out.println("My new Product");
        }
        public void show()
        {
            System.out.println("Local methods of productImpl2
        }
        public static void main(String args[])
        {
            Product p2=new ProductImpl2();
            p2.allProducts();
            p2.newProducts();

        }
}
```

```
<terminated> ProductImpl2 [Java Application] C:\Users\sandhya.puppala\Des
All products
My new Product
```

## Exception Handling

1) Exception Handling is Mechanism.
2) Exceptions are built Mechanism of java.
3) All Exception are executed while abnormal conditions only.
4) If there is Normal flow it wont execute any exceptions.
5) Once any exceptions are occurring in java then remaining lines of code is unreachable.
6) Java.lang.Throwable  is super class for Exception and errors.
7) There are 2 types of Exceptions are there in Java
   A) Checked Exceptions
   B) Unchecked Exceptions
8) All checked Exceptions are compile time exceptions.
9) All Unchecked Exceptions are Runtime Exceptions.
10) There are 5 keywords in Exception Handling.
   A) Try
   B) Catch()
   C) Finally
   D) throws
   E) Throw
11) try is for business logic
12) Catch is for handling Exceptions
13) finally  is block,if exceptions id occurs or not,finally block will execute.
14) throws an exception will be executed method by method
15) throw is for runtime exceptions and will call predicted exceptions or user defined exception.
16) try followed by either catch block or finally block.
17) We should follow exceptions hierarchial.
18) We can create our own (user defined) exceptions
19) Our own exceptions extends exception or Runtime exception.
20) All exceptions extends exception or runtime exception
21) There is two exception in class,Developer should be handle one after one.
22) Developer cant handle the error.

Program for first 2 points
Exception Handling is Mechanism.
Exceptions are built Mechanism of java.

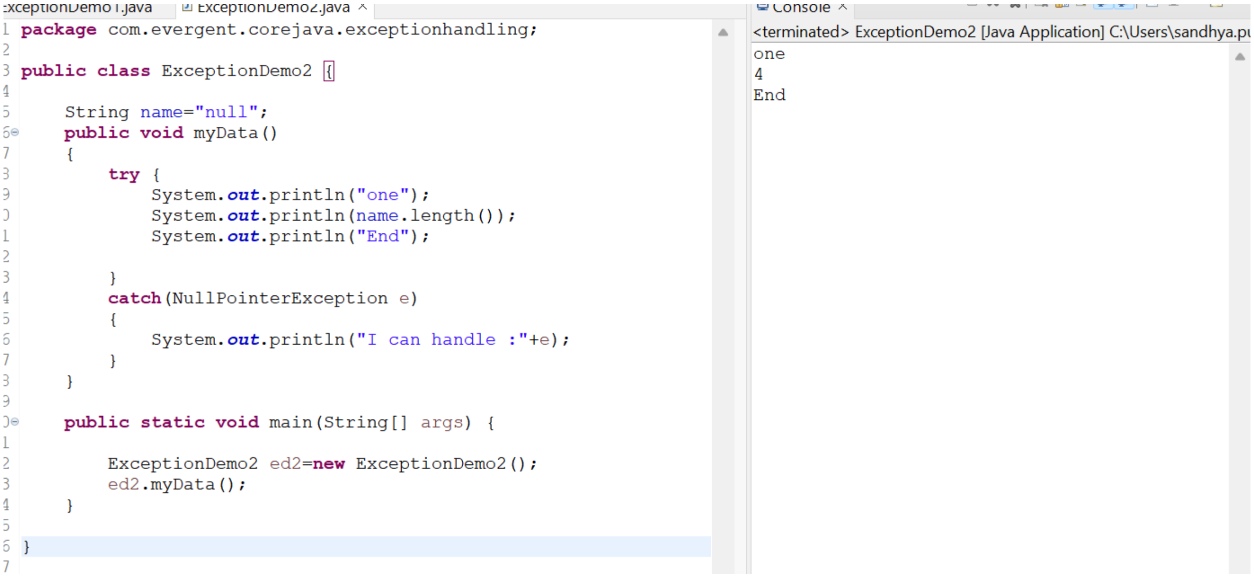


```
//1)Exception Handling is Mechanism.
//2)Exceptions are built Mechanism of java.
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo1 {

    public static void main(String[] args) {
        String name="null";
        System.out.println(name.length());

    }

```

```
<terminated> ExceptionDemo1 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_
4
```

Program for 3,4,5 points

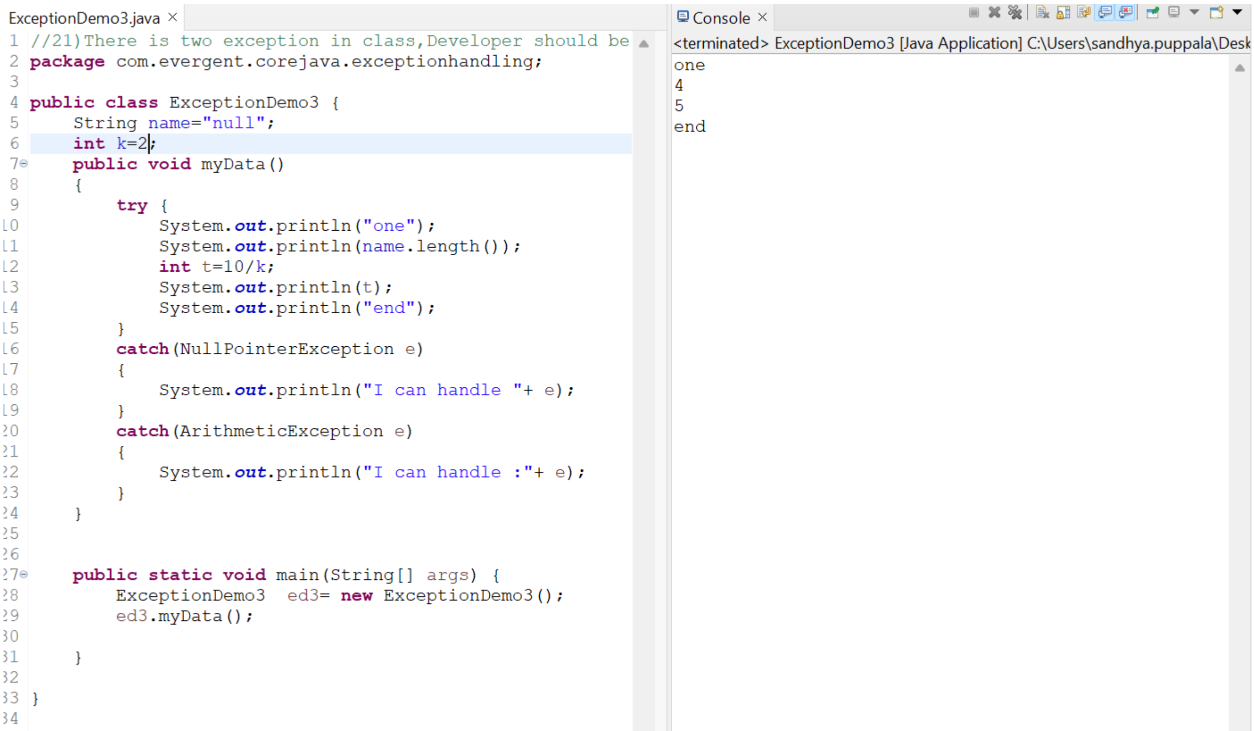
All Exception are executed while abnormal conditions only.
If there is Normal flow it wont execute any exceptions.
Once any exceptions are occurring in java then remaining lines of code is unreachable.



```
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo2 {

    String name="null";
    public void myData()
    {
        try {
            System.out.println("one");
            System.out.println(name.length());
            System.out.println("End");

        }
        catch(NullPointerException e)
        {
            System.out.println("I can handle :"+e);
        }
    }

    public static void main(String[] args) {

        ExceptionDemo2 ed2=new ExceptionDemo2();
        ed2.myData();
    }

}
```

```
<terminated> ExceptionDemo2 [Java Application] C:\Users\sandhya.pu
one
4
End
```

Program for point 21
23) There is two exception in class,Developer should be handle one after one.

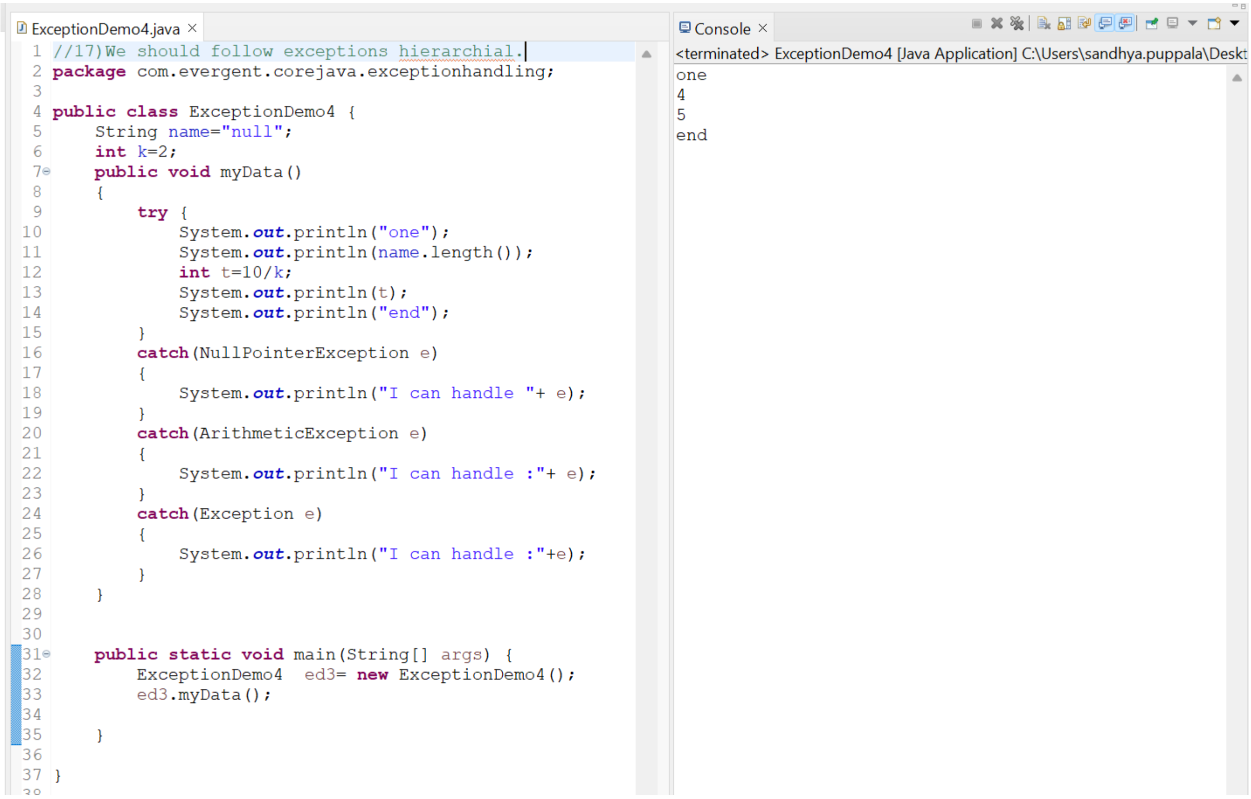


```java
//21)There is two exception in class,Developer should be
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo3 {
    String name="null";
    int k=2;
    public void myData()
    {
        try {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            System.out.println(t);
            System.out.println("end");
        }
        catch(NullPointerException e)
        {
            System.out.println("I can handle "+ e);
        }
        catch(ArithmeticException e)
        {
            System.out.println("I can handle :"+ e);
        }
    }


    public static void main(String[] args) {
        ExceptionDemo3  ed3= new ExceptionDemo3();
        ed3.myData();

