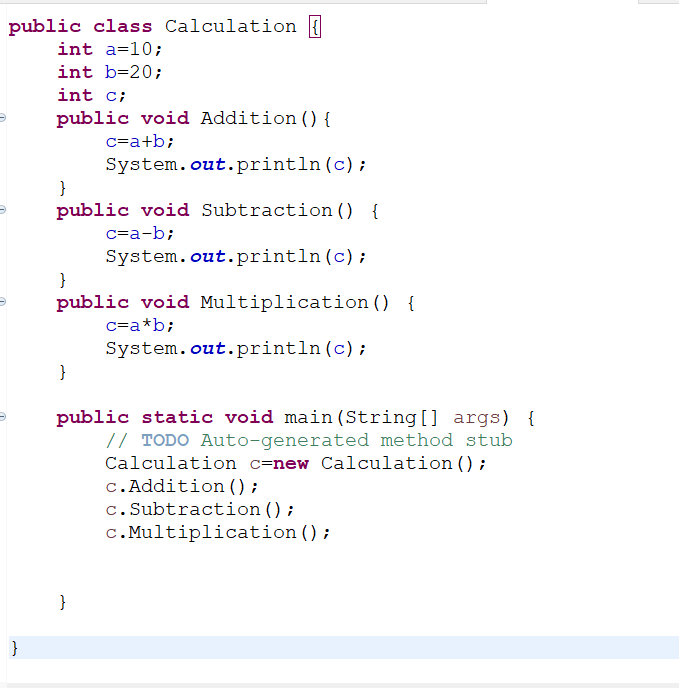
## OOPS(Object Oriented Programing System)

Java is a High Level Language .

The Four Main Pillars Of Java is :

1. Encapsulation:Encapsulations is a Mechanism Binding Attributes and Methods Together Inside the class And Object Creation is called Encapsulation.

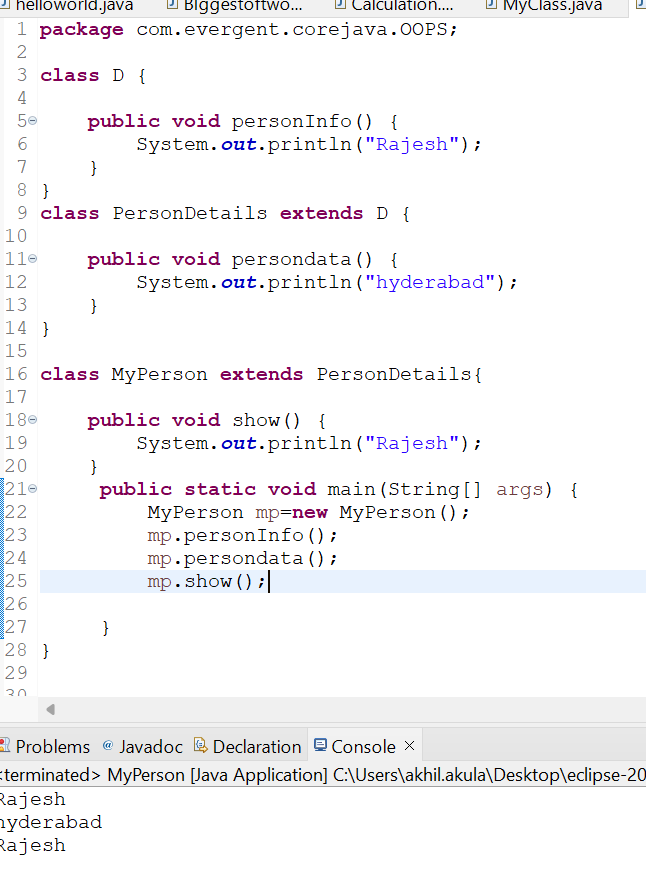
Example :Cpasule,Office Bag



1. Inheritance: Reusability of Existing Functionalities from super class to sub class.Types of Inheritance ;
2. Single Level Inheritance
3. Multi level Inheritance
4. Hierarchical Inheritance
5. Mutiple Inheritance
6. Hybrid Inheritance

* Java Does not Support Multiple Inheritance via class Because of Abiguity.
* **Class**-Class is Blueprint Or Templete for Creating the Objects
* **Objects**-Objects are the Instance of the class it has both state and behaviour

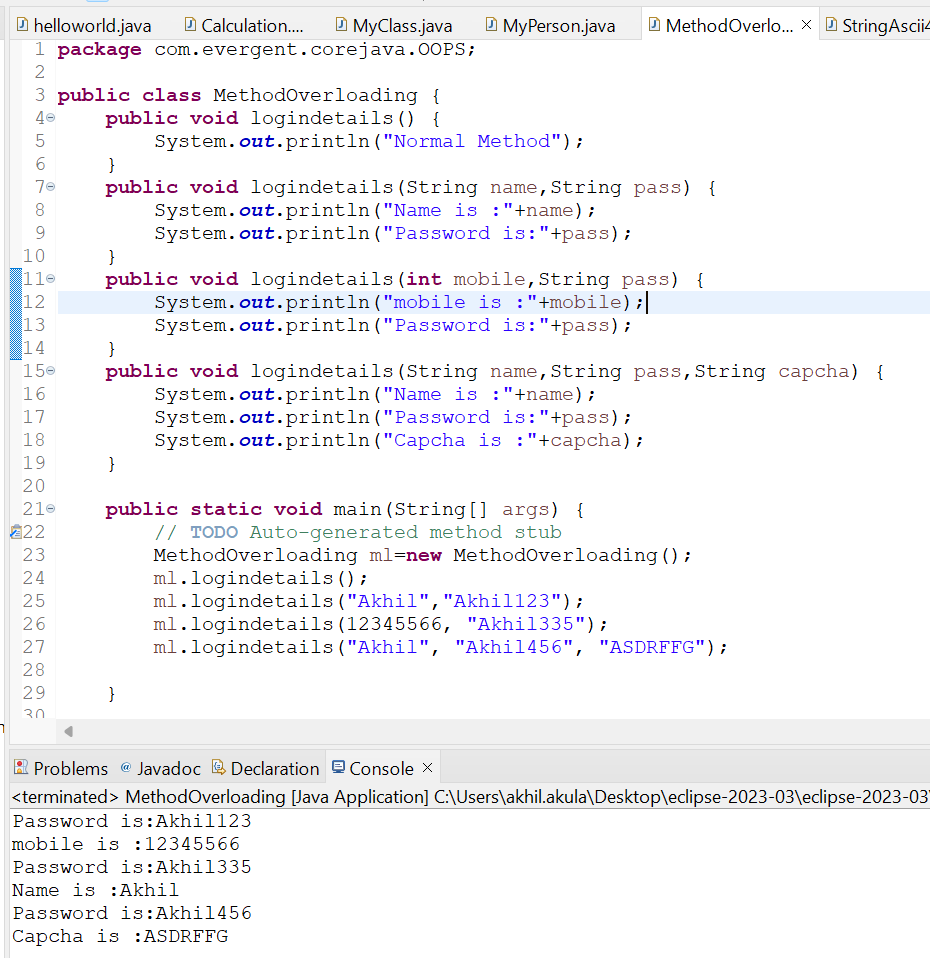
Programs On Inheritance:



## **PoLymorphism:**Poly Means many and Morphism means Forms Performing the one Task in Many Forms called Polymorphism.

###### Method OverLoading:OverLoading Means Method names are Same But Parameters Should be Different ,the return type may or may not be same it will happen in same class Or Different class called Method OverLoading also called as Compile Time Polymorphsim.

###### Progarms:



###### MethodOverriding : Overriding means Method names are same,parameters are same ,same return type it happen in two different classes Through Inheritance .

##### Programs:

##### 

Constructor’s

###### Constructors are mainly for Intialization

###### Class name and Constructors name are same

###### Constructors are two types:

###### Default Constructor

###### Parameterized Constructor

###### Constructors Does not have return types not even void if we declare as void it

###### will consider as Method not Constructor

###### 5.Constructors are OverLoaded

###### 6.Every class has atleast One default Constructor Either you create a Constructor or compiler will generate its own constructor

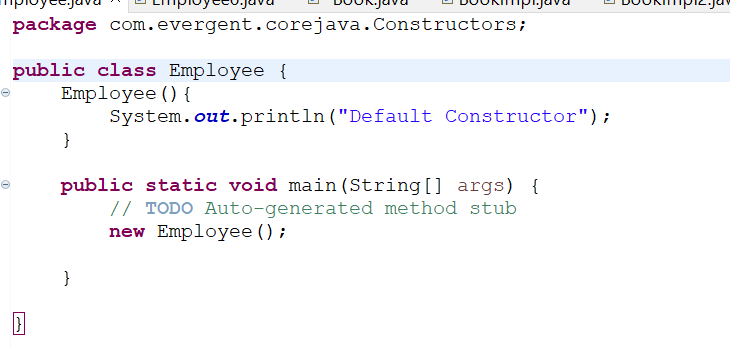
###### 7.This is a keyword always point to Instance/global Variable

###### 8.By This keyword we can call One Constructor through other constructor via this keyword

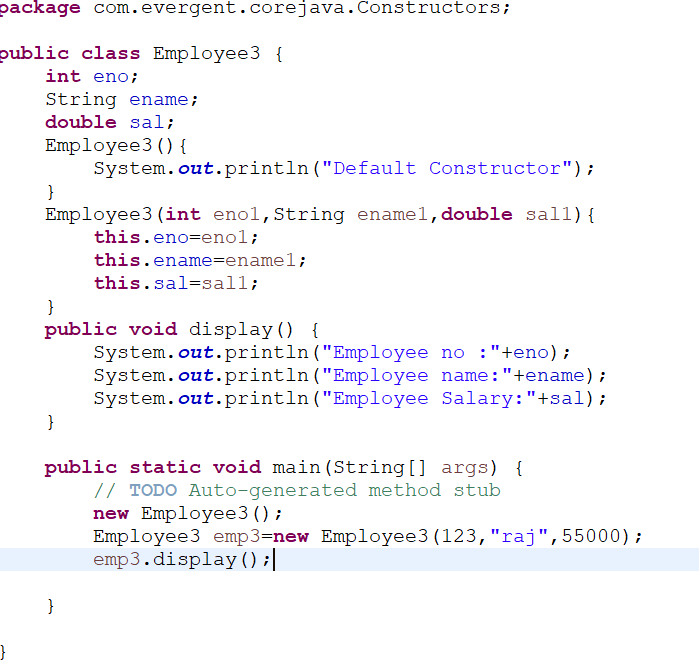
###### 9.super is a keyword used to call super class Constructor as well as super class Methods

