**INDEX**

**05-08-2024**

1. Languages & Packages
2. Java Features
   1. Why Java is platform independent
3. JDK, JRE, JVM
4. Basic Java
   1. Datatypes
   2. Operators
   3. Conditions(if, if-else)
   4. Loops(Nested for loops)
5. Packages

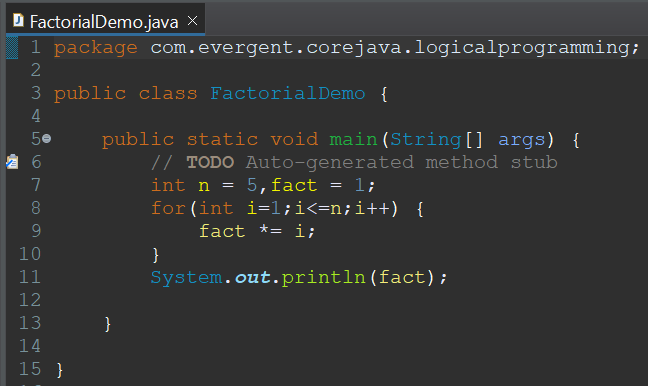
CoreJAVA-Workspace :

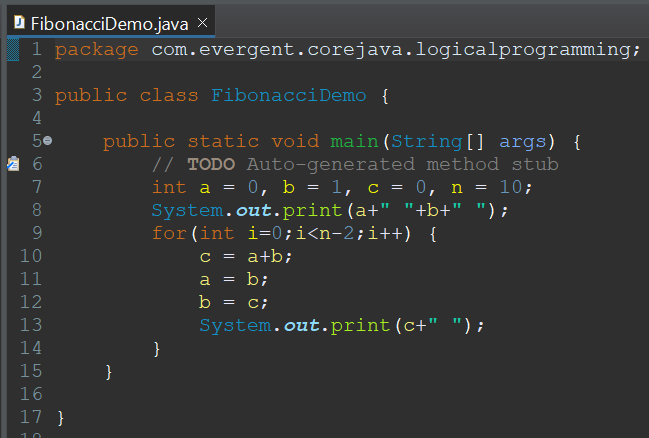
CoreJAVA-Development

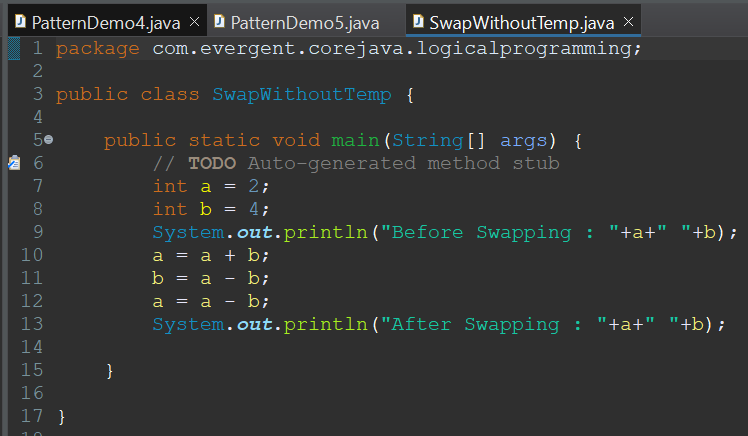