    }

}
```

```
<terminated> ExceptionDemo3 [Java Application] C:\Users\sandhya.puppala\Desk
one
4
5
end
```

Program for 17 point

We should follow exceptions hierarchial.

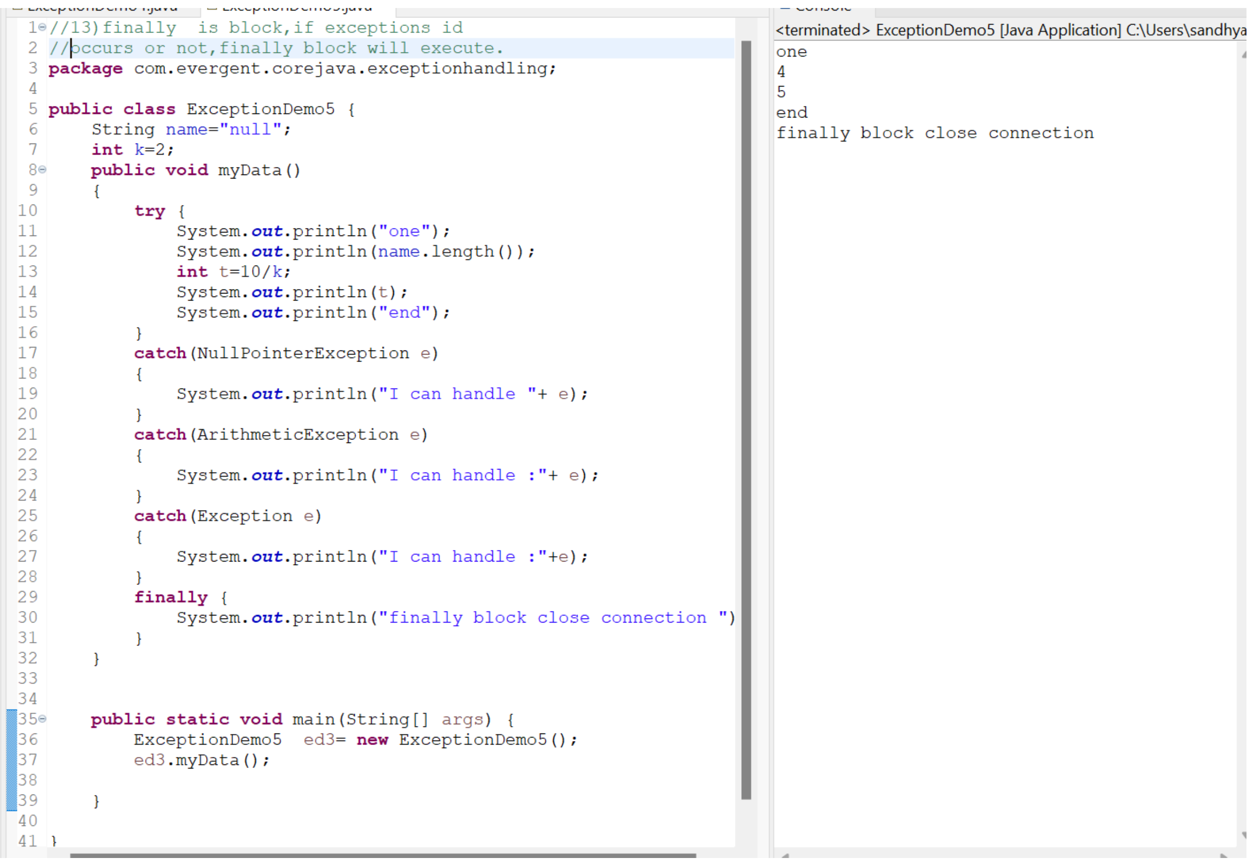


```java
//17)We should follow exceptions hierarchial.
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo4 {
    String name="null";
    int k=2;
    public void myData()
    {
        try {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            System.out.println(t);
            System.out.println("end");
        }
        catch(NullPointerException e)
        {
            System.out.println("I can handle "+ e);
        }
        catch(ArithmeticException e)
        {
            System.out.println("I can handle :"+ e);
        }
        catch(Exception e)
        {
            System.out.println("I can handle :"+e);
        }
    }


    public static void main(String[] args) {
        ExceptionDemo4  ed3= new ExceptionDemo4();
        ed3.myData();

    }

}
```

```
<terminated> ExceptionDemo4 [Java Application] C:\Users\sandhya.puppala\Deskt
one
4
5
end
```

Program for 13 point

finally  is block,if exceptions id occurs or not,finally block will execute.

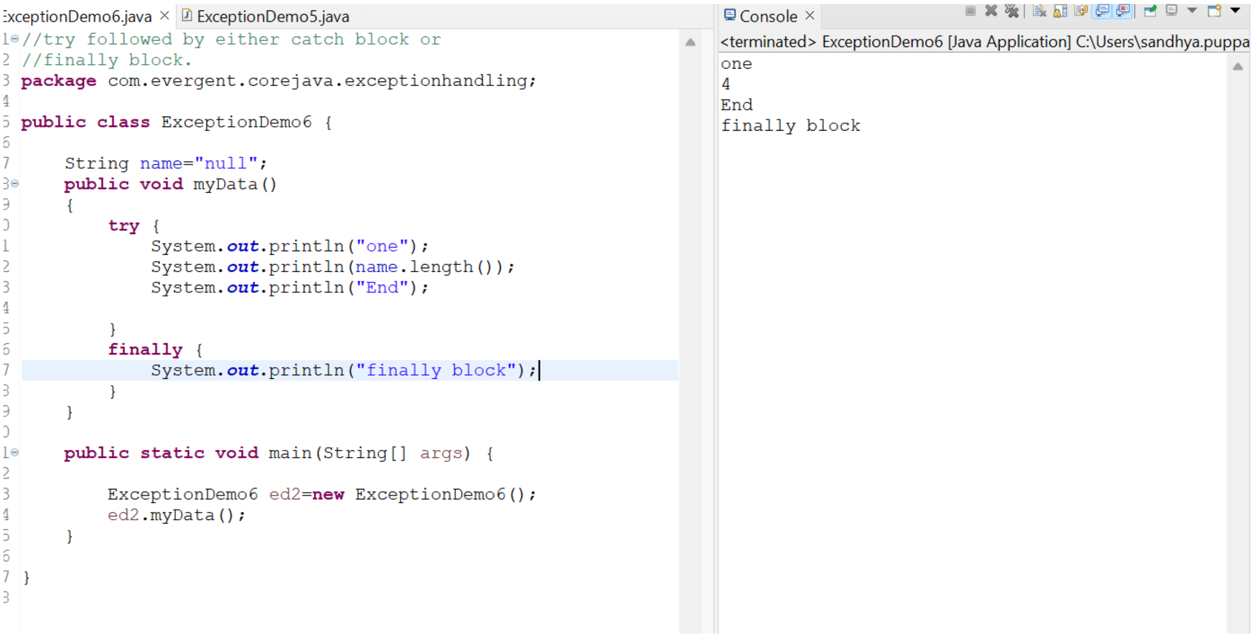


```java
//13)finally  is block,if exceptions id
//occurs or not,finally block will execute.
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo5 {
    String name="null";
    int k=2;
    public void myData()
    {
        try {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            System.out.println(t);
            System.out.println("end");
        }
        catch(NullPointerException e)
        {
            System.out.println("I can handle "+ e);
        }
        catch(ArithmeticException e)
        {
            System.out.println("I can handle :"+ e);
        }
        catch(Exception e)
        {
            System.out.println("I can handle :"+e);
        }
        finally {
            System.out.println("finally block close connection ")
        }
    }


    public static void main(String[] args) {
        ExceptionDemo5  ed3= new ExceptionDemo5();
        ed3.myData();

    }

}
```

```
<terminated> ExceptionDemo5 [Java Application] C:\Users\sandhya
one
4
5
end
finally block close connection
```

Program for 16 point

try followed by either catch block or finally block.

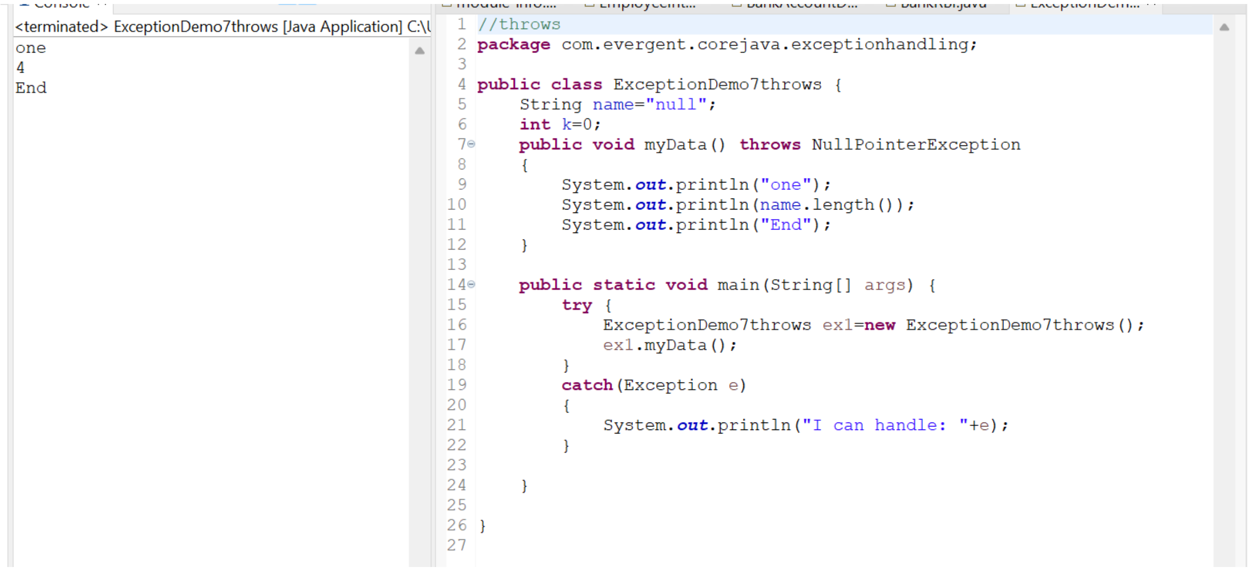


```java
//try followed by either catch block or
//finally block.
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo6 {

    String name="null";
    public void myData()
    {
        try {
            System.out.println("one");
            System.out.println(name.length());
            System.out.println("End");

        }
        finally {
            System.out.println("finally block");
        }
    }

    public static void main(String[] args) {

        ExceptionDemo6 ed2=new ExceptionDemo6();
        ed2.myData();
    }

}
```

```
<terminated> ExceptionDemo6 [Java Application] C:\Users\sandhya.puppa
one
4
End
finally block
```

Throws:

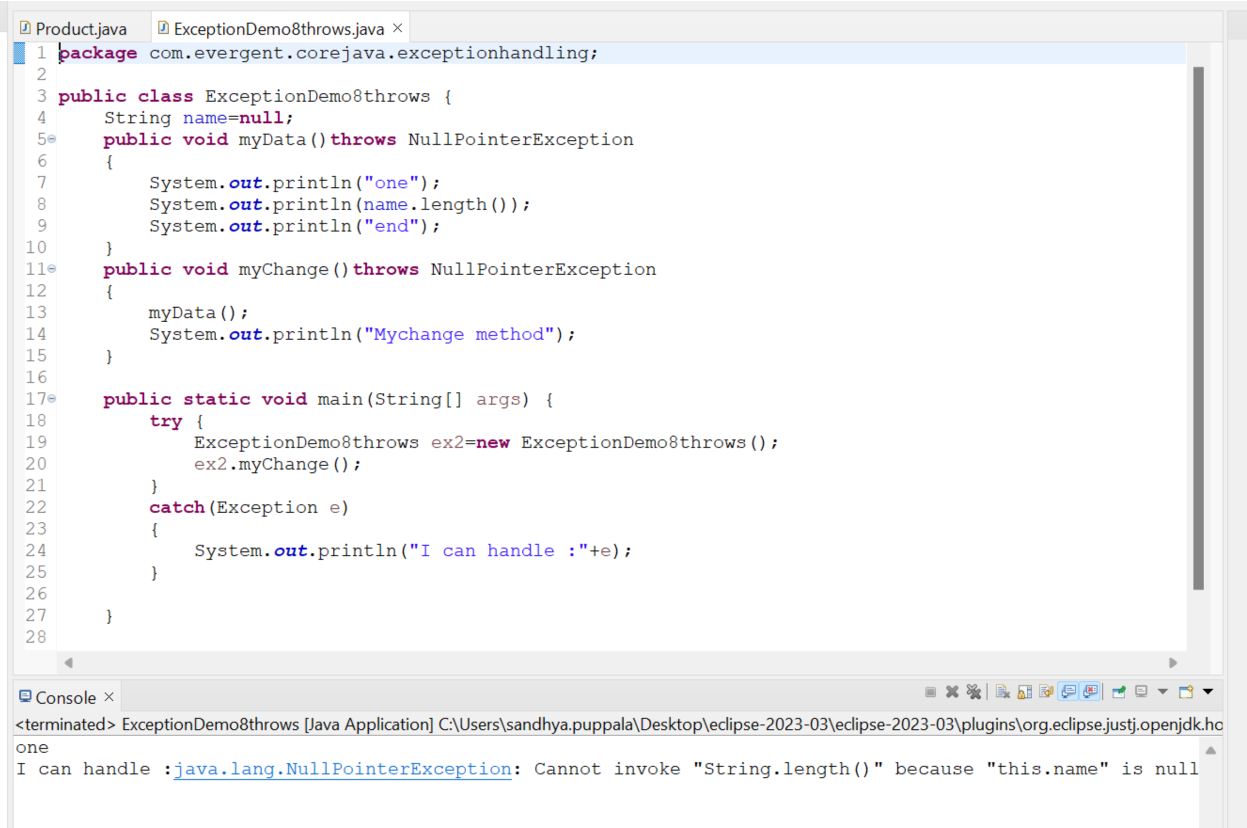
Program 1