###### 10.Copy Constructor ,Copy constructor means Copy’s object reference

###### Example :Basic Program on Constructor



Example :program on Default and parameterizes Constructor



Example :Progarm on this keyword



Example :Program on Super keyword



# Static Keyword

###### Staic is a keyword

###### We can Declare static as Method and variable

###### We can Acess Static methods and static variables Through classname.methodname,classname.variablename

###### Static methods can Acess Static Methods and Static variables

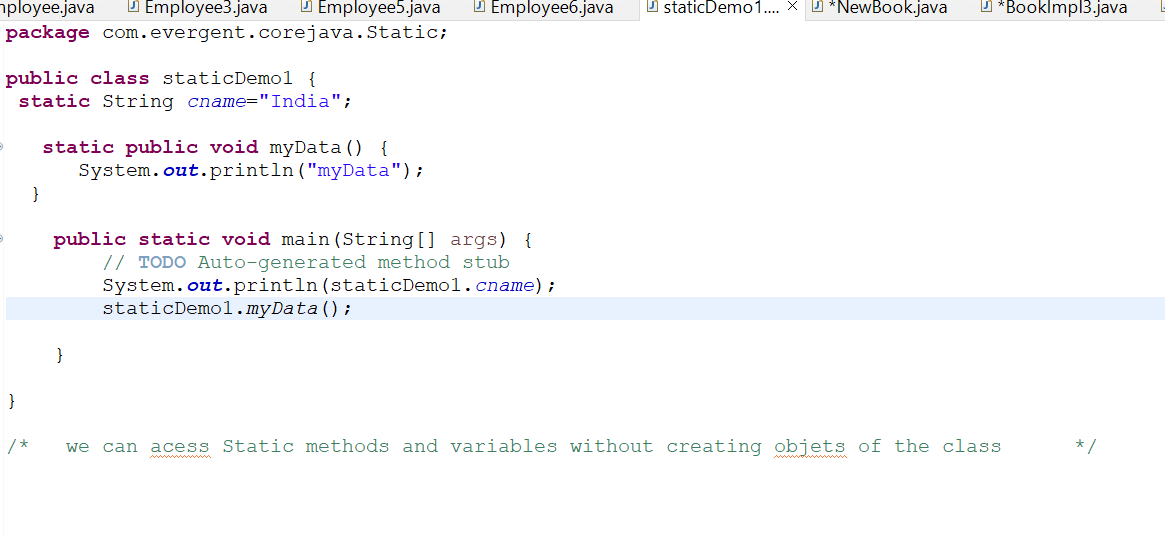
###### Static methods cannot Acess Non-Static Methods and Non-Static Variables

###### Non-static methods can Acess Static methods and Static variables

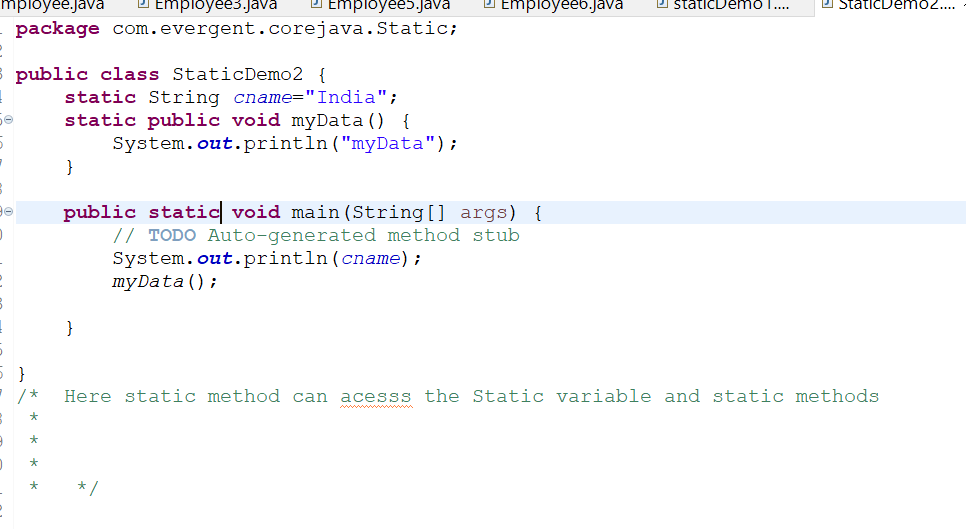
###### Static Block ,Static Block is Executed before the Main Method

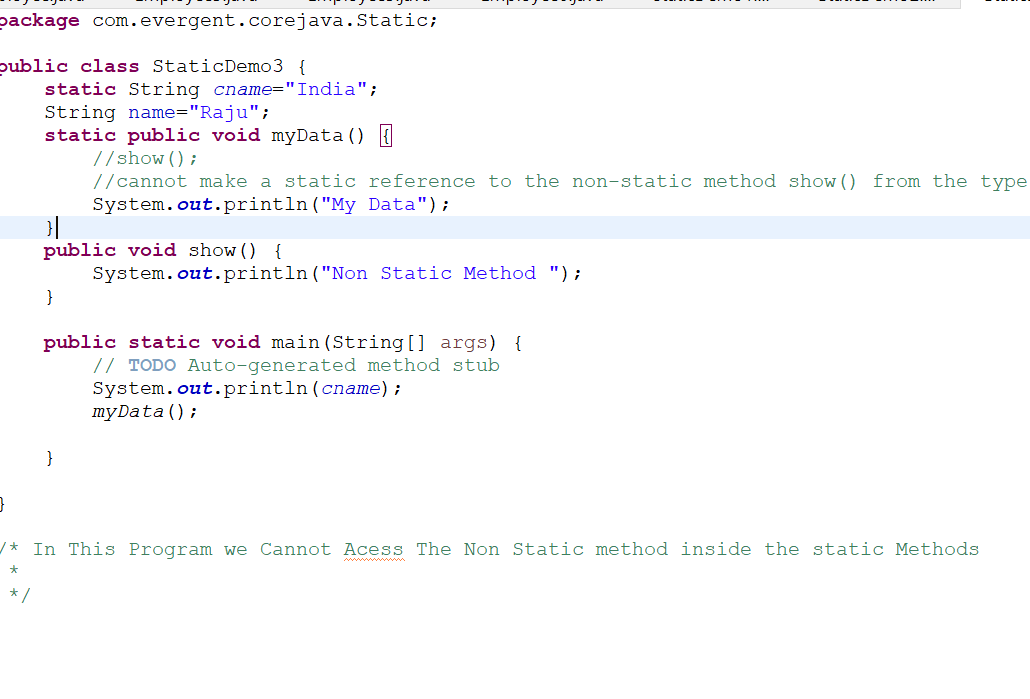
###### Static Methods And Static variables are called without any object creation

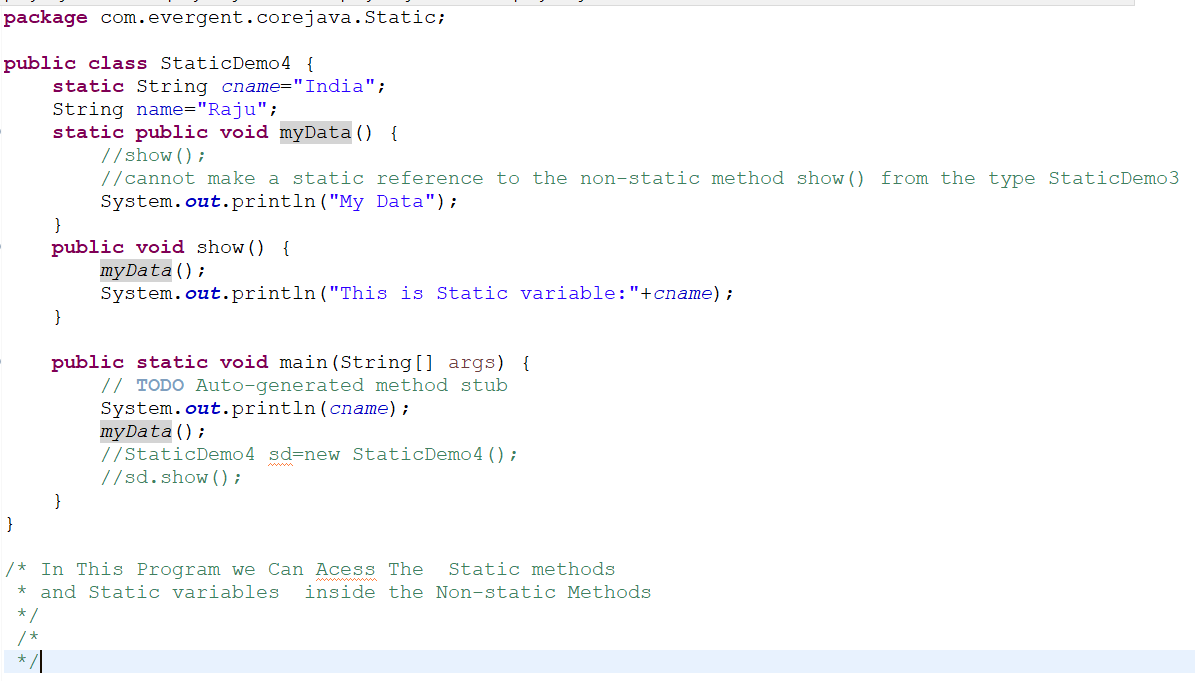
Example :Basic Program on Static Keyword