CoreJAVA-Application

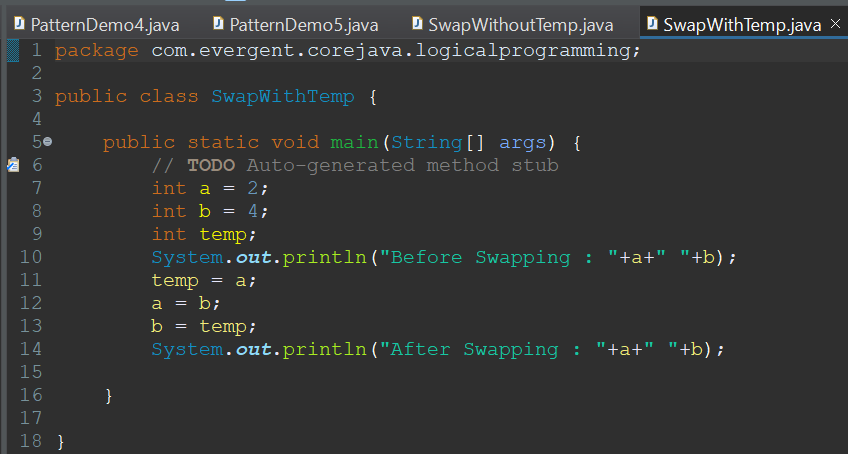
**06-08-2024**

1. **Logical Programming**

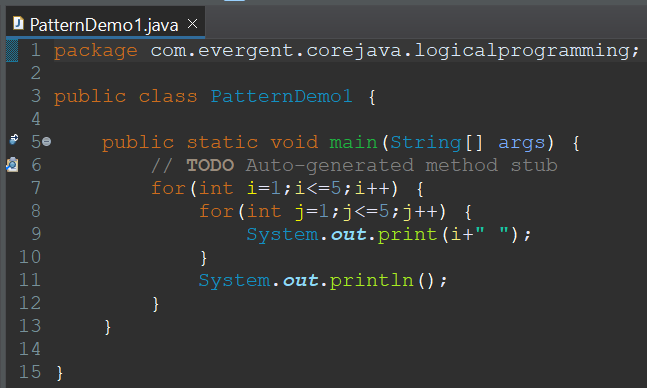


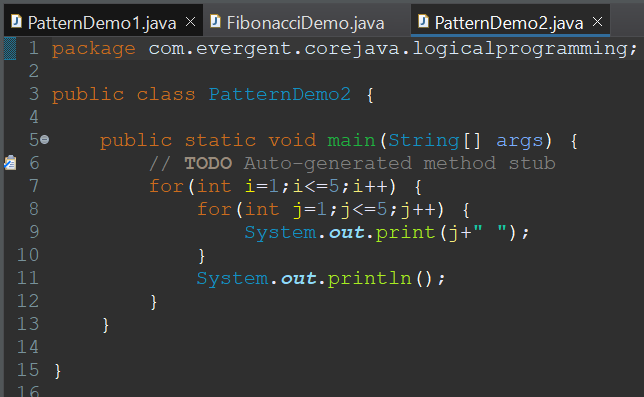


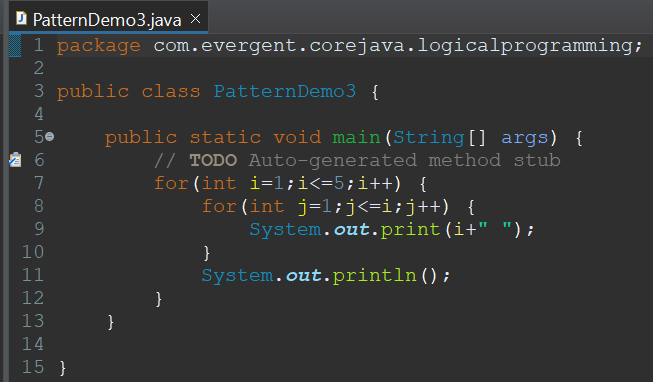


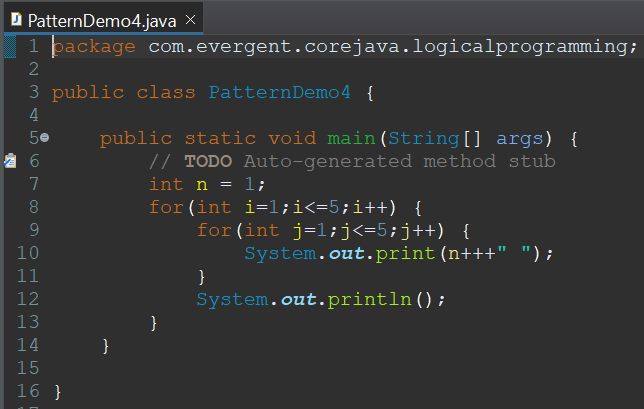


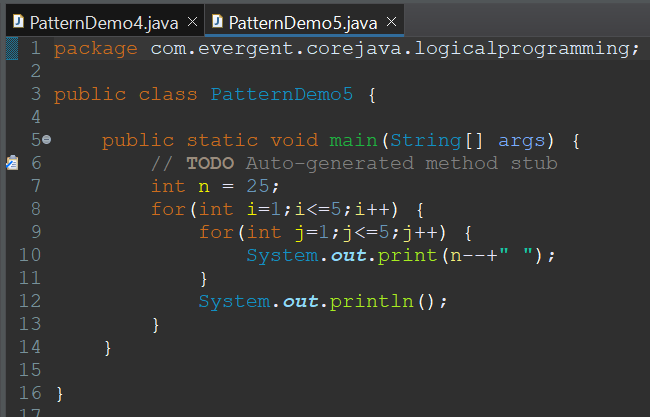
1. **Patterns**



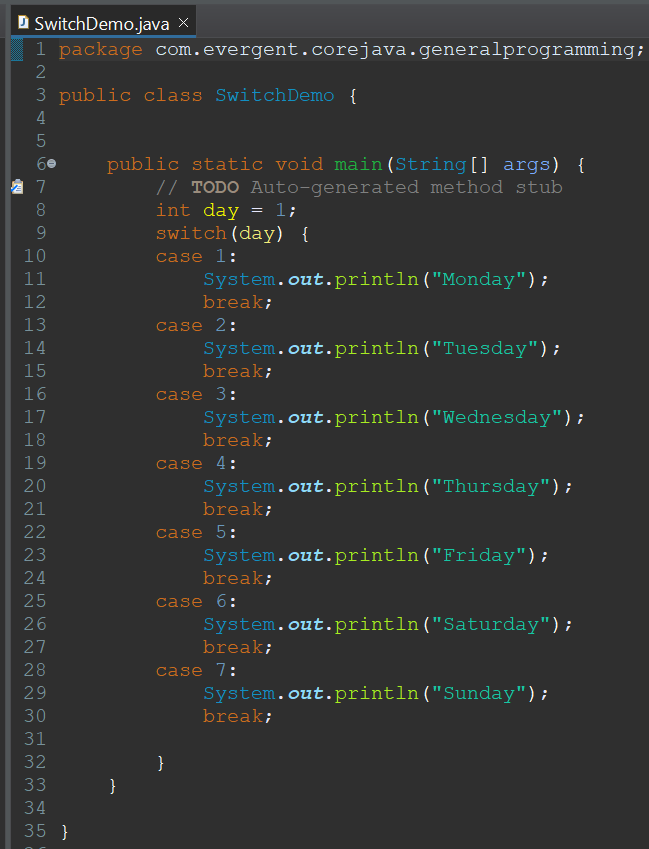




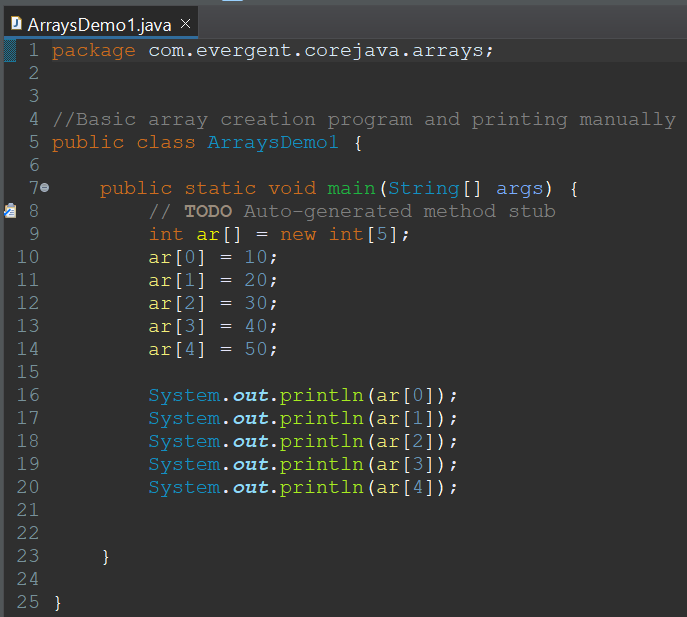