```
<terminated> ExceptionDemo7throws [Java Application] C:\(
one
4
End
```

```java
//throws
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo7throws {
    String name="null";
    int k=0;
    public void myData() throws NullPointerException
    {
        System.out.println("one");
        System.out.println(name.length());
        System.out.println("End");
    }

    public static void main(String[] args) {
        try {
            ExceptionDemo7throws ex1=new ExceptionDemo7throws();
            ex1.myData();
        }
        catch(Exception e)
        {
            System.out.println("I can handle: "+e);
        }

    }

}
```

Program 2)

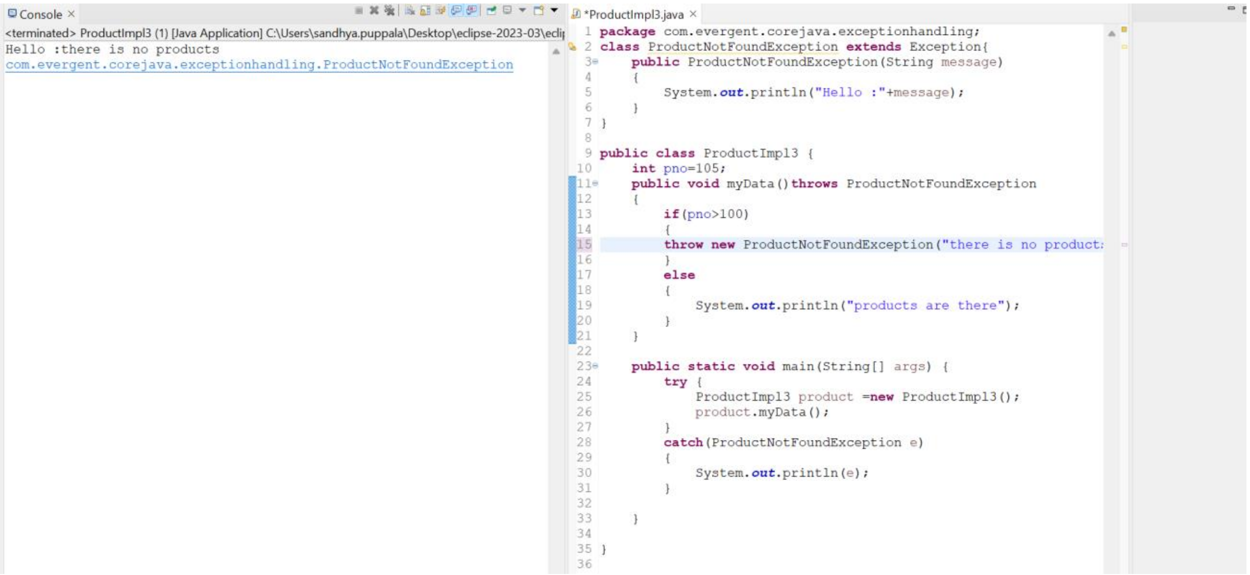


```java
package com.evergent.corejava.exceptionhandling;

public class ExceptionDemo8throws {
    String name=null;
    public void myData()throws NullPointerException
    {
        System.out.println("one");
        System.out.println(name.length());
        System.out.println("end");
    }
    public void myChange()throws NullPointerException
    {
        myData();
        System.out.println("Mychange method");
    }

    public static void main(String[] args) {
        try {
            ExceptionDemo8throws ex2=new ExceptionDemo8throws();
            ex2.myChange();
        }
        catch(Exception e)
        {
            System.out.println("I can handle :"+e);
        }

    }

```

```
<terminated> ExceptionDemo8throws [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.ho
one
I can handle :java.lang.NullPointerException: Cannot invoke "String.length()" because "this.name" is null
```

## Program 3)

## User defines exceptions

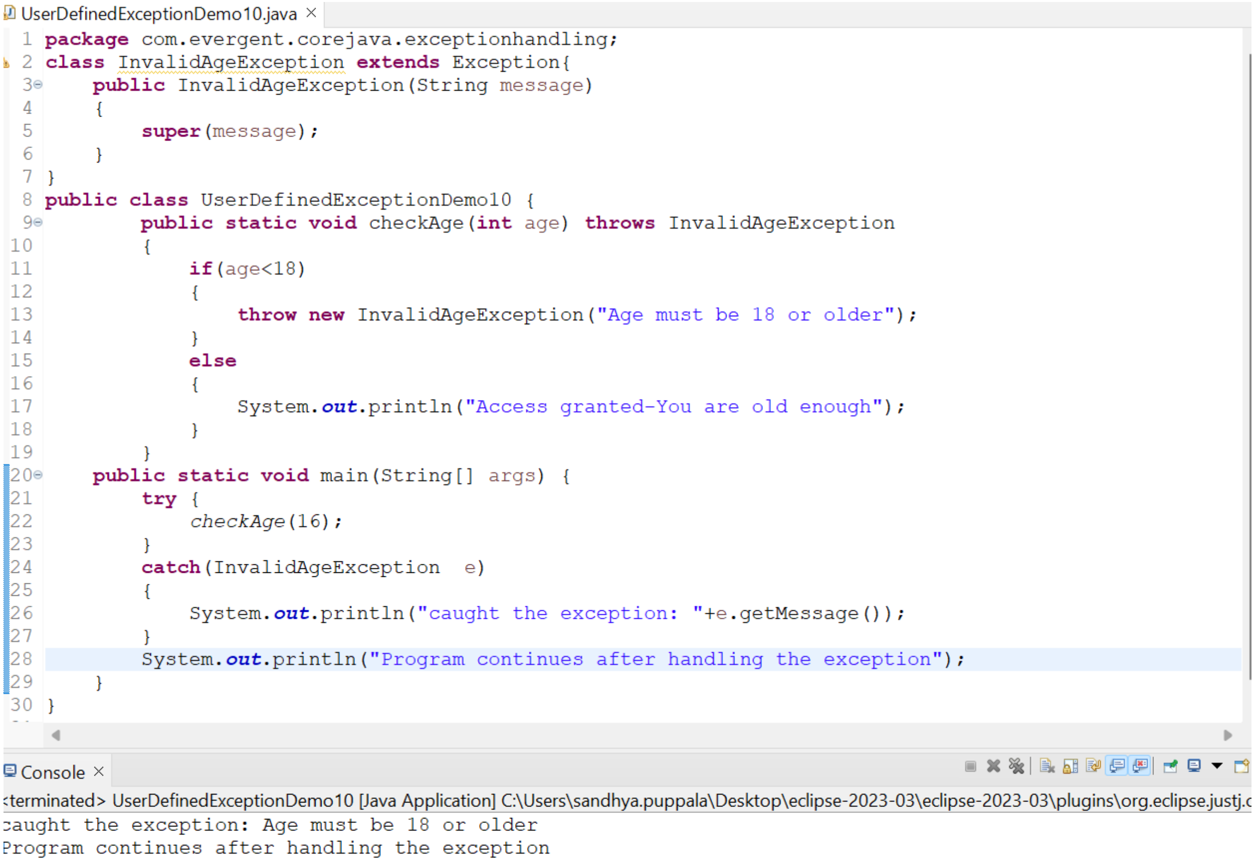


```
<terminated> ProductImpl3 (1) [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclip
Hello :there is no products
com.evergent.corejava.exceptionhandling.ProductNotFoundException
```

```java
package com.evergent.corejava.exceptionhandling;
class ProductNotFoundException extends Exception{
    public ProductNotFoundException(String message)
    {
        System.out.println("Hello :"+message);
    }
}

public class ProductImpl3 {
    int pno=105;
    public void myData()throws ProductNotFoundException
    {
        if(pno>100)
        {
        throw new ProductNotFoundException("there is no product:
        }
        else
        {
            System.out.println("products are there");
        }
    }

    public static void main(String[] args) {
        try {
            ProductImpl3 product =new ProductImpl3();
            product.myData();
        }
        catch(ProductNotFoundException e)
        {
            System.out.println(e);
        }

    }

}

```

Program4)

User defined exceptions using super keyword.

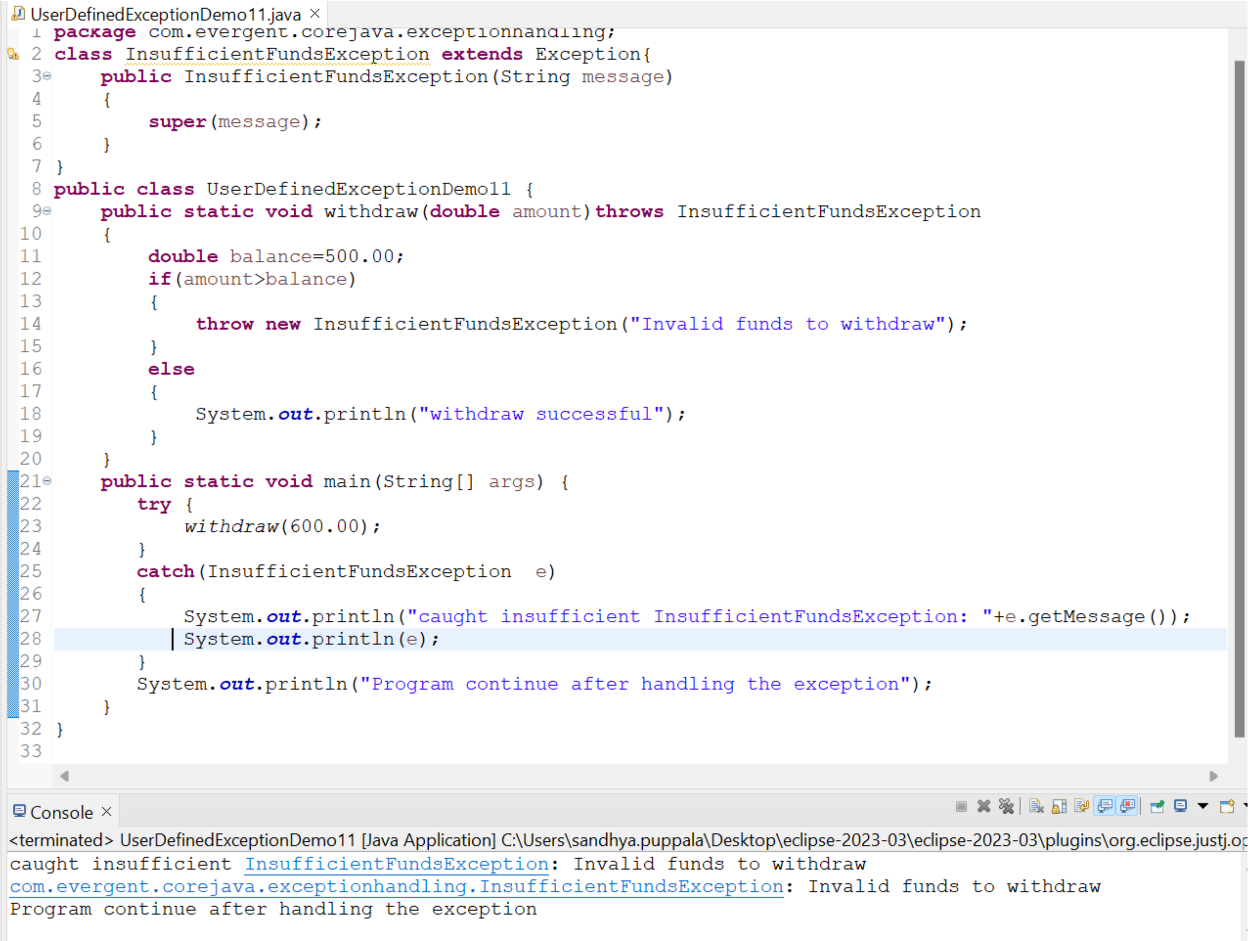