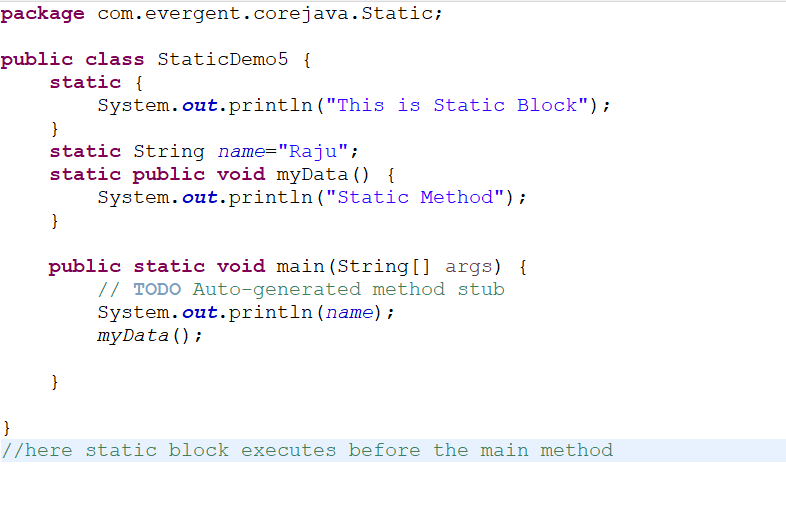


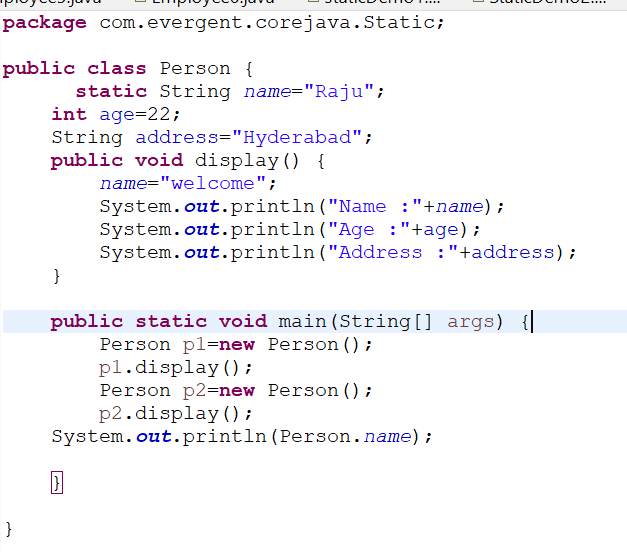
Example: without any class name we are calling method and variables of static











## Final Keyword

###### Final is a KeyWord

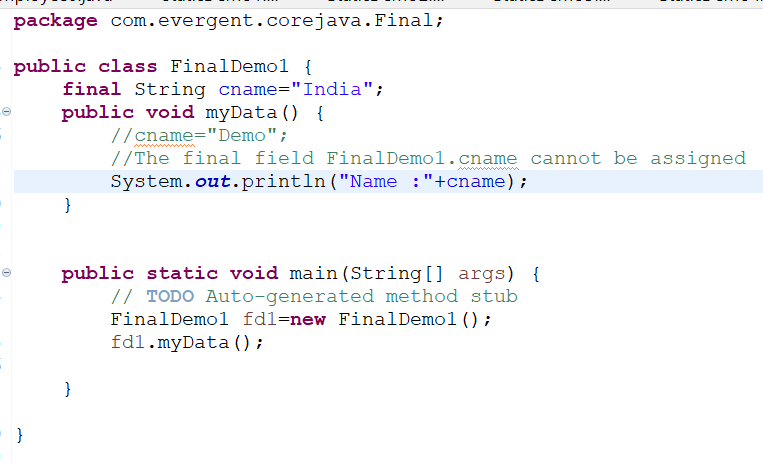
###### We can declare Final as Class,Method And Variable

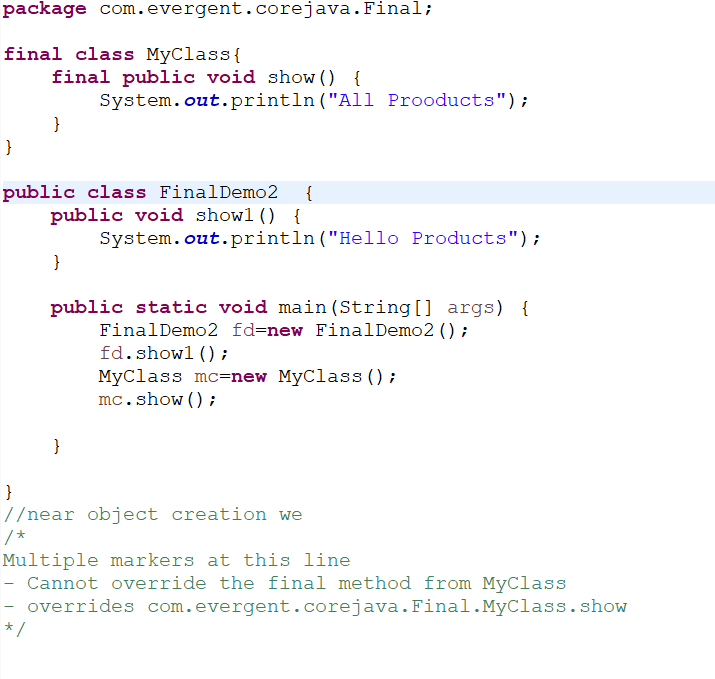
###### Final variables cannot be Modified

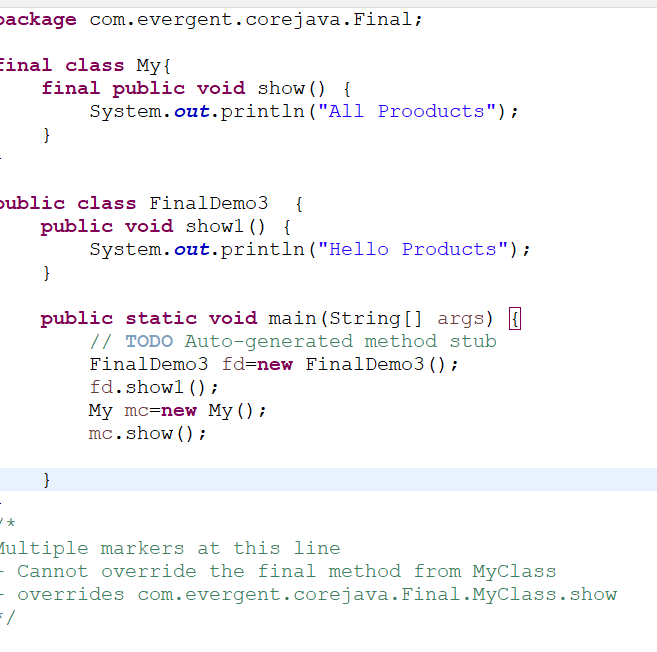
###### Final class cannot be Inherited

###### Final Methods Cannot be Overrided

Examples:







## Strings

### String class

### String Buffer

### String Builder

## String class

* String class is the Final class
* String is immutable
* String Is Non-Synchronized(Not thread Safe i.e Multiple threads Executes Simultaneously)

### String Buffer

* String Buffer is a Final class
* String is Mutable
* String Buffer is Synchronized (Thread Safe i.e Multiple Threads Executes One After One)
* Legasive API(Not Recommended To Use)
* String Buffer is Mainly used in Database

### String Builder

* String Builder is a Final class
* String is Mutable
* String Builder is Mutable

|  |  |  |
| --- | --- | --- |
| String | String Buffer | String Builder |
| 1)String class is Final class | 1)String Buffer is Final class | 1)string Builder is Final class |
| 2)String is Immutable | 2)String Buffer is Mutable | 2)String Builder is Mutable |
| 3)Non-Synchronized | 3)Synchronized | 3)Synchronized |
|  |  |  |

1)**Str1.equals(str2)(equals Method always Checks the Content)**

* Here The String content is saving in the Heap Memory every time you use new it will save in new Memory location.

String str1=new String(“Hello”);

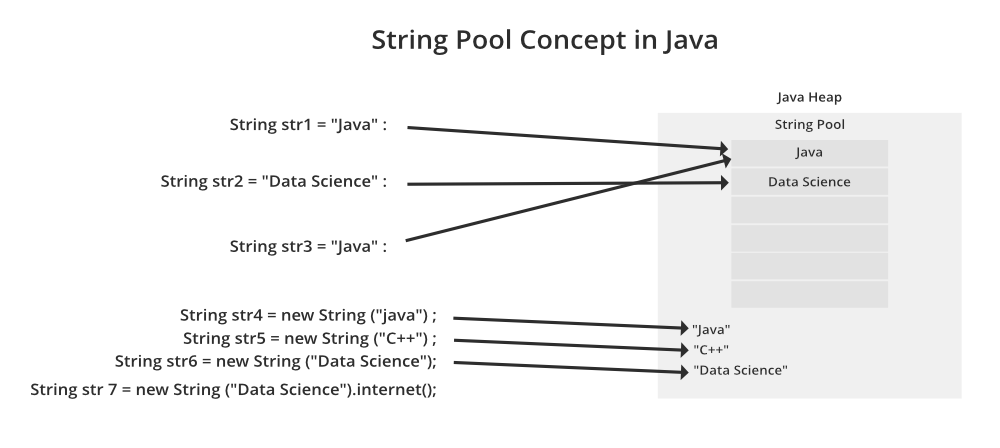
String str2=new String(“Hello”);

1. **Str==str1(== it always Checks the memory location)**

* Here String content is saving in the String Constant pool.it will manage the Memory in String case

String s1=”Java”;

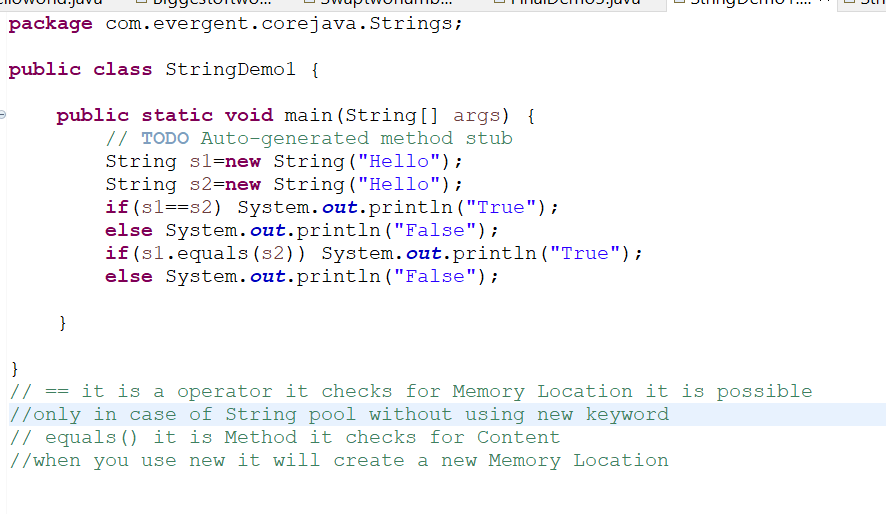
String s2=”Java;