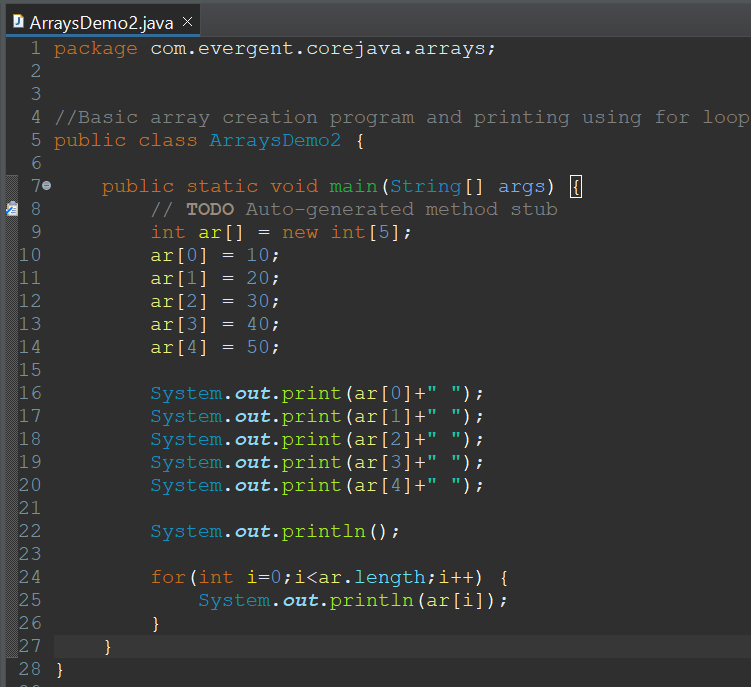


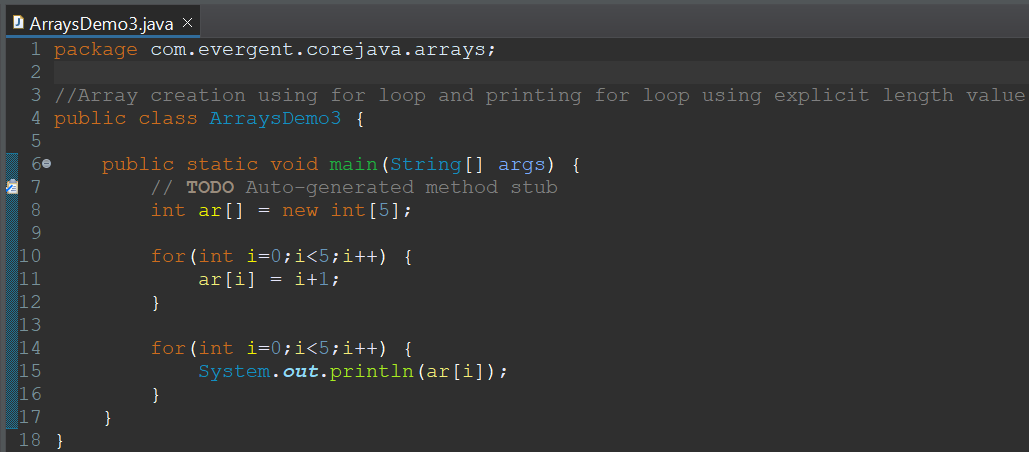
1. Switch Case

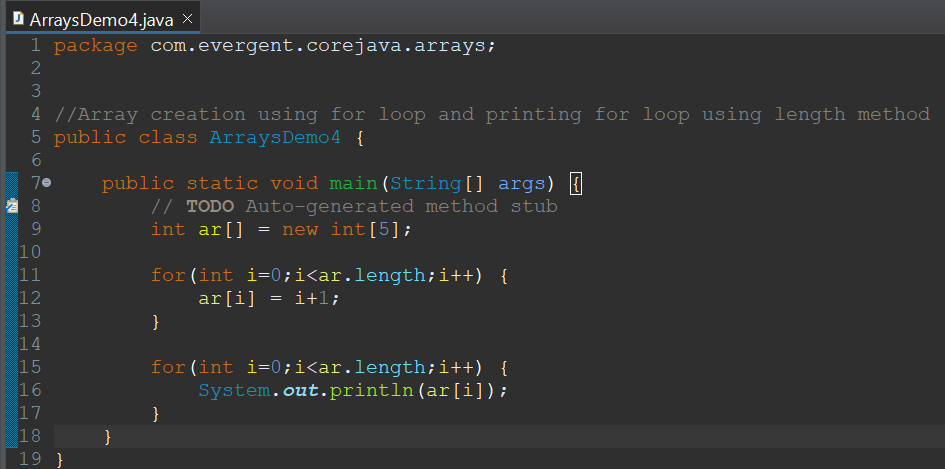


1. **Arrays**Arrays are used to store similar type of data in contiguous memory locations.  
   Arrays are of fixed size.  
   Arrays provide easy way to retrieve elements.  
   Arrays are of 2 types  
   a) 1D Arrays  
   b)2D Arrays
   1. 1D Array's