```java
package com.evergent.corejava.exceptionhandling;
class InvalidAgeException extends Exception{
    public InvalidAgeException(String message)
    {
        super(message);
    }
}
public class UserDefinedExceptionDemo10 {
        public static void checkAge(int age) throws InvalidAgeException
        {
            if(age<18)
            {
                throw new InvalidAgeException("Age must be 18 or older");
            }
            else
            {
                System.out.println("Access granted-You are old enough");
            }
        }
    public static void main(String[] args) {
        try {
            checkAge(16);
        }
        catch(InvalidAgeException  e)
        {
            System.out.println("caught the exception: "+e.getMessage());
        }
        System.out.println("Program continues after handling the exception");
    }
}
```

```
<terminated> UserDefinedExceptionDemo10 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.c
caught the exception: Age must be 18 or older
Program continues after handling the exception
```

Program 5)

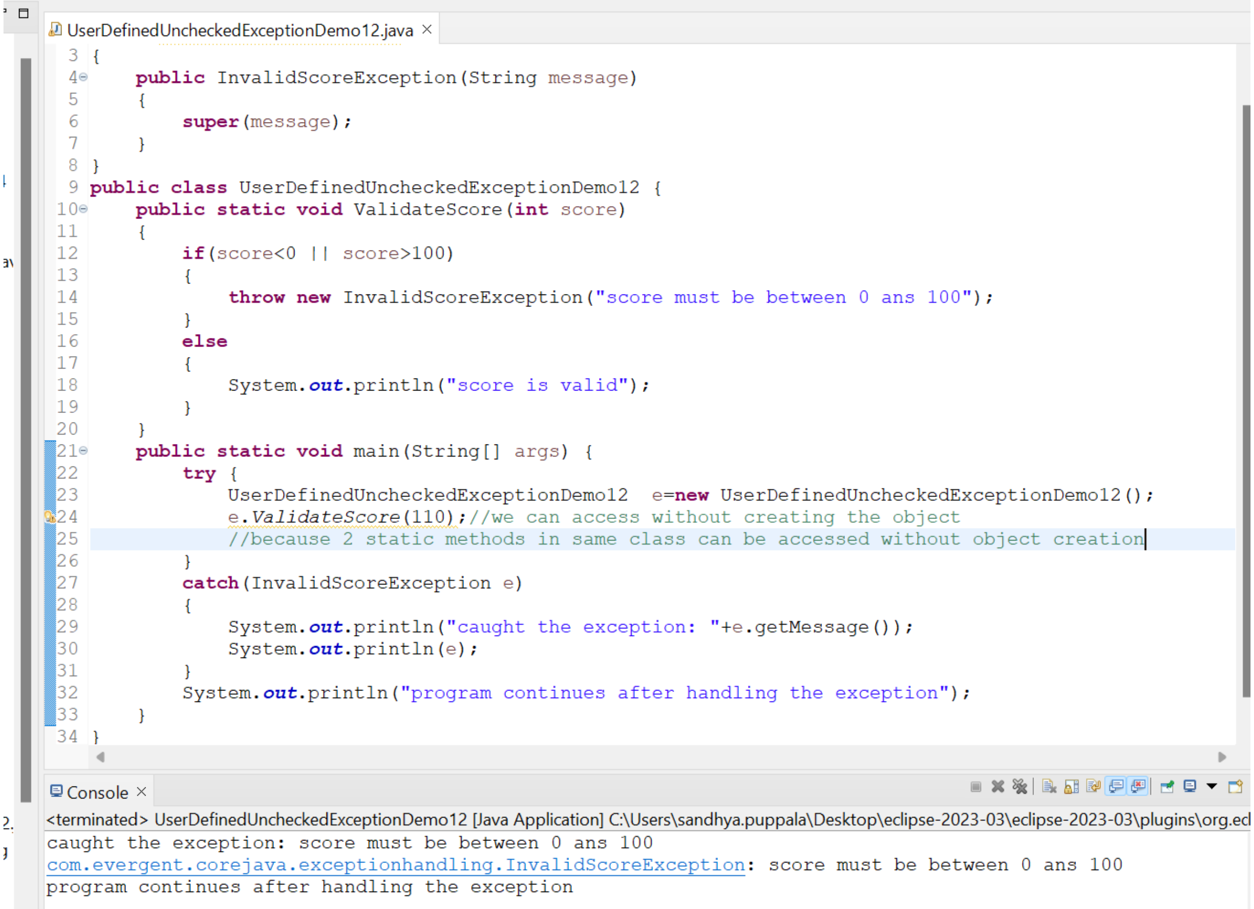
User defined exceptions using super keyword.



```java
package com.evergent.corejava.exceptionhandling;
class InsufficientFundsException extends Exception{
    public InsufficientFundsException(String message)
    {
        super(message);
    }
}
public class UserDefinedExceptionDemo11 {
    public static void withdraw(double amount)throws InsufficientFundsException
    {
        double balance=500.00;
        if(amount>balance)
        {
            throw new InsufficientFundsException("Invalid funds to withdraw");
        }
        else
        {
            System.out.println("withdraw successful");
        }
    }
    public static void main(String[] args) {
       try {
           withdraw(600.00);
       }
       catch(InsufficientFundsException  e)
       {
           System.out.println("caught insufficient InsufficientFundsException: "+e.getMessage());
           System.out.println(e);
       }
       System.out.println("Program continue after handling the exception");
    }
}
```

```
<terminated> UserDefinedExceptionDemo11 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.op
caught insufficient InsufficientFundsException: Invalid funds to withdraw
com.evergent.corejava.exceptionhandling.InsufficientFundsException: Invalid funds to withdraw
Program continue after handling the exception
```

Program 6)
Extends the Runtimeexception instead of Exception

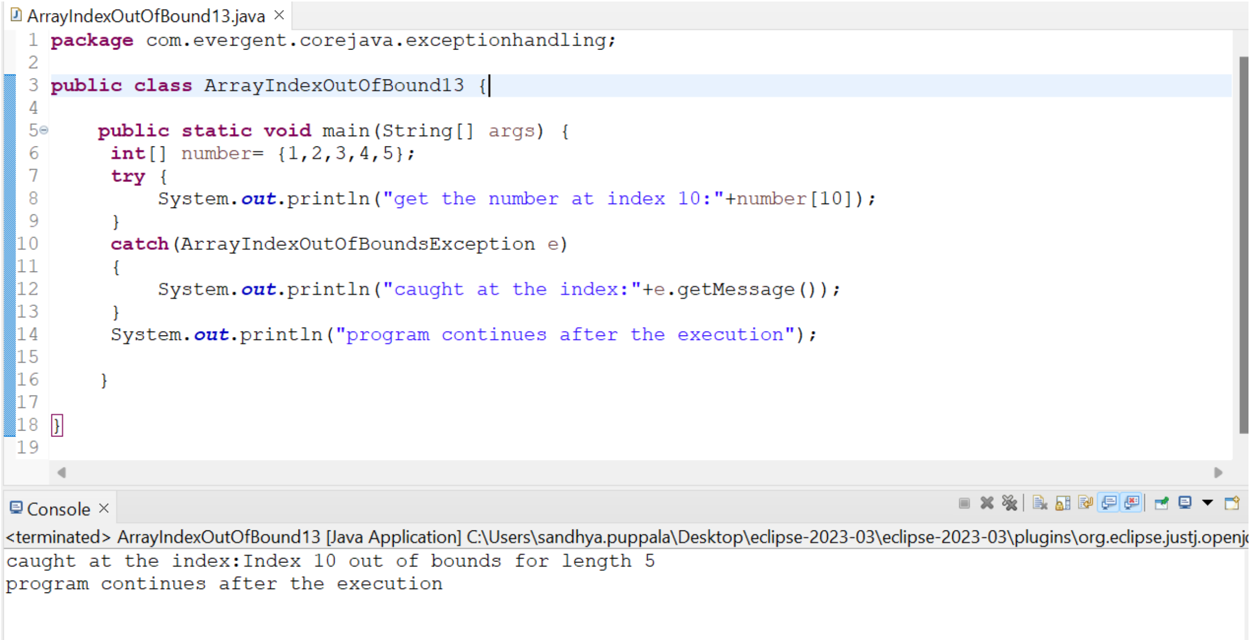


```java
{
    public InvalidScoreException(String message)
    {
        super(message);
    }
}
public class UserDefinedUncheckedExceptionDemo12 {
    public static void ValidateScore(int score)
    {
        if(score<0 || score>100)
        {
            throw new InvalidScoreException("score must be between 0 ans 100");
        }
        else
        {
            System.out.println("score is valid");
        }
    }
    public static void main(String[] args) {
        try {
            UserDefinedUncheckedExceptionDemo12  e=new UserDefinedUncheckedExceptionDemo12();
            e.ValidateScore(110);//we can access without creating the object
            //because 2 static methods in same class can be accessed without object creation
        }
        catch(InvalidScoreException e)
        {
            System.out.println("caught the exception: "+e.getMessage());
            System.out.println(e);
        }
        System.out.println("program continues after handling the exception");
    }
}
```

```
<terminated> UserDefinedUncheckedExceptionDemo12 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.ecl
caught the exception: score must be between 0 ans 100
com.evergent.corejava.exceptionhandling.InvalidScoreException: score must be between 0 ans 100
program continues after handling the exception
```

Program 7)

Program for ArrayIndexOutOfBounds Exception

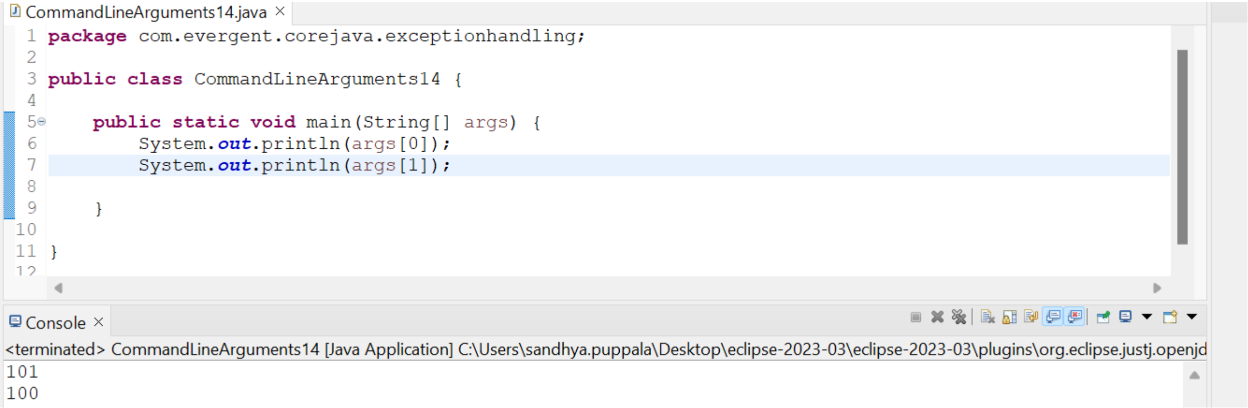


```java
package com.evergent.corejava.exceptionhandling;

public class ArrayIndexOutOfBound13 {

    public static void main(String[] args) {
     int[] number= {1,2,3,4,5};
     try {
         System.out.println("get the number at index 10:"+number[10]);
     }
     catch(ArrayIndexOutOfBoundsException e)
     {
         System.out.println("caught at the index:"+e.getMessage());
     }
     System.out.println("program continues after the execution");