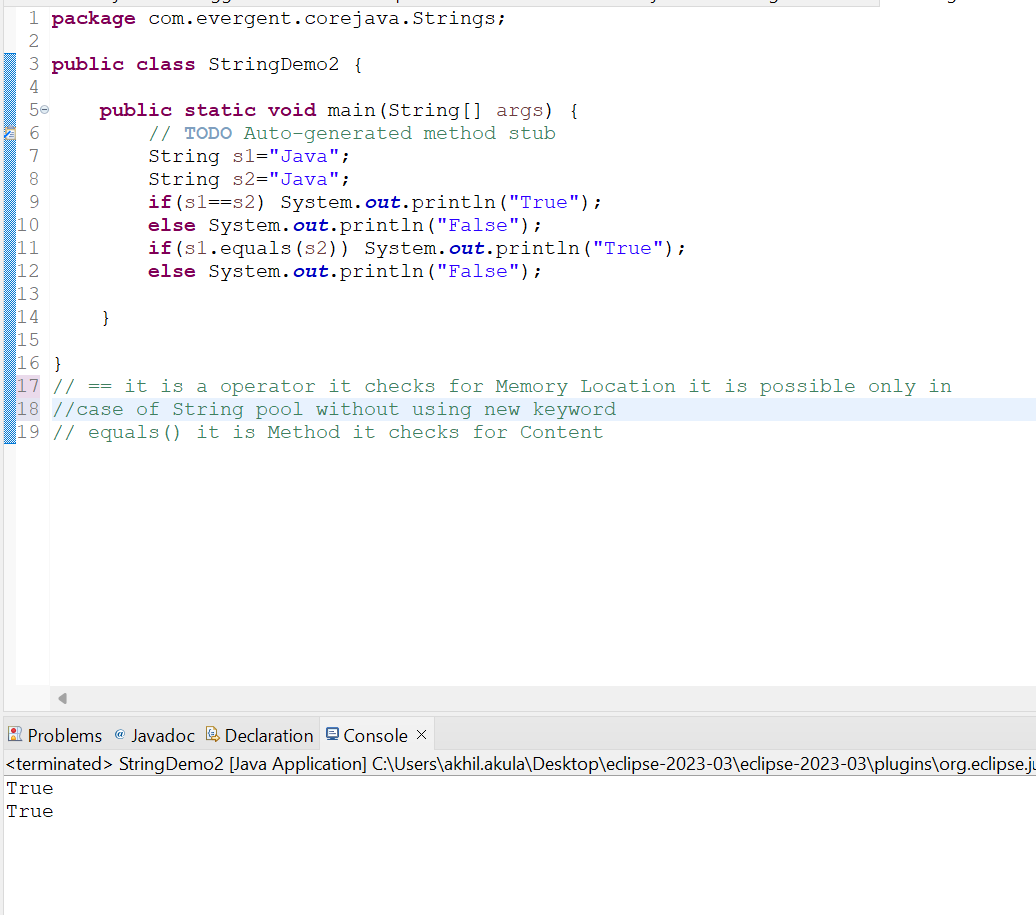


###### String Defination

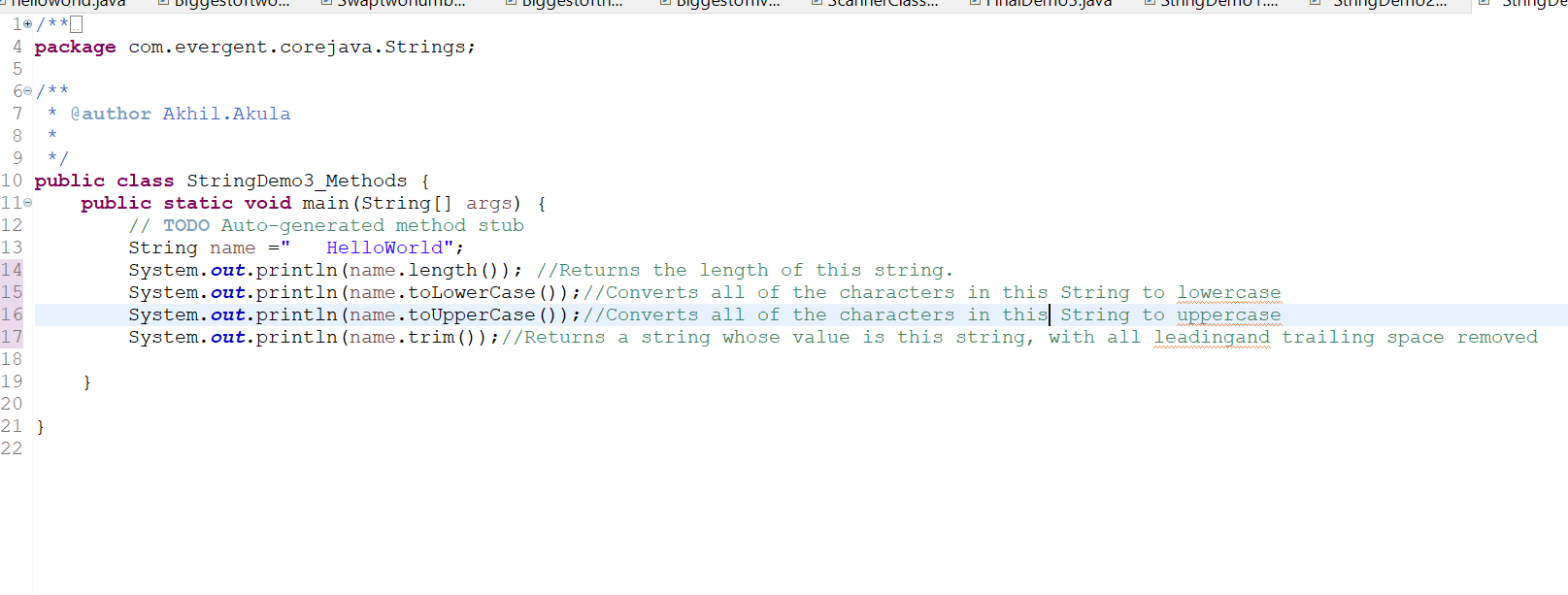
* Once String Object is Created it cannot Modified if we try to modify the existing String then it will goes new Memory Location And the existing String is for Eligible Garbage Collection.
* String is Sequence of Characters ,often Called Text.
* String is declared in two Ways :
* 1-String s1=new String(“Hello”); (Heap Memory)
* 2-String s2=”Java”;(String Constant pool)
* Once String object is created It cannot Modified
* All String class Methods are Non-Synchronized.

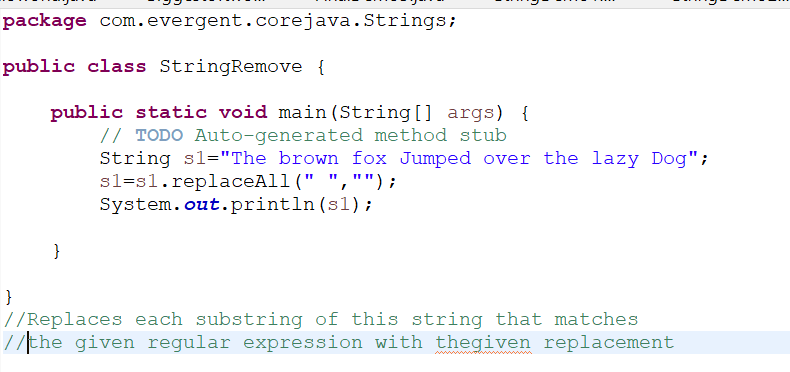
### Example Program:

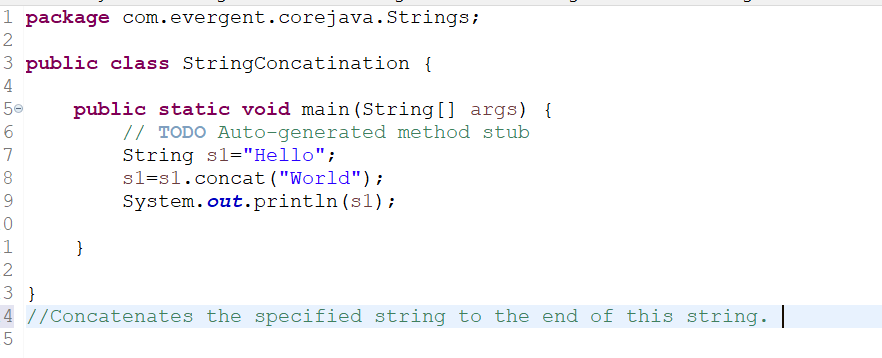




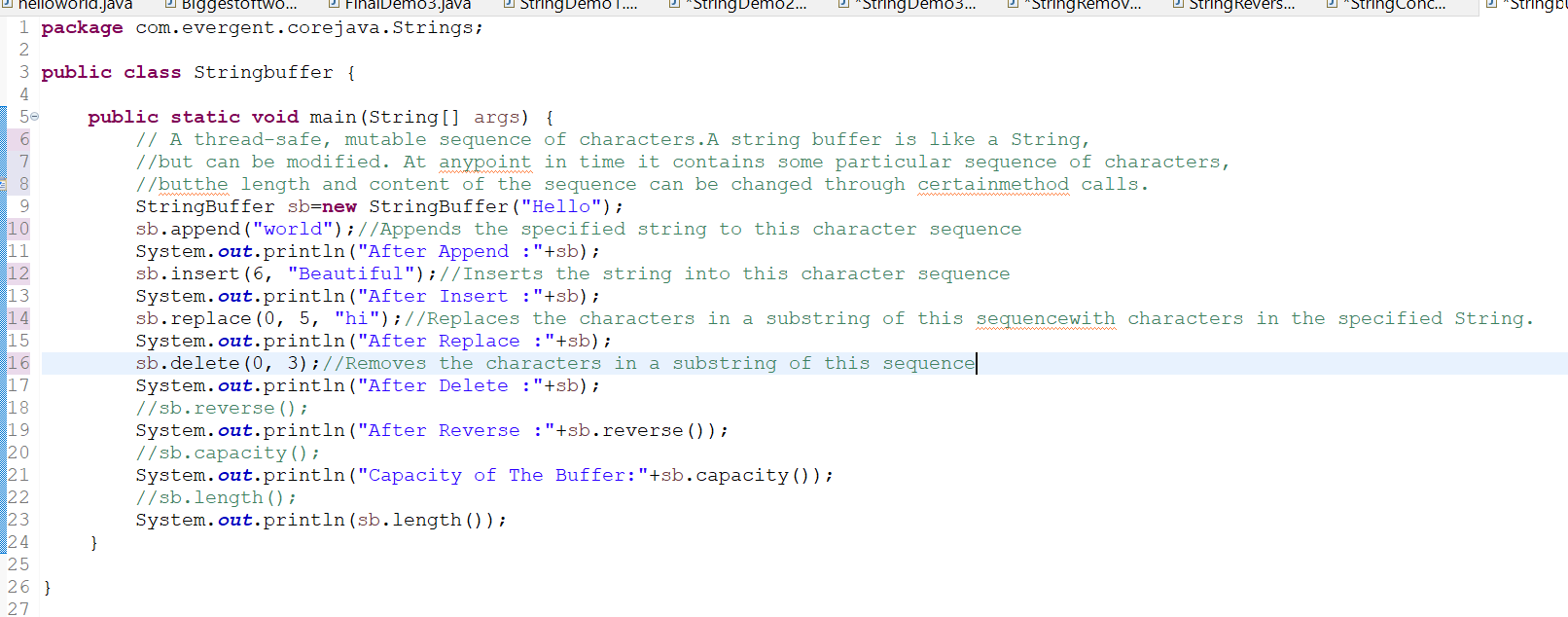
### String Methods:



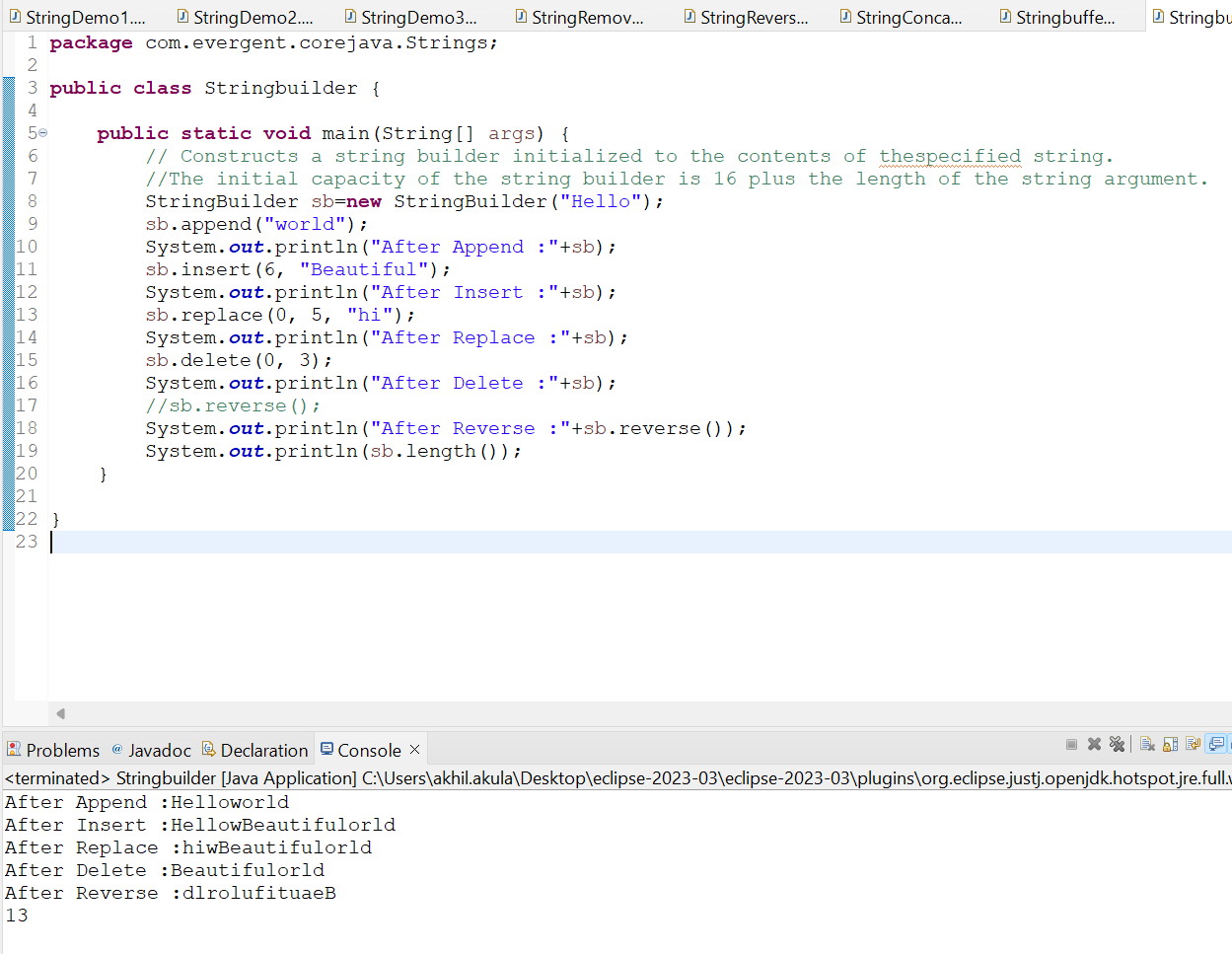




### String Buffer Methods:



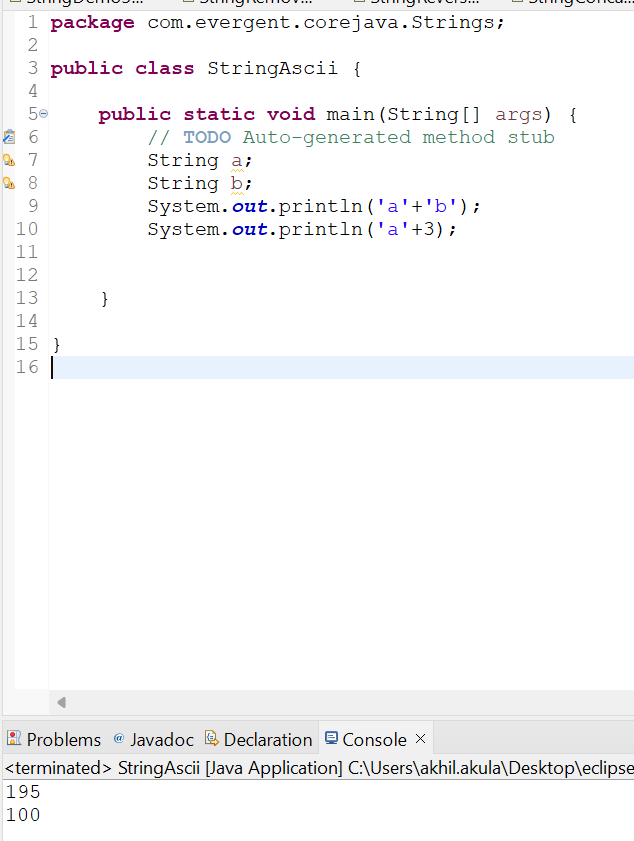
### String Builder Methods:



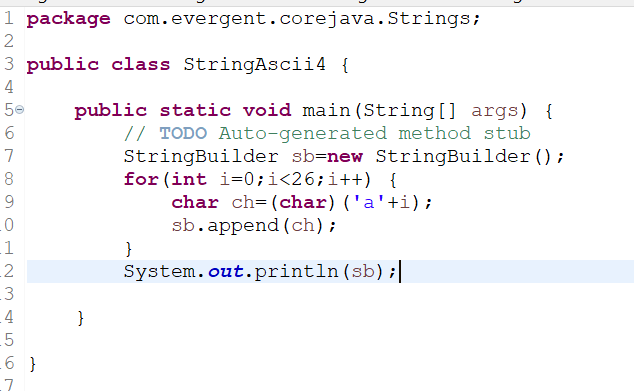
#### ASCII(AMERICAN STANDARD FOR INTERNATIONAL INTERCHANGE)

A-65 a-97

**PROGRAMS:**







## INTERFACE

###### Interface is a keyword

###### By Default Every Method in the Interface is abstract which only have method signature only not Implementation

###### We Can Define Variables in interface which can be default,public ,static and final

###### If any class Implements interface then that class should override all the methods of interface otherwise that class will show compile time error

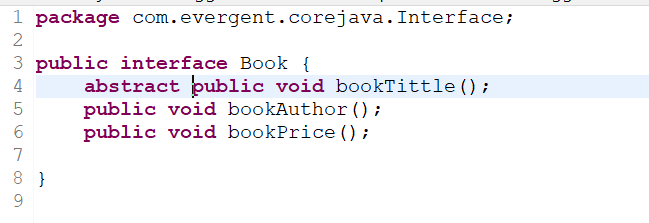
###### We cannot create Object for Interface but we can Create reference by creating reference to interface we can acess all the method in interface not class methods

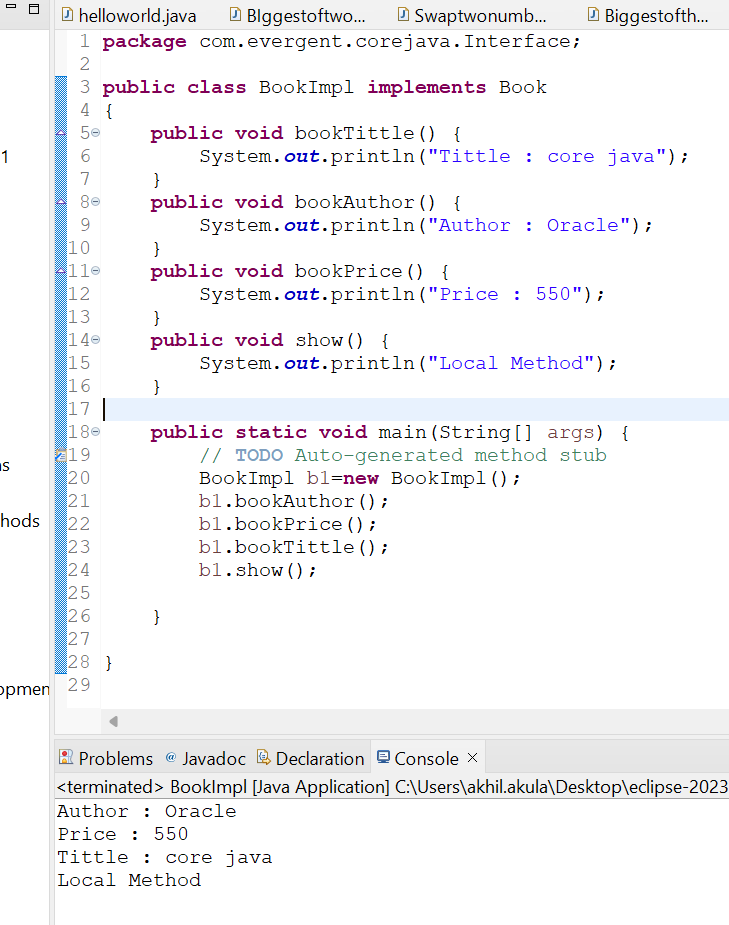
###### One Interface extends other Interface.