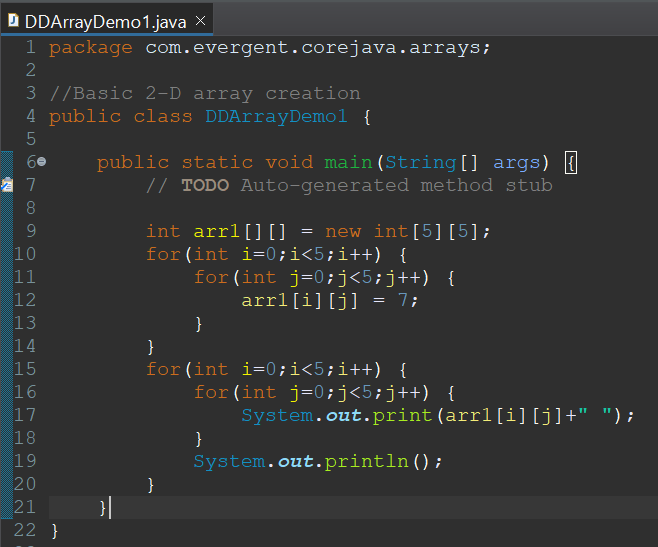






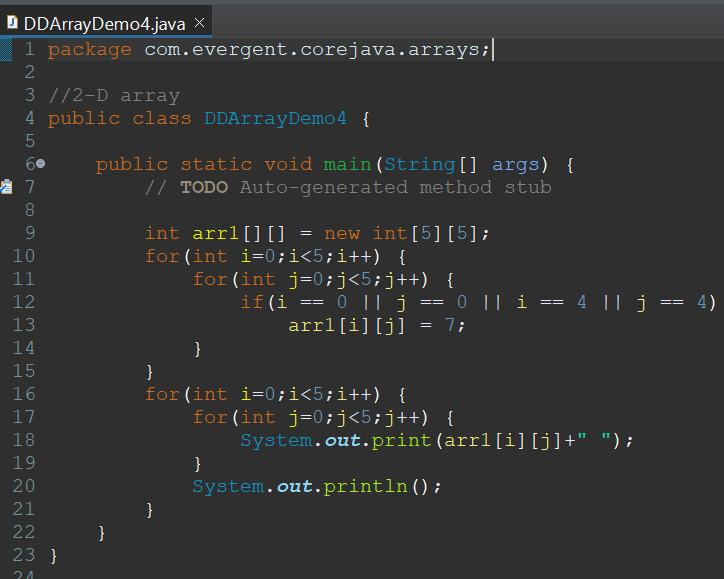


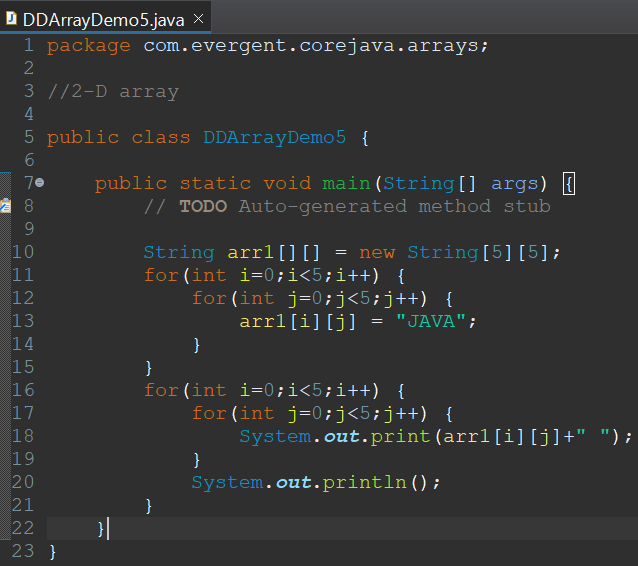
* 1. 2D Array's





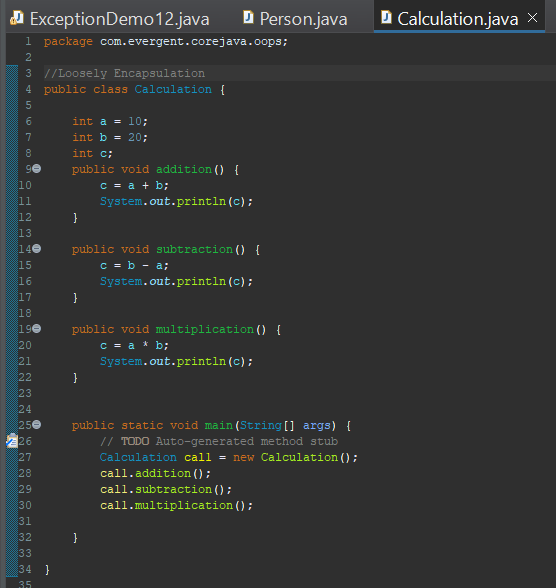
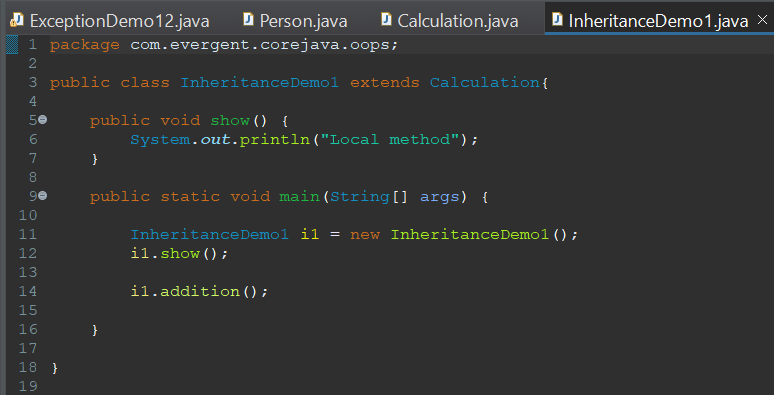
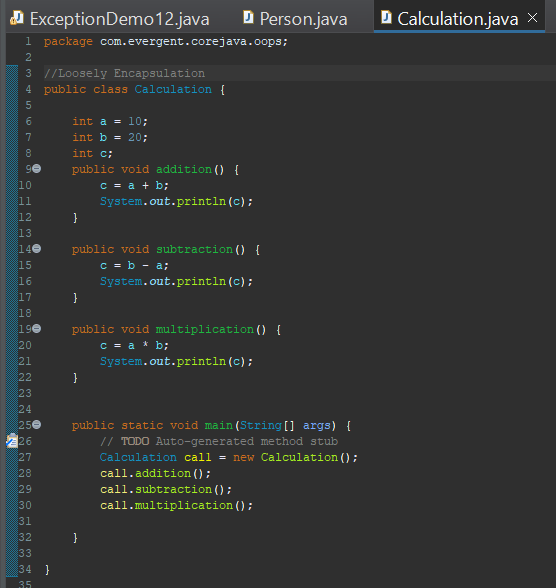
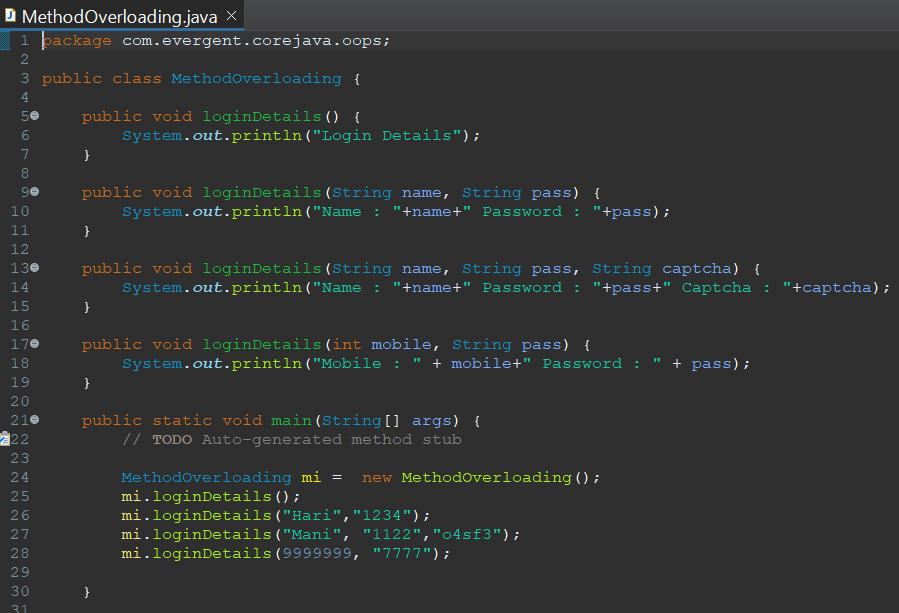
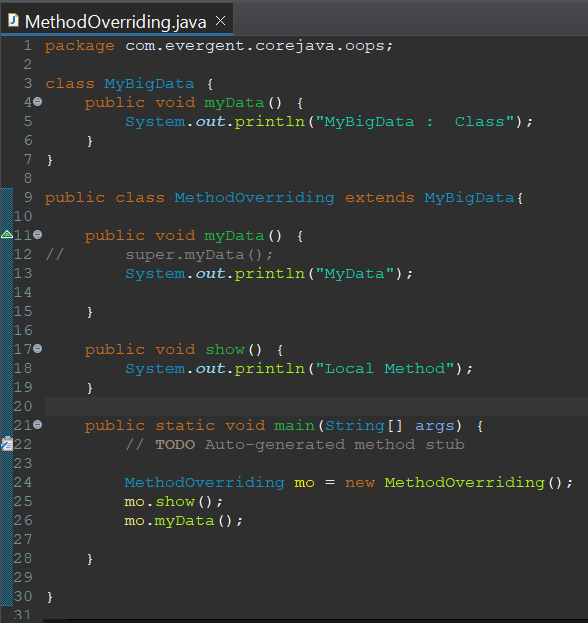
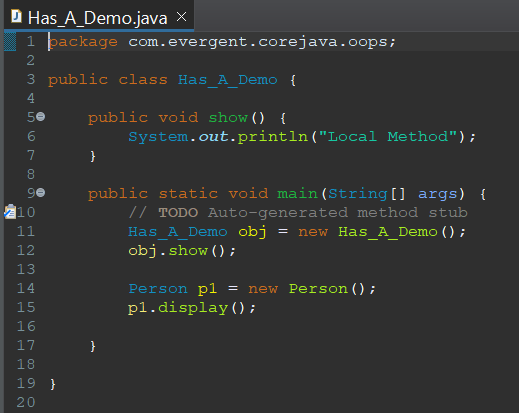






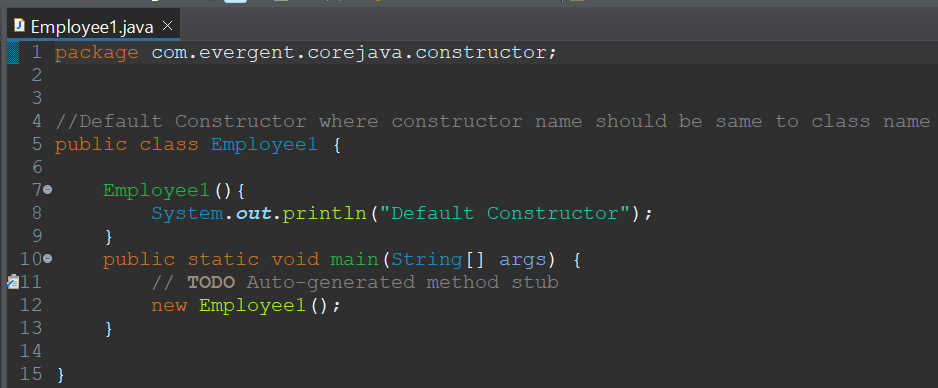
1. **Enum's  
   These are special classes which has their own data types.  
   These are constant’s that cannot be changed and should be in capital letter.  
   These are static and final by default.  
   During compile time only we know all values like deck of cards, colors, days etc.  
   We can add variables, methods and constructors in enums.  
   We cannot inherit other classes nor make enum class as super class to be extended.**
2. **Event Management System**