    }

}

```

```
<terminated> ArrayIndexOutOfBound13 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openj
caught at the index:Index 10 out of bounds for length 5
program continues after the execution
```

## Program 8)

## Program for commandlineArguments

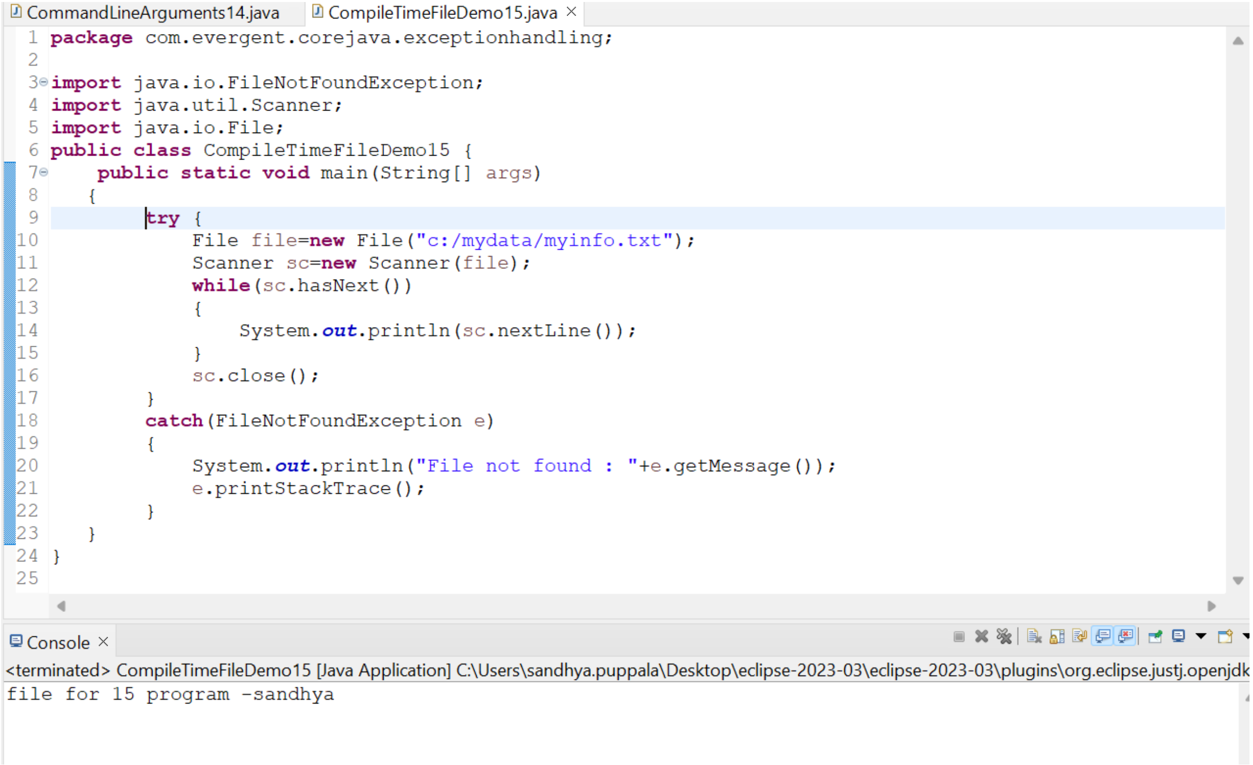


```java
package com.evergent.corejava.exceptionhandling;

public class CommandLineArguments14 {

    public static void main(String[] args) {
        System.out.println(args[0]);
        System.out.println(args[1]);

    }

}
```

```
<terminated> CommandLineArguments14 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjd
101
100
```

## Program 9)

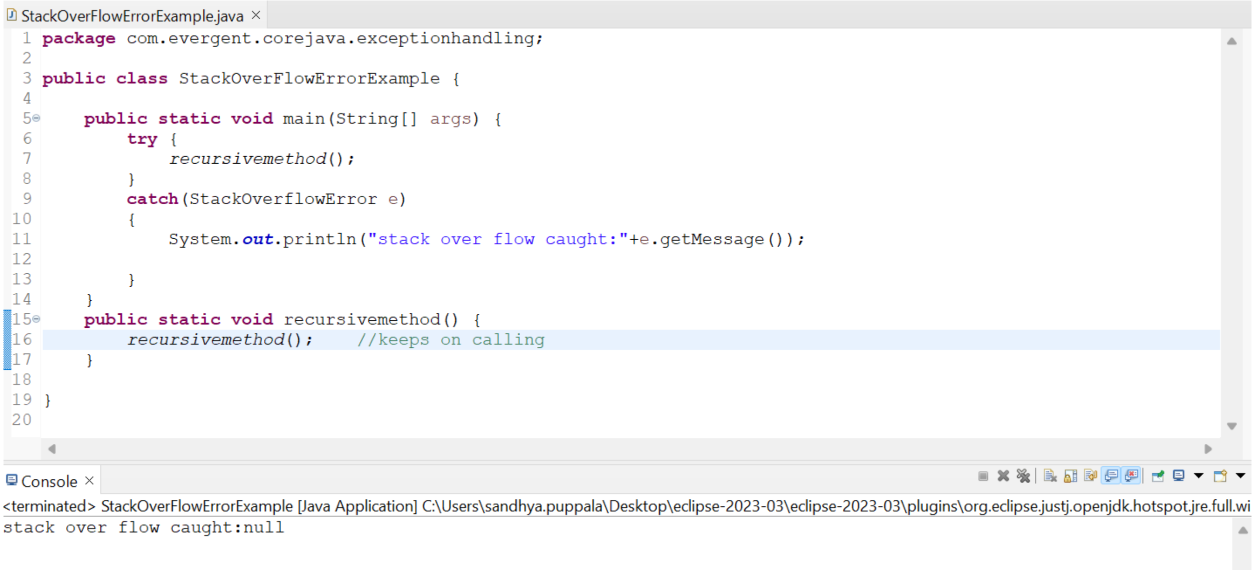
## Example program for FileNotFoundException



```java
package com.evergent.corejava.exceptionhandling;

import java.io.FileNotFoundException;
import java.util.Scanner;
import java.io.File;
public class CompileTimeFileDemo15 {
    public static void main(String[] args)
    {
        try {
            File file=new File("c:/mydata/myinfo.txt");
            Scanner sc=new Scanner(file);
            while(sc.hasNext())
            {
                System.out.println(sc.nextLine());
            }
            sc.close();
        }
        catch(FileNotFoundException e)
        {
            System.out.println("File not found : "+e.getMessage());
            e.printStackTrace();
        }
    }
}
```

```
<terminated> CompileTimeFileDemo15 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk
file for 15 program -sandhya
```

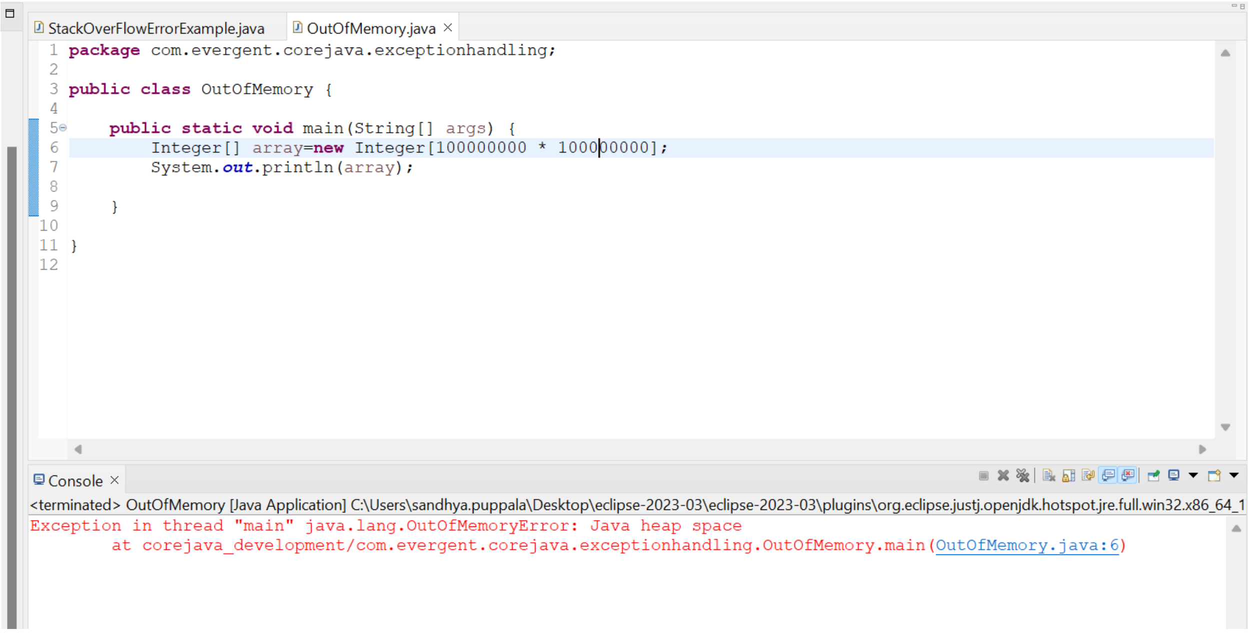
## Example program for StackOverFlow



```java
package com.evergent.corejava.exceptionhandling;

public class StackOverFlowErrorExample {

    public static void main(String[] args) {
        try {
            recursivemethod();
        }
        catch(StackOverflowError e)
        {
            System.out.println("stack over flow caught:"+e.getMessage());

        }
    }
    public static void recursivemethod() {
        recursivemethod();     //keeps on calling
    }

}
```

```
<terminated> StackOverFlowErrorExample [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.wi
stack over flow caught:null
```

## Example Program for OutOfMemory



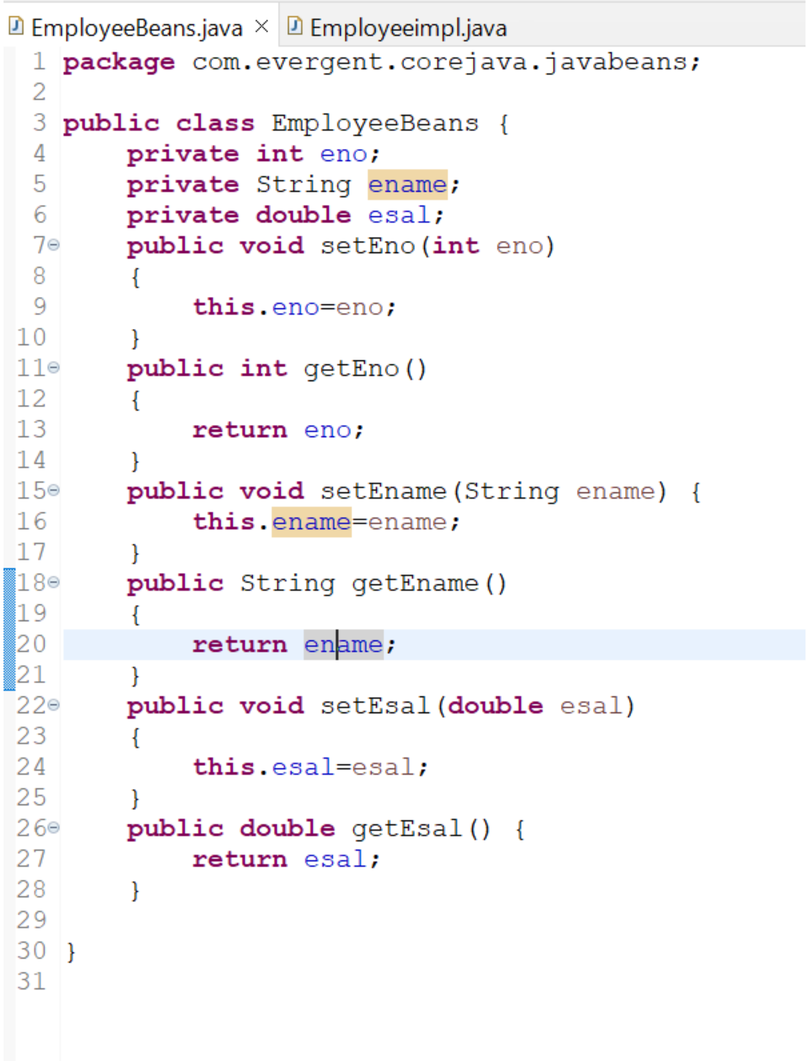
```java
package com.evergent.corejava.exceptionhandling;

public class OutOfMemory {

    public static void main(String[] args) {
        Integer[] array=new Integer[100000000 * 100000000];
        System.out.println(array);