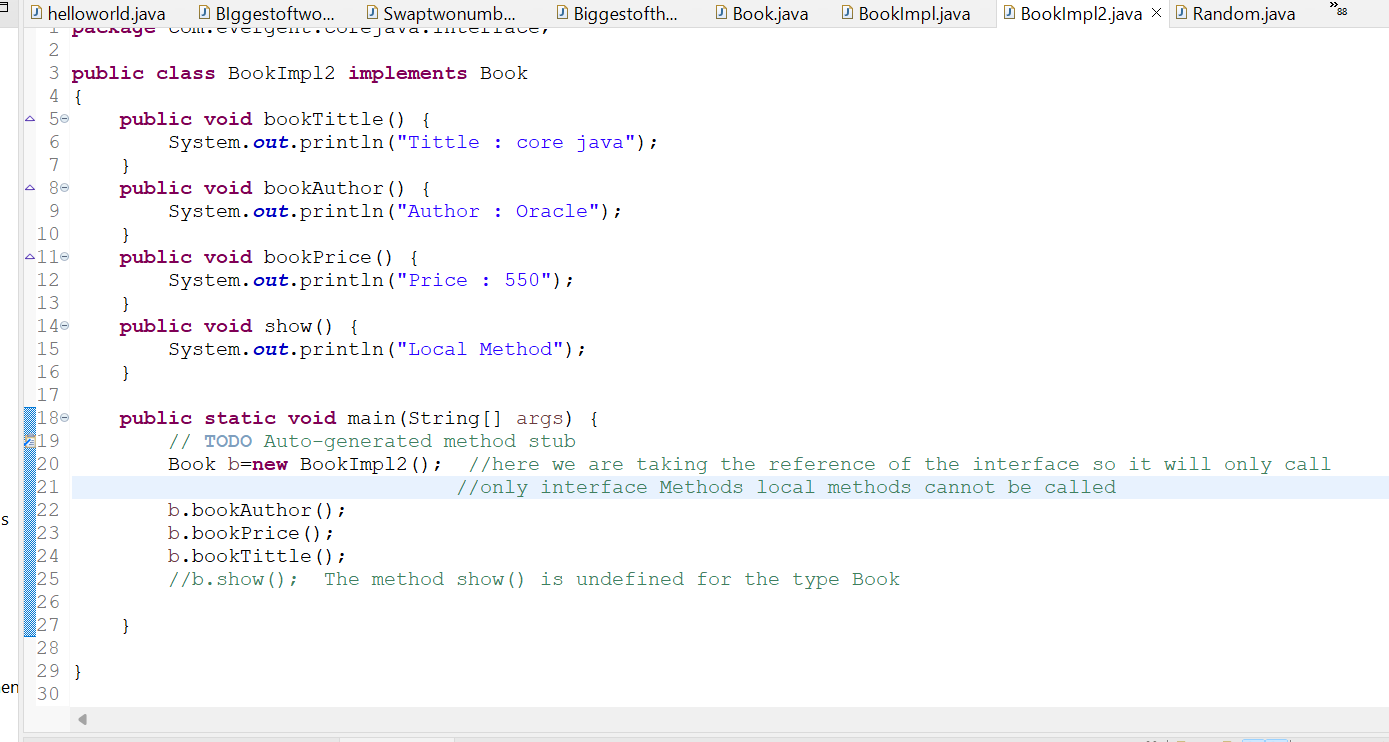
###### In java , Multiple Inheritance possible via Interface

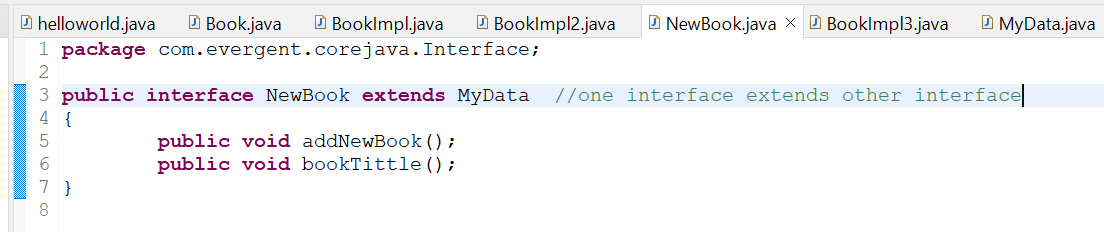
###### Implements is a keyword used to implement interface to class

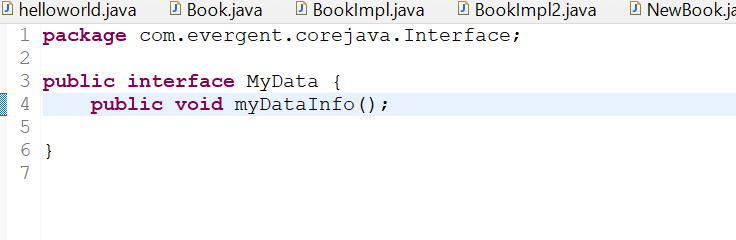
###### Example Programs:



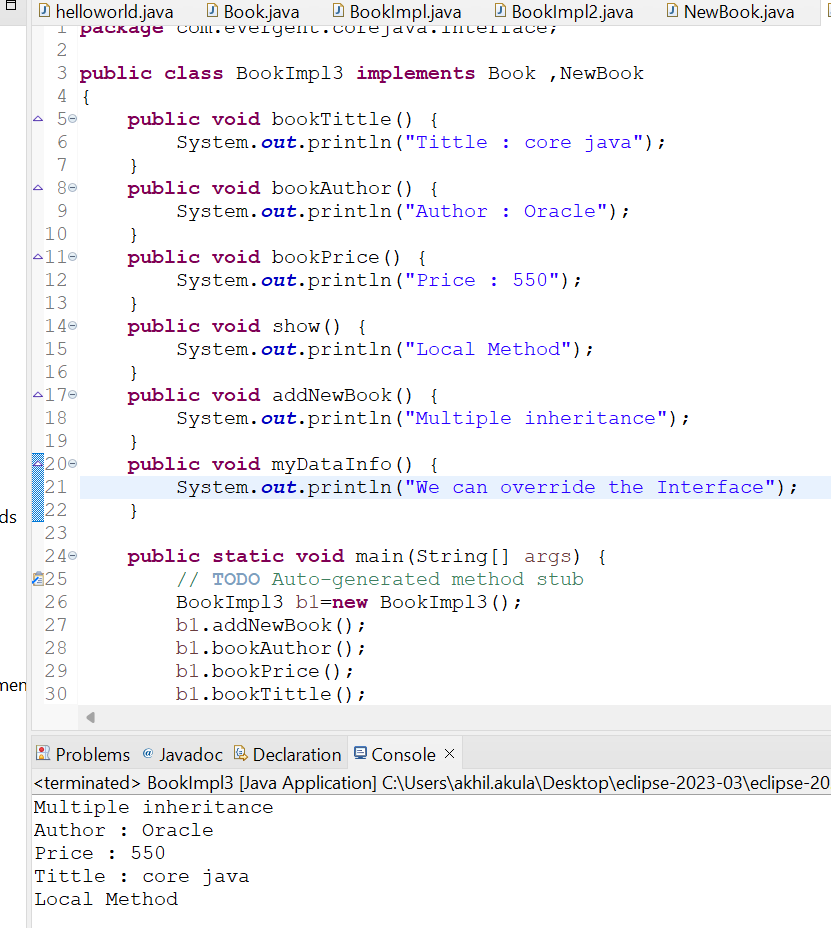








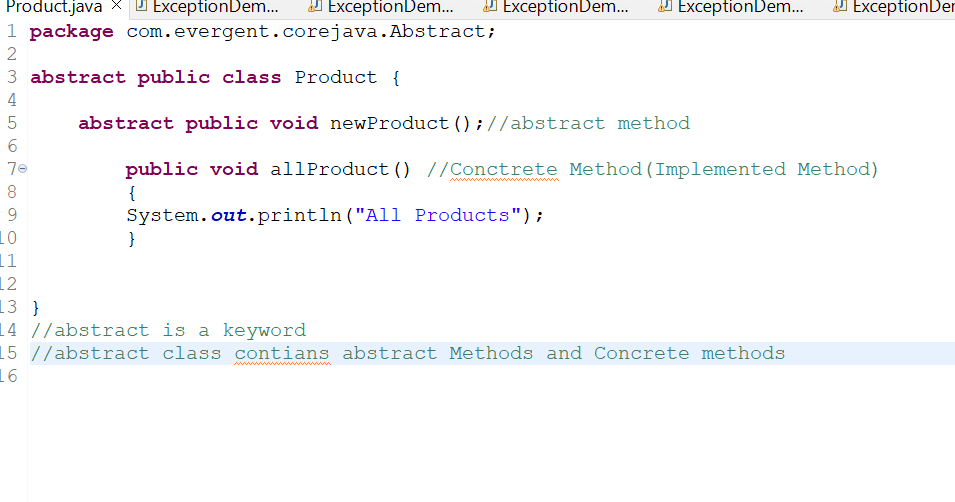
###### Multiple Inheritance:

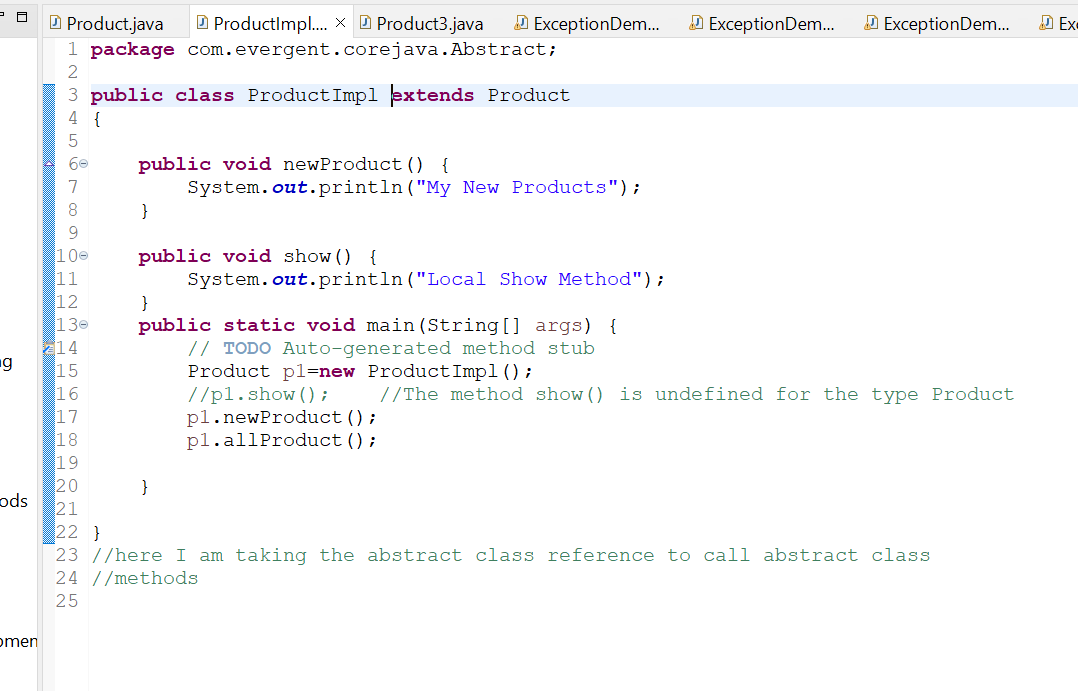


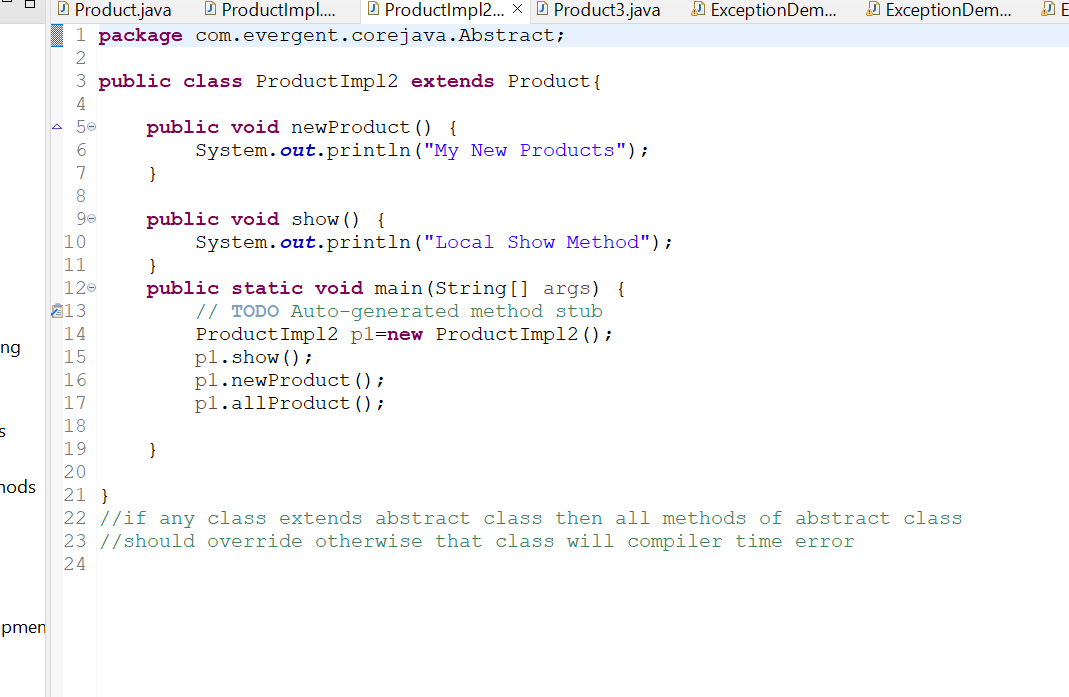
##### Abstract Keyword

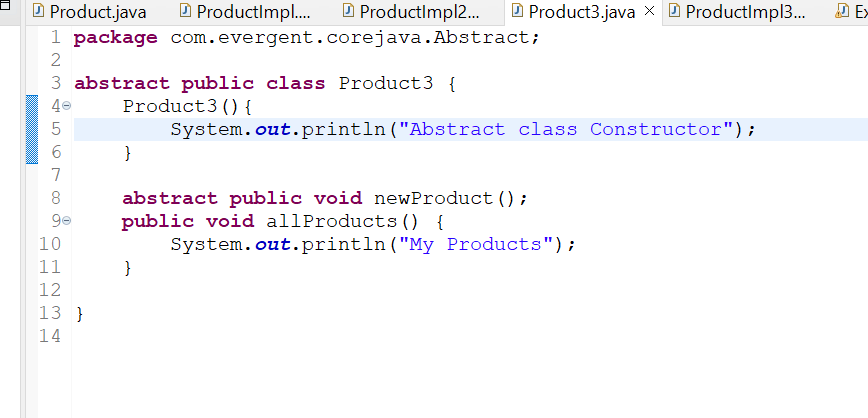
* Abstract is a keyword.
* Abstract class contains abstract methods(Non Implemented Methods) and concrete methods(Implemented Methods
* Any class contains abstract methods then that class sholud be declared as abstract Keyword otherwise that class will show compile time error
* One class can extend abstract class ,and that class should override all abstract class methods otherwise that class shows compile time error.
* We can declare constructors inside the abstract class and we can acess that constructors while sub class object creation
* We cannot create object to abstract class instead we can create Reference to abstract class