**07-08-2024**

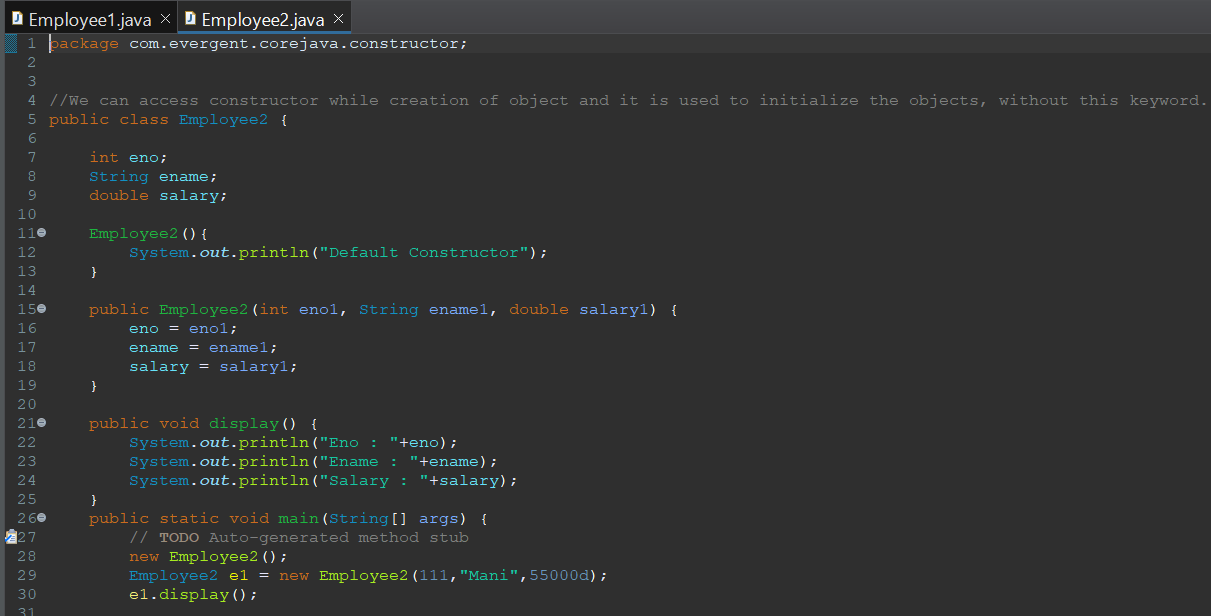
1. **OOPS**
   1. Encapsulation  
      
   2. Inheritance  
        
      
   3. Polymorphism  
        
      
   4. Abstraction
   5. Method Flows  
       Method with return type and with out arguments.  
       Method with return type and with arguments.  
       Method with out return type and with out arguments.  
       Method with out return type and with arguments.
   6. IS-A and HAS-A relationship  
      
2. System class

**08-08-2024**

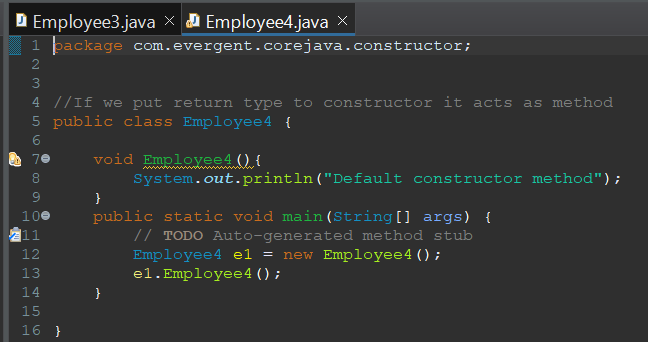
1. **Constructors**
   1. Constructor name should be equal to class name



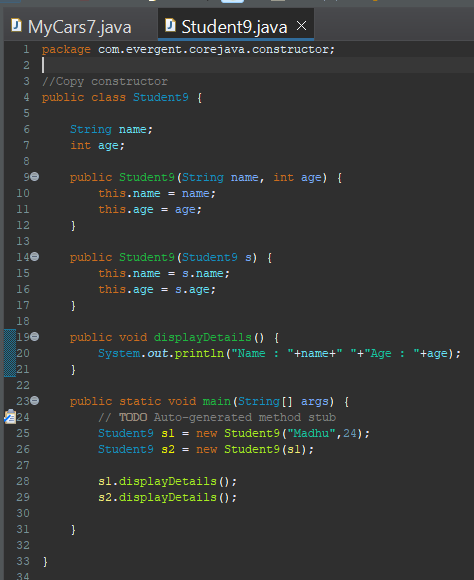
* 1. There are 2 types of constructor
     1. Default
     2. Parameterized



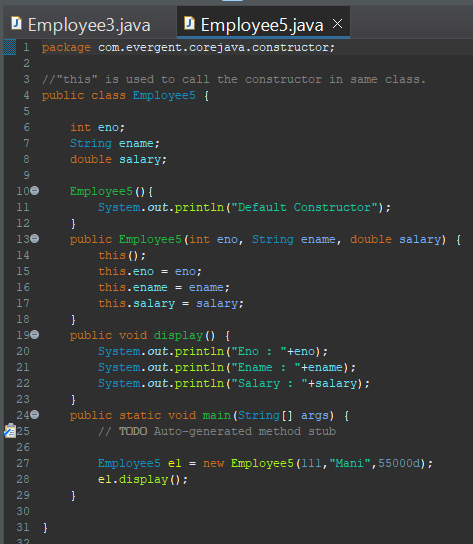
* 1. Constructors are called during object creation
  2. Constructors are used for initialization of objects
  3. Constructors doesn’t have return type not even void, if we define return type for constructors it will act as method