    }

}
```

```
<terminated> OutOfMemory [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_1
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
	at corejava_development/com.evergent.corejava.exceptionhandling.OutOfMemory.main(OutOfMemory.java:6)
```

# JAVA BEANS

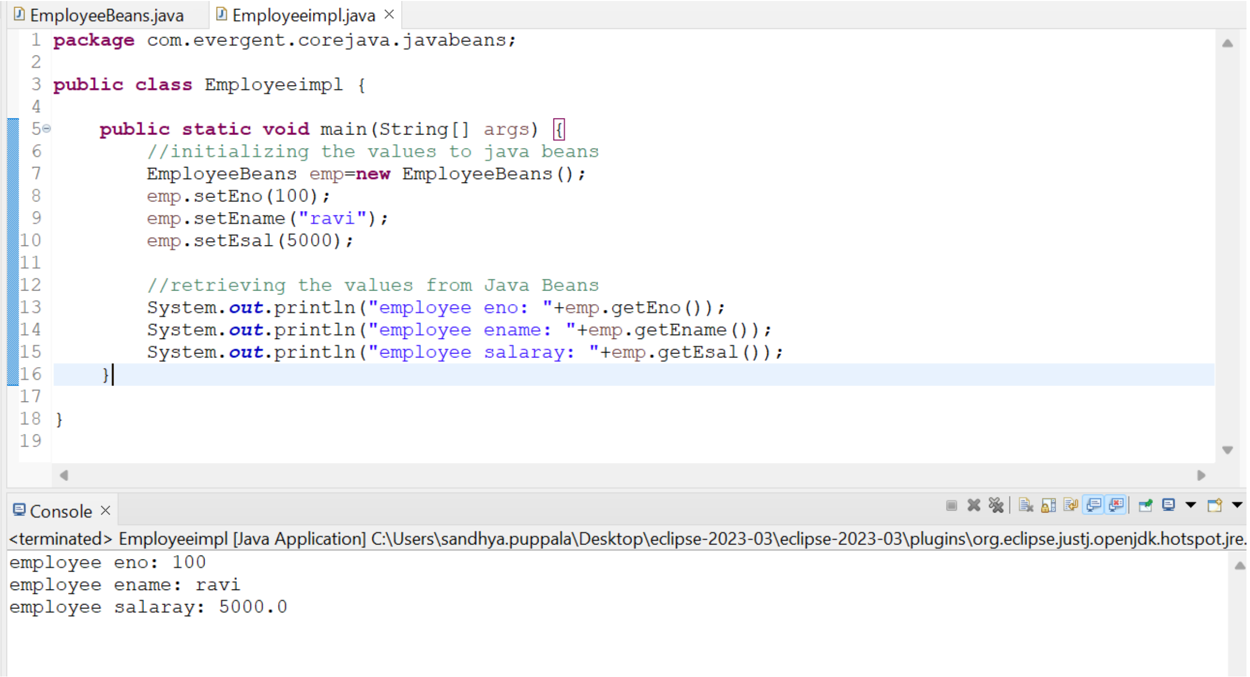
1) Java Bean is a mechanism.
2) Java Bean is light weight.
3) All attributes are private and get/set methods are public.
4) Implements java.serializable interface.
5) We cam achieve tightly encapsulation through Java Beans.

Program 1)



```
package com.evergent.corejava.javabeans;

public class EmployeeBeans {
    private int eno;
    private String ename;
    private double esal;
    public void setEno(int eno)
    {
        this.eno=eno;
    }
    public int getEno()
    {
        return eno;
    }
    public void setEname(String ename) {
        this.ename=ename;
    }
    public String getEname()
    {
        return ename;
    }
    public void setEsal(double esal)
    {
        this.esal=esal;
    }
    public double getEsal() {
        return esal;
    }

}
```

Implementation class

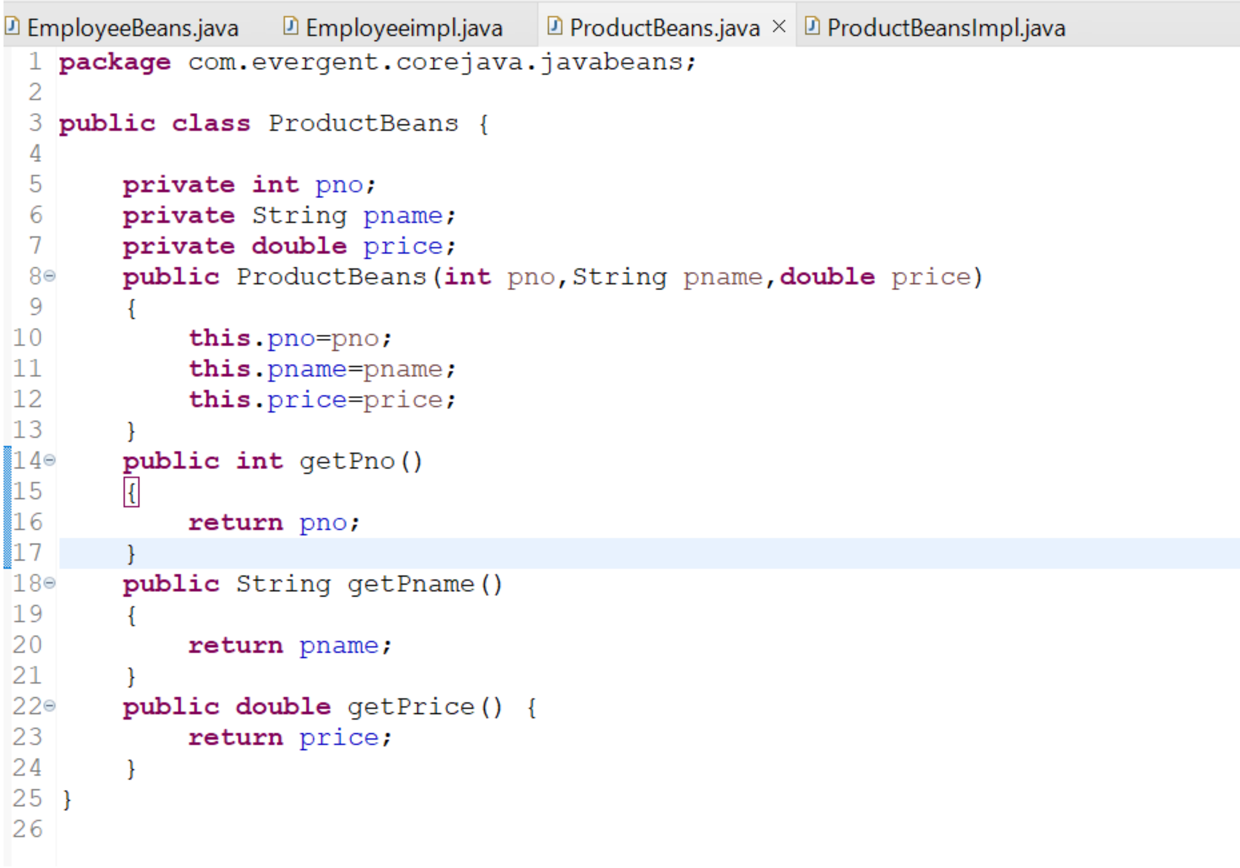


```java
package com.evergent.corejava.javabeans;

public class Employeeimpl {

    public static void main(String[] args) {
        //initializing the values to java beans
        EmployeeBeans emp=new EmployeeBeans();
        emp.setEno(100);
        emp.setEname("ravi");
        emp.setEsal(5000);

        //retrieving the values from Java Beans
        System.out.println("employee eno: "+emp.getEno());
        System.out.println("employee ename: "+emp.getEname());
        System.out.println("employee salaray: "+emp.getEsal());
    }

}
```

```
<terminated> Employeeimpl [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.
employee eno: 100
employee ename: ravi
employee salaray: 5000.0
```

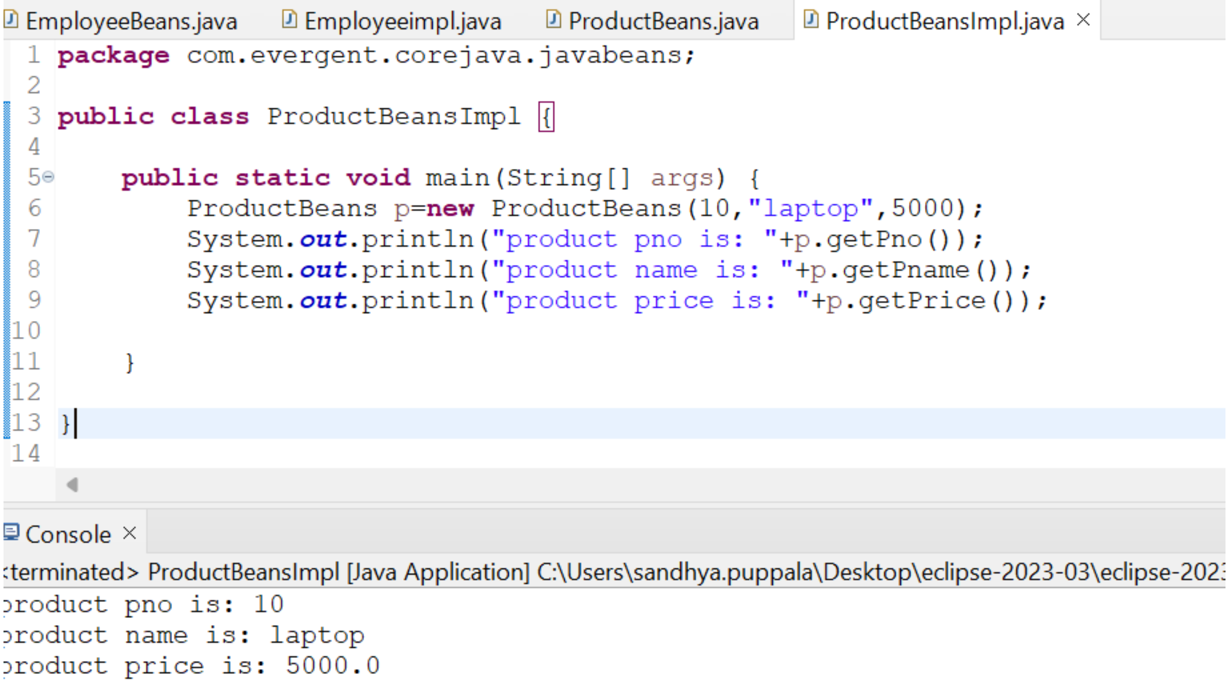
Program 2)

Program initializing the values with constructor and retrieving the values with java beans



```java
package com.evergent.corejava.javabeans;

public class ProductBeans {

    private int pno;
    private String pname;
    private double price;
    public ProductBeans(int pno,String pname,double price)
    {
        this.pno=pno;
        this.pname=pname;
        this.price=price;
    }
    public int getPno()
    {
        return pno;
    }
    public String getPname()
    {
        return pname;
    }
    public double getPrice() {
        return price;
    }
}

```

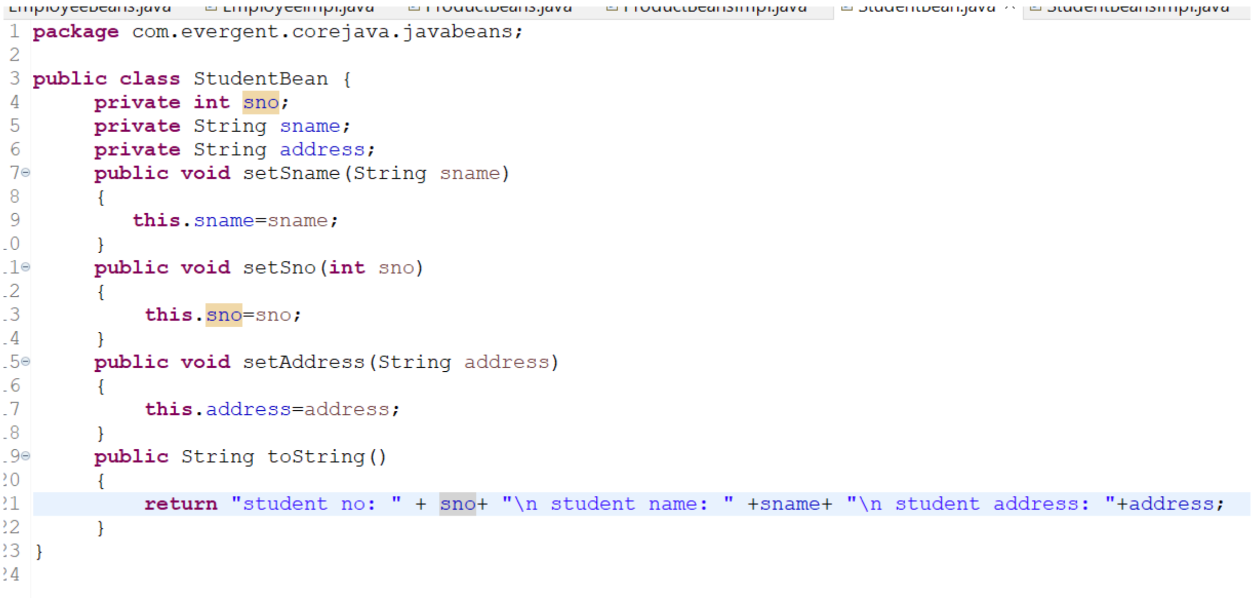
Implementation class