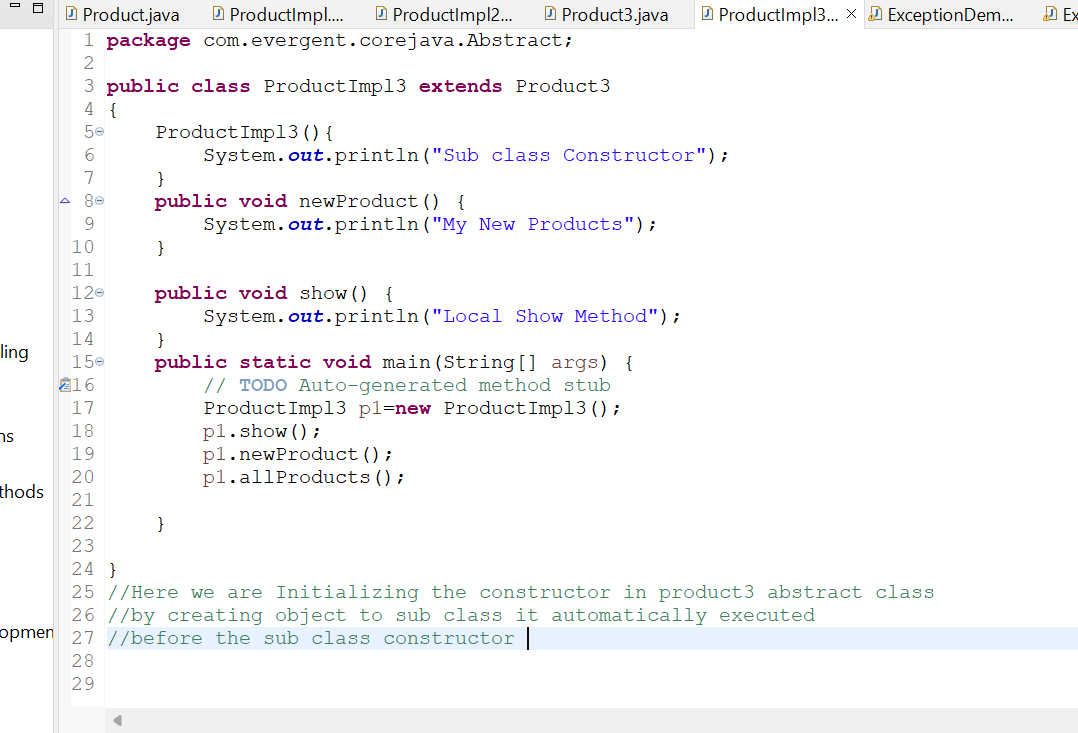
**Programs:**







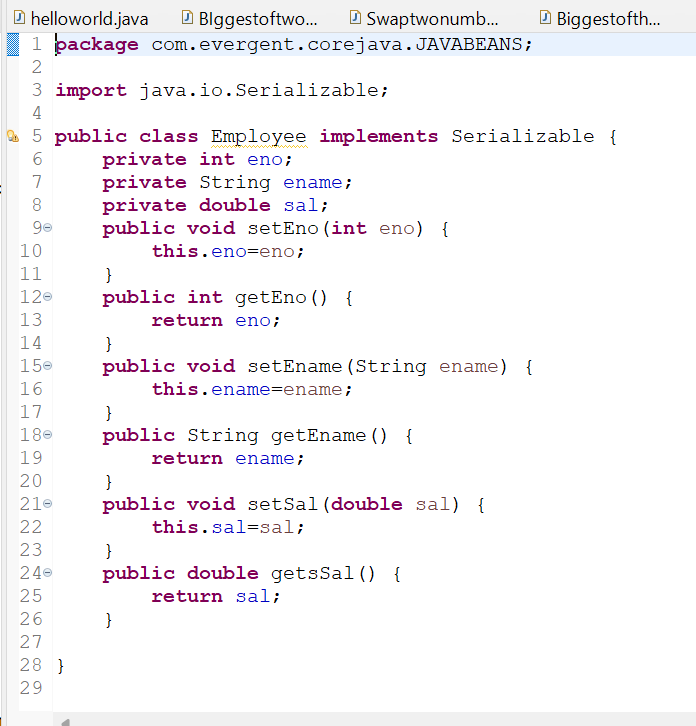


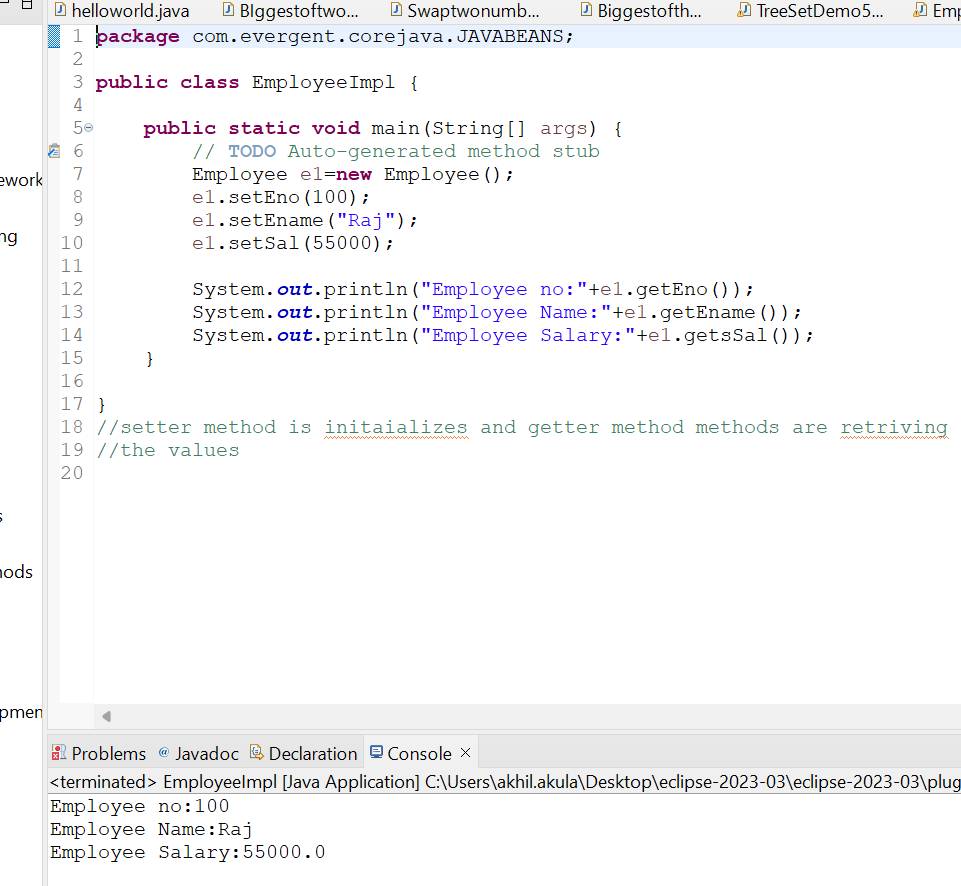


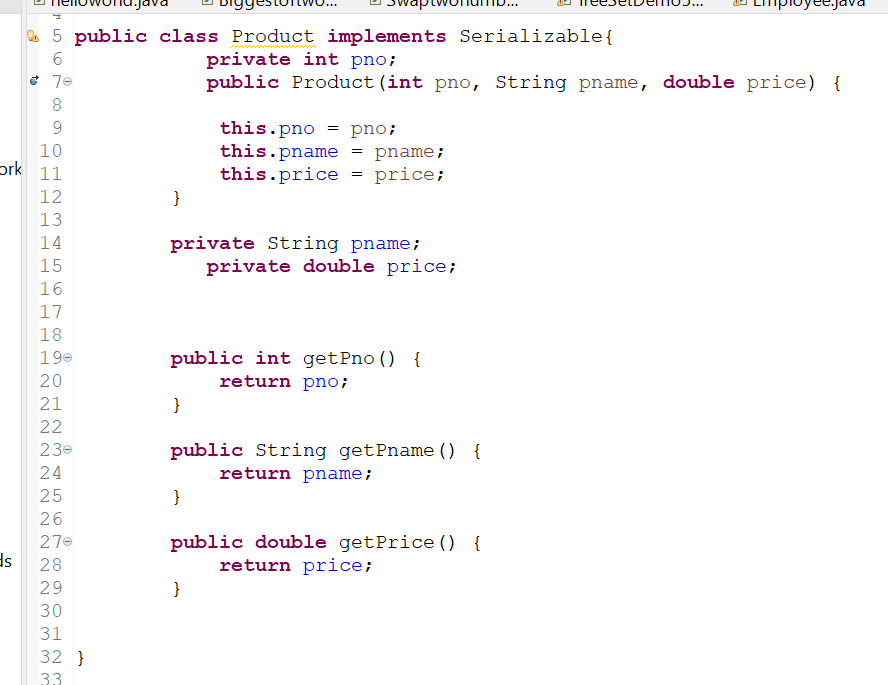
##### Java Beans

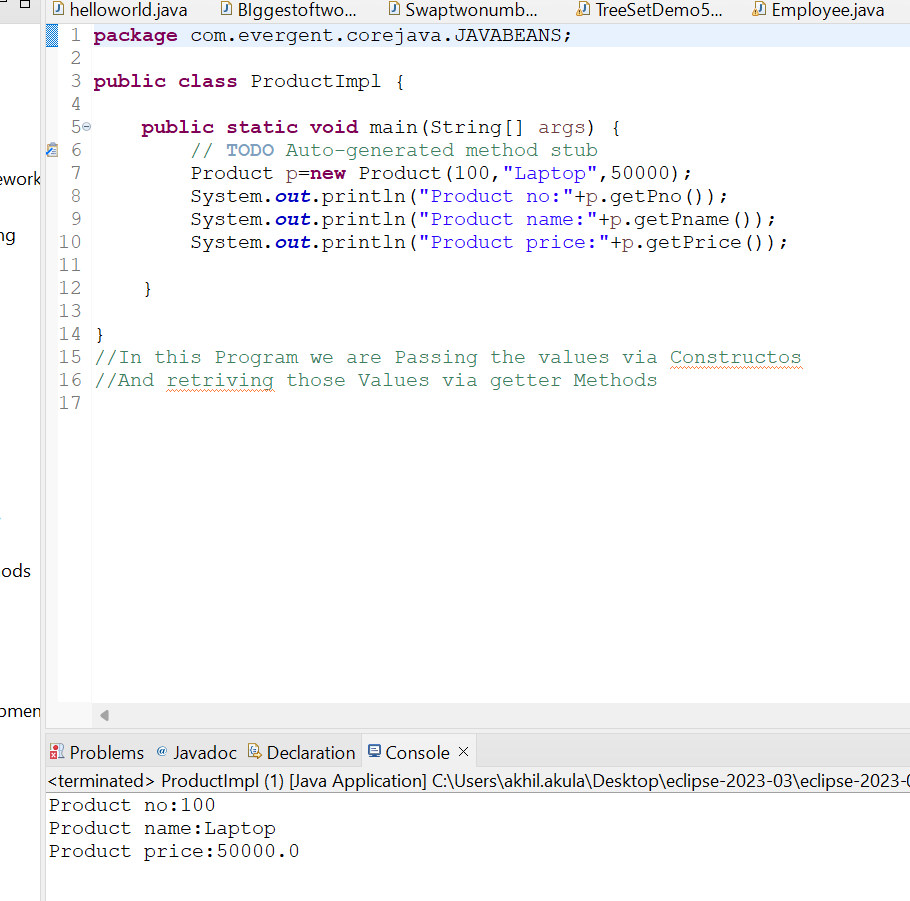
* Java Beans is a mechanism
* Java is Light Weight(I.e less Methods)
* All Attributes are private, get/set methods are public
* Implements java.io.serializable
* We can achieve Tightly encapsulation through java
* Java Beans is nothing but it acts as storage .we can store our data in beans
* Serialization is The process of converting objects into binary data so that we can store that data into a file otherwise we cant store object data directly to a file

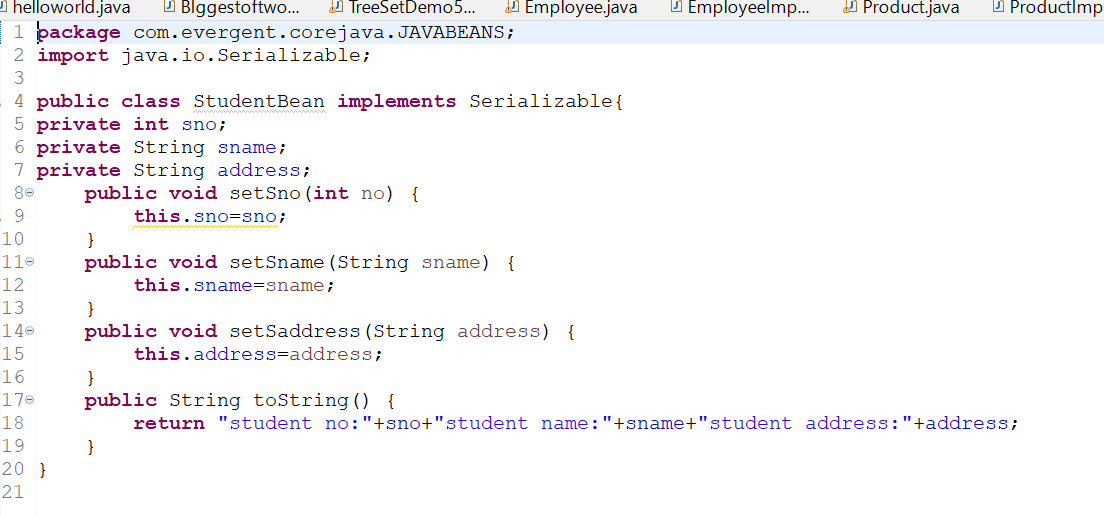
##### Programs ;











##### 

##### Exception Handling

* **Exception Handling is a Mechanism**
* **Exceptions are inbuilt Mechanism in java**
* **All Exceptions are Executed while abnormal conditions only**
* **Normal flow it wont execute any exceptions**
* **Once any exceptions are occurring in java code then remaining lines of code is unreachable**
* **Java.lang.Throwable is a super class for Exceptions and error**
* **There are two types of Exceptions in java**
* **Checked Exceptions**
* **Unchecked Exceptions**
* **All Checked Exceptions are Compile Time Exceptions**
* **All Unchecked Exceptions are Run Time Exceptions**
* **There are 5 Keyword in Exceptions Handling**
* **Try**
* **Catch**
* **Finally**
* **Throws**
* **Throw**
* **Try block is for Bussiness Logic**
* **Catch block is for Handling the Exceptions**
* **Finally will Executes Irrespective of Errors**
* **Throws an exceptions will be executed method by method**
* **Throw is for Run Time Exceptions and will call pre defined Exceptions (Checked and Unchecked )classes**
* **Try followed by either catch block or finally block**
* **We should follow Exceptions Hierarchical**
* **We can create our own exceptions (user Defined)**
* **Our own exceptions extends Exceptions or Run Time Exceptions**
* **All exceptions classes are into java.lang package**
* **If our class having two exceptions then developer should handle them one after other**
* **Errors are not in developers control**