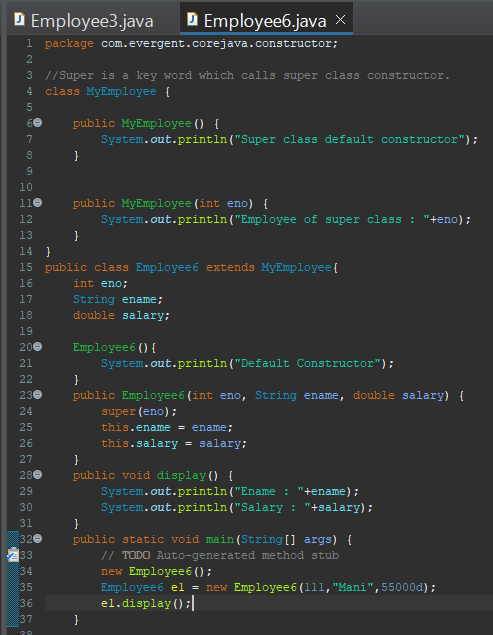


* 1. Every class will have default constructor
  2. this & super are used
  3. Also have other type of constructor called Copy Constructor



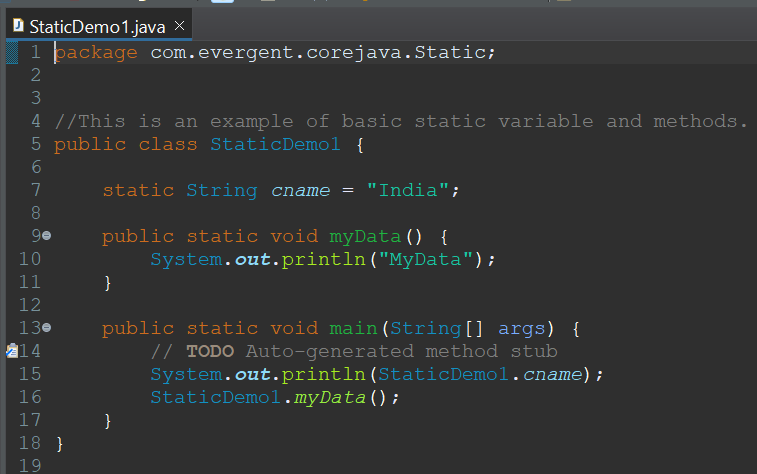
* 1. Constructors can be overloaded, but cannot be overridden
  2. Constructor chaining - this() is used to call current class constructor, super() is used to call super class constructor



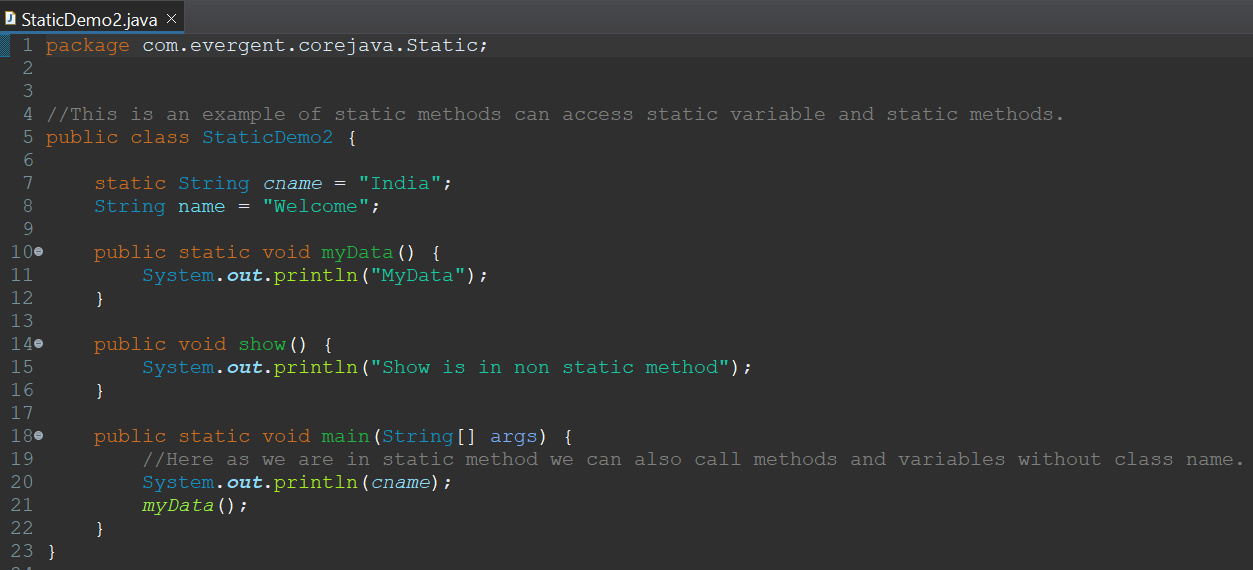


**09-08-2024**

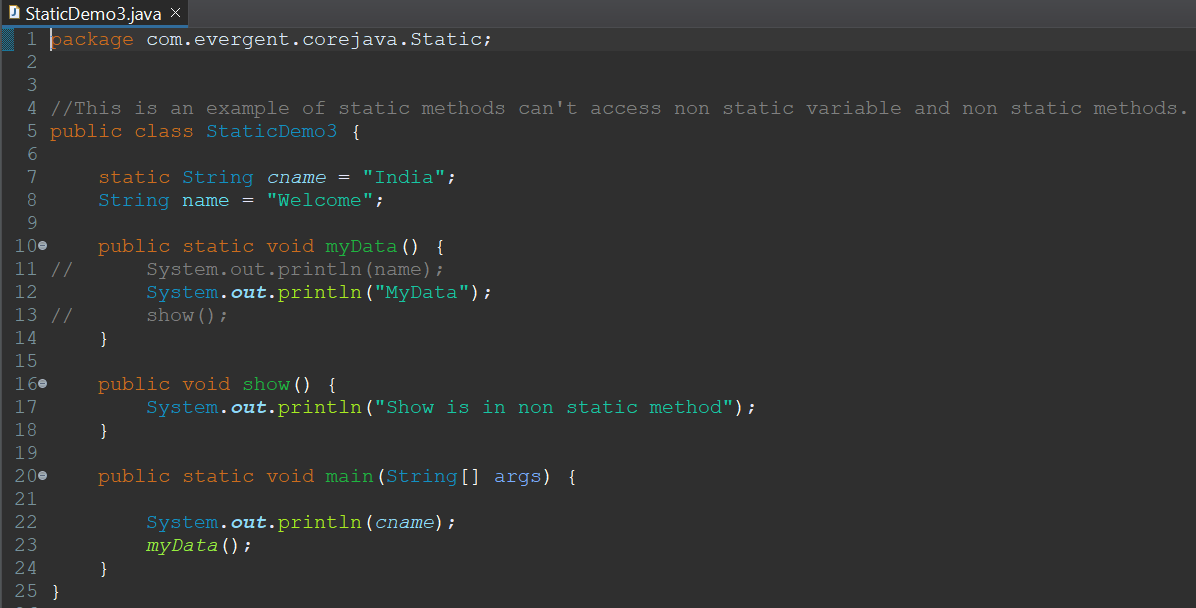
1. **Static** 
   1. Static is a keyword
   2. We can declare variables and methods as static



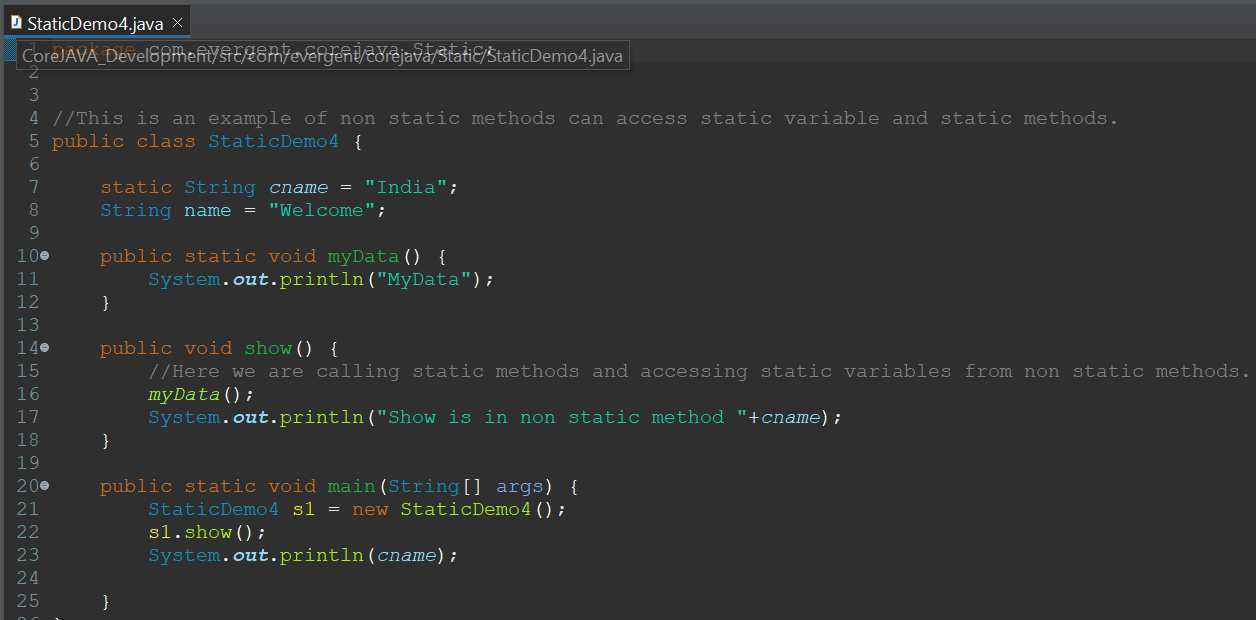
* 1. Static variables and methods are called by using ClassName.method and Classname.Variables
  2. Static method can access Static variables and Static methods



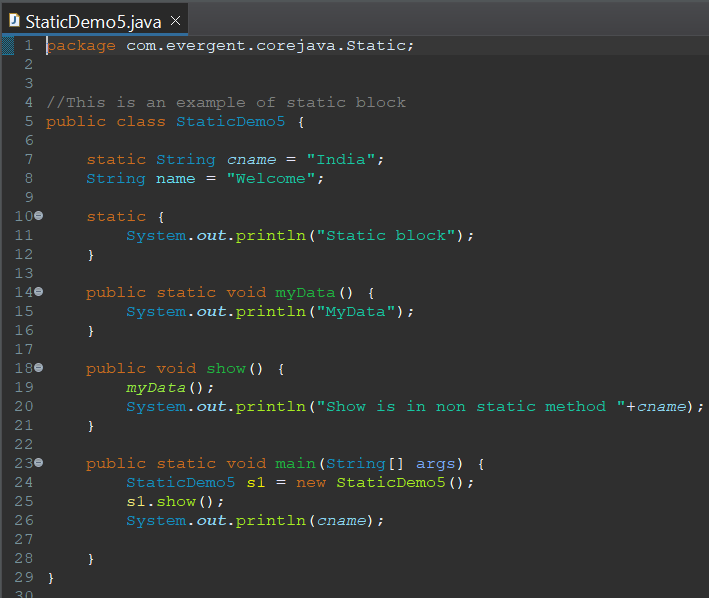
* 1. Static method cannot access Non static variables and Non static Methods