```java
package com.evergent.corejava.javabeans;

public class ProductBeansImpl {

    public static void main(String[] args) {
        ProductBeans p=new ProductBeans(10,"laptop",5000);
        System.out.println("product pno is: "+p.getPno());
        System.out.println("product name is: "+p.getPname());
        System.out.println("product price is: "+p.getPrice());

    }

}

```

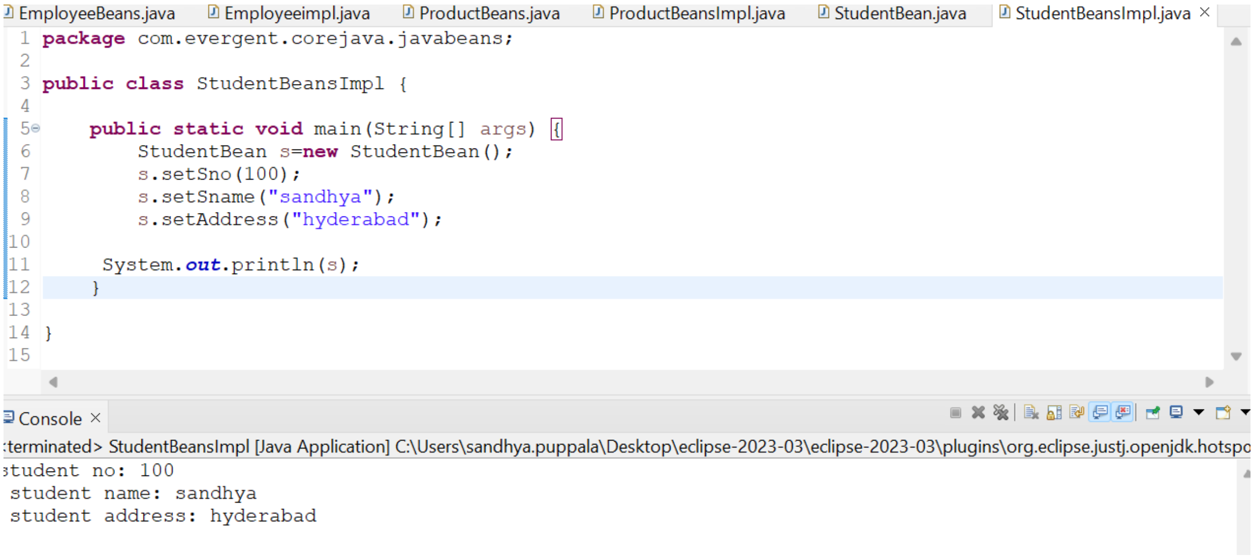
```
<terminated> ProductBeansImpl [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-202
product pno is: 10
product name is: laptop
product price is: 5000.0
```

Program 3)
Retrieving the values with over-riding the toString method



```
package com.evergent.corejava.javabeans;

public class StudentBean {
	private int sno;
	private String sname;
	private String address;
	public void setSname(String sname)
	{
	   this.sname=sname;
	}
	public void setSno(int sno)
	{
	    this.sno=sno;
	}
	public void setAddress(String address)
	{
	    this.address=address;
	}
	public String toString()
	{
	    return "student no: " + sno+ "\n student name: " +sname+ "\n student address: "+address;
	}
}
```

Implementation class

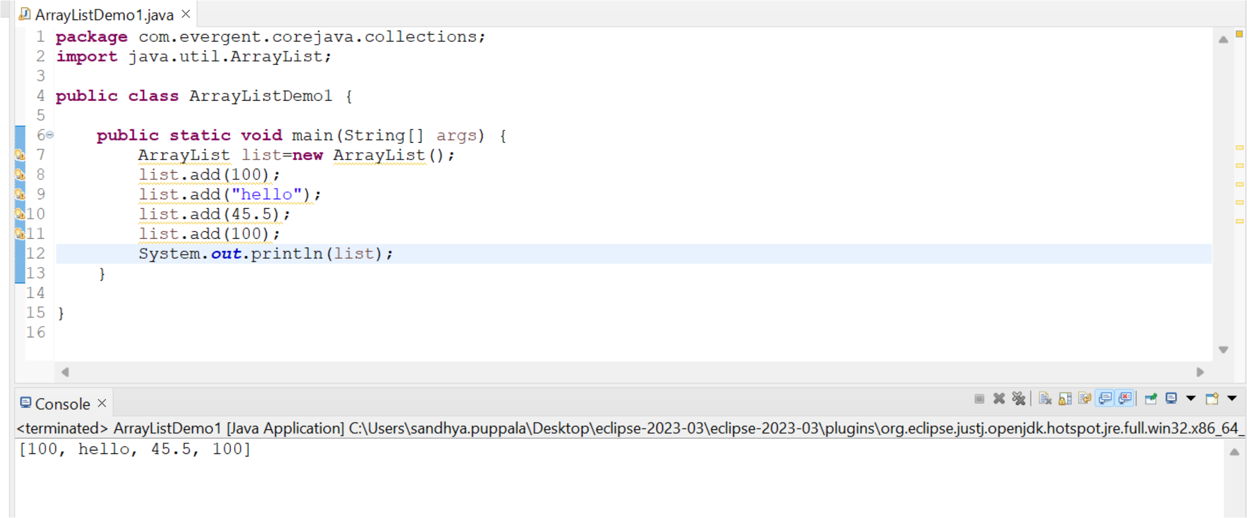


```
package com.evergent.corejava.javabeans;

public class StudentBeansImpl {

	public static void main(String[] args) {
		StudentBean s=new StudentBean();
		s.setSno(100);
		s.setSname("sandhya");
		s.setAddress("hyderabad");

	 System.out.println(s);
	}

}
```

```
<terminated> StudentBeansImpl [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspo
student no: 100
 student name: sandhya
 student address: hyderabad
```

# Collections Framework

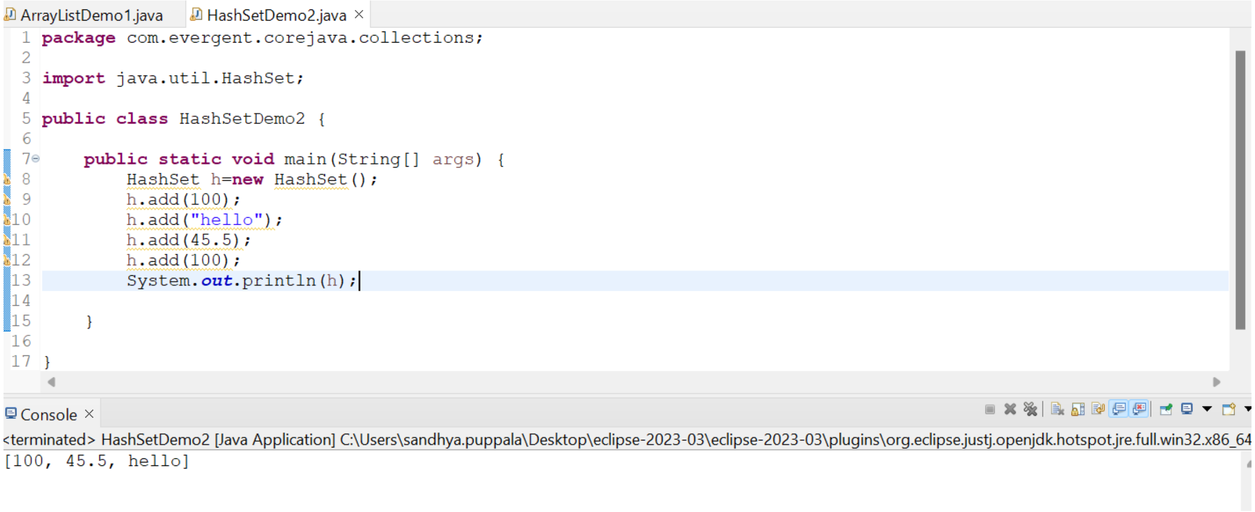
## ArrayList



```java
package com.evergent.corejava.collections;
import java.util.ArrayList;

public class ArrayListDemo1 {

    public static void main(String[] args) {
        ArrayList list=new ArrayList();
        list.add(100);
        list.add("hello");
        list.add(45.5);
        list.add(100);
        System.out.println(list);
    }

}

```

```
<terminated> ArrayListDemo1 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
[100, hello, 45.5, 100]
```

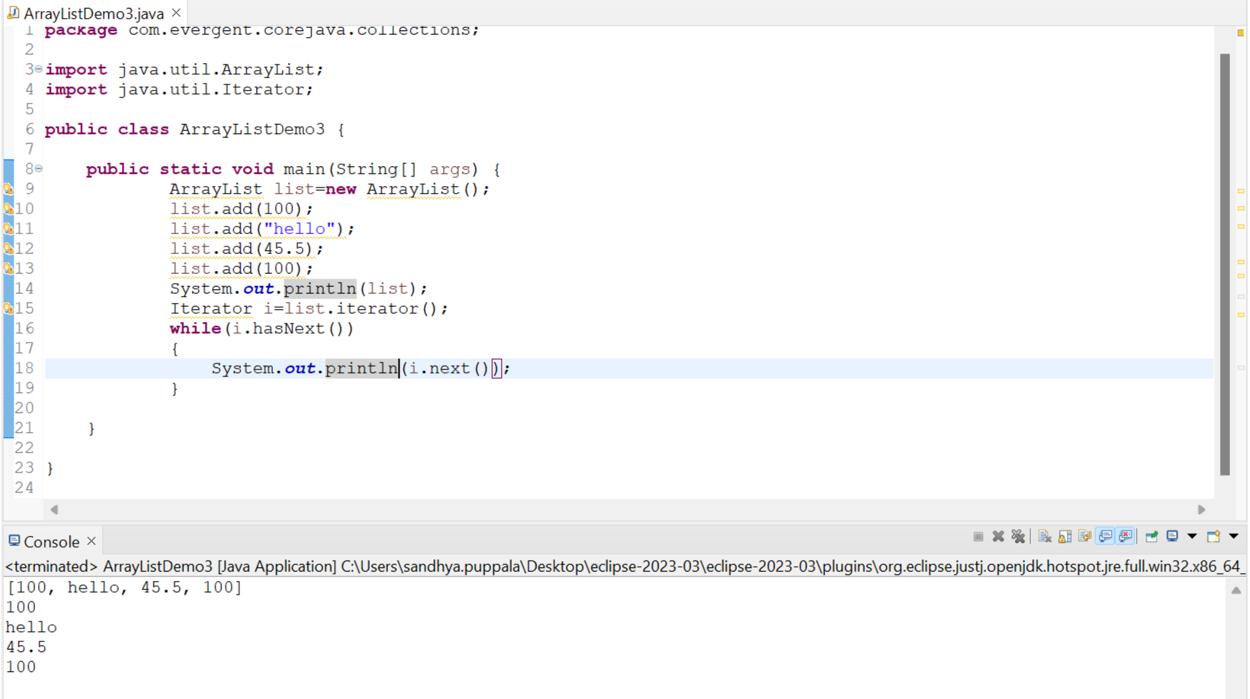
## HashSet



```java
package com.evergent.corejava.collections;

import java.util.HashSet;

public class HashSetDemo2 {

    public static void main(String[] args) {
        HashSet h=new HashSet();
        h.add(100);
        h.add("hello");
        h.add(45.5);
        h.add(100);
        System.out.println(h);

    }

}
```

```
<terminated> HashSetDemo2 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
[100, 45.5, hello]
```

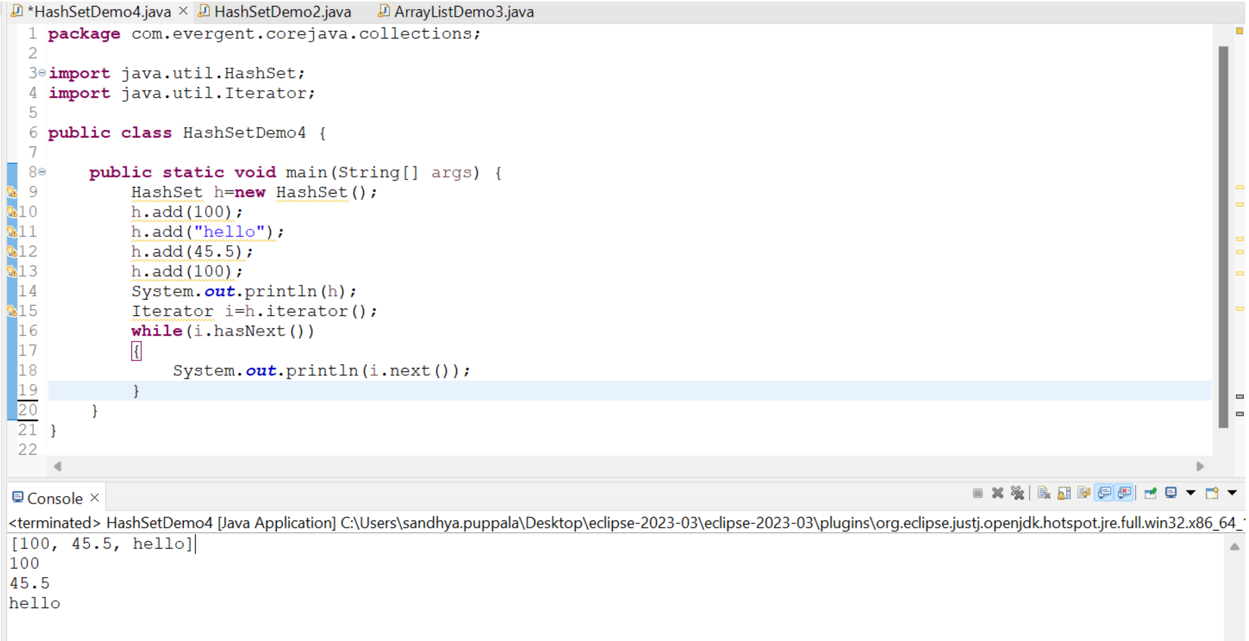
## hasNext() program



```java
package com.evergent.corejava.collections;

import java.util.ArrayList;
import java.util.Iterator;

public class ArrayListDemo3 {

    public static void main(String[] args) {
            ArrayList list=new ArrayList();
            list.add(100);
            list.add("hello");
            list.add(45.5);
            list.add(100);
            System.out.println(list);
            Iterator i=list.iterator();
            while(i.hasNext())
            {
                System.out.println(i.next());
            }

    }

}
```

```
<terminated> ArrayListDemo3 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
[100, hello, 45.5, 100]
100
hello
45.5
100
```

Program 4)

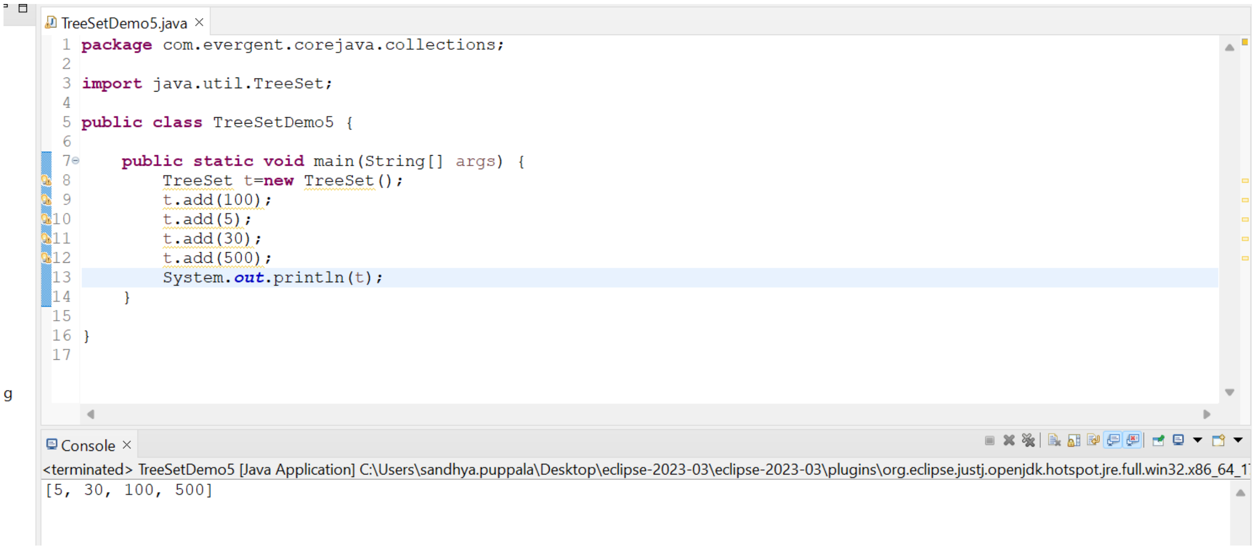


```java
package com.evergent.corejava.collections;

import java.util.HashSet;
import java.util.Iterator;

public class HashSetDemo4 {

    public static void main(String[] args) {
        HashSet h=new HashSet();
        h.add(100);
        h.add("hello");
        h.add(45.5);
        h.add(100);
        System.out.println(h);
        Iterator i=h.iterator();
        while(i.hasNext())
        {
            System.out.println(i.next());
        }
    }
}
```

```
<terminated> HashSetDemo4 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
[100, 45.5, hello]
100
45.5
hello
```

Program 5)

TreeSet is for Ordering(Ascending order)



```
package com.evergent.corejava.collections;

import java.util.TreeSet;

public class TreeSetDemo5 {

    public static void main(String[] args) {
        TreeSet t=new TreeSet();
        t.add(100);
        t.add(5);
        t.add(30);
        t.add(500);
        System.out.println(t);
    }

}

```

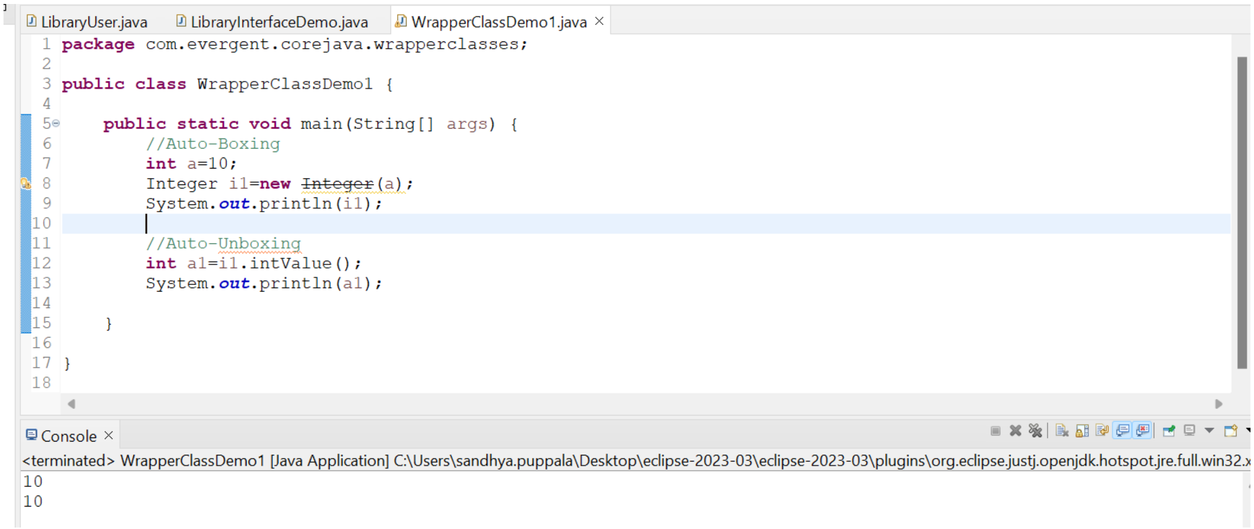
```
<terminated> TreeSetDemo5 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_1
[5, 30, 100, 500]
```

Day 14

# Wrapper Classes

Auto Boxing ans Auto Unboxing

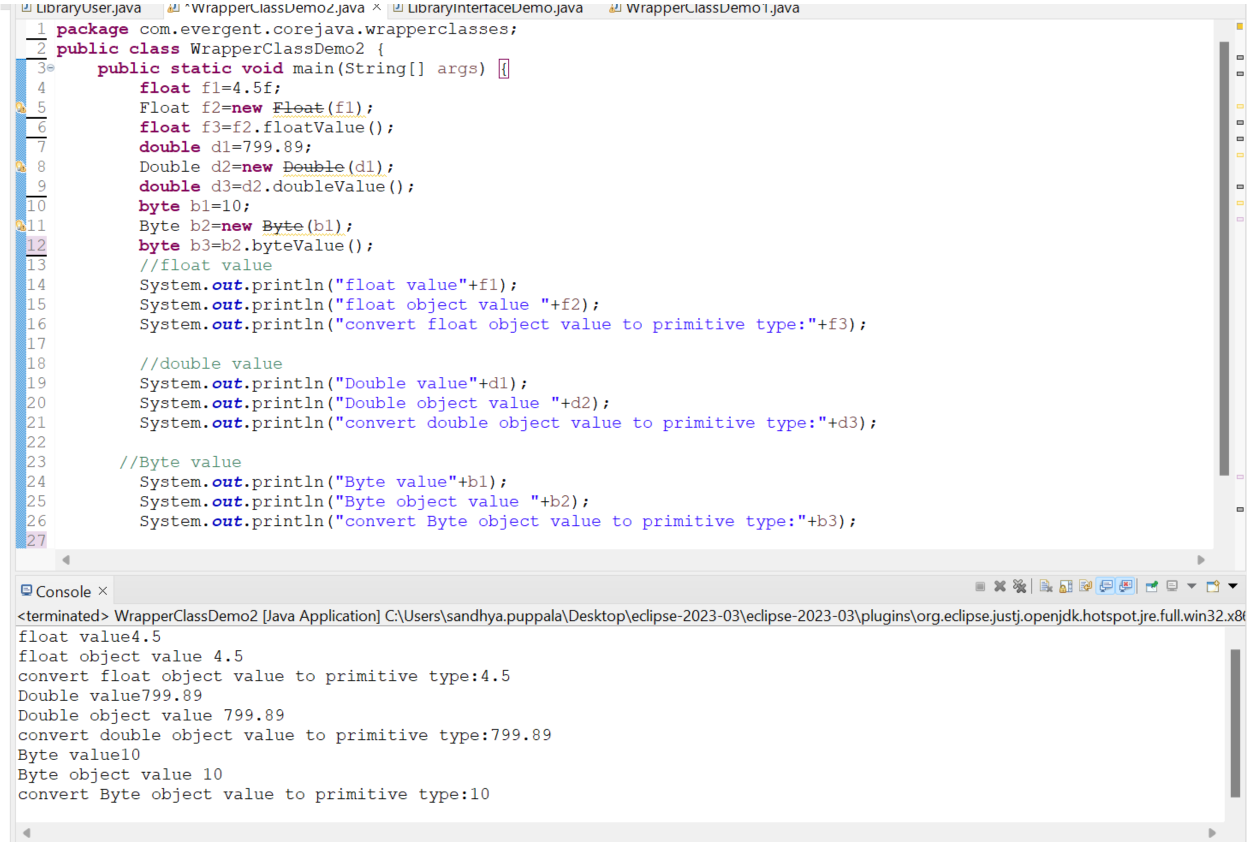
1)program



```
package com.evergent.corejava.wrapperclasses;

public class WrapperClassDemo1 {

    public static void main(String[] args) {
        //Auto-Boxing
        int a=10;
        Integer i1=new ~~Integer~~(a);
        System.out.println(i1);

        //Auto-Unboxing
        int a1=i1.intValue();
        System.out.println(a1);

    }

}

```

```
<terminated> WrapperClassDemo1 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x
10
10
```

2)  Program



```java
package com.evergent.corejava.wrapperclasses;
public class WrapperClassDemo2 {
    public static void main(String[] args) {
        float f1=4.5f;
        Float f2=new ~~Float~~(f1);
        float f3=f2.floatValue();
        double d1=799.89;
        Double d2=new ~~Double~~(d1);
        double d3=d2.doubleValue();
        byte b1=10;
        Byte b2=new ~~Byte~~(b1);
        byte b3=b2.byteValue();
        //float value
        System.out.println("float value"+f1);
        System.out.println("float object value "+f2);
        System.out.println("convert float object value to primitive type:"+f3);

        //double value
        System.out.println("Double value"+d1);
        System.out.println("Double object value "+d2);
        System.out.println("convert double object value to primitive type:"+d3);

      //Byte value
        System.out.println("Byte value"+b1);
        System.out.println("Byte object value "+b2);
        System.out.println("convert Byte object value to primitive type:"+b3);

```

<terminated> WrapperClassDemo2 [Java Application] C:\Users\sandhya.puppala\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x8

```
float value4.5
float object value 4.5
convert float object value to primitive type:4.5
Double value799.89
Double object value 799.89
convert double object value to primitive type:799.89
Byte value10
Byte object value 10
convert Byte object value to primitive type:10
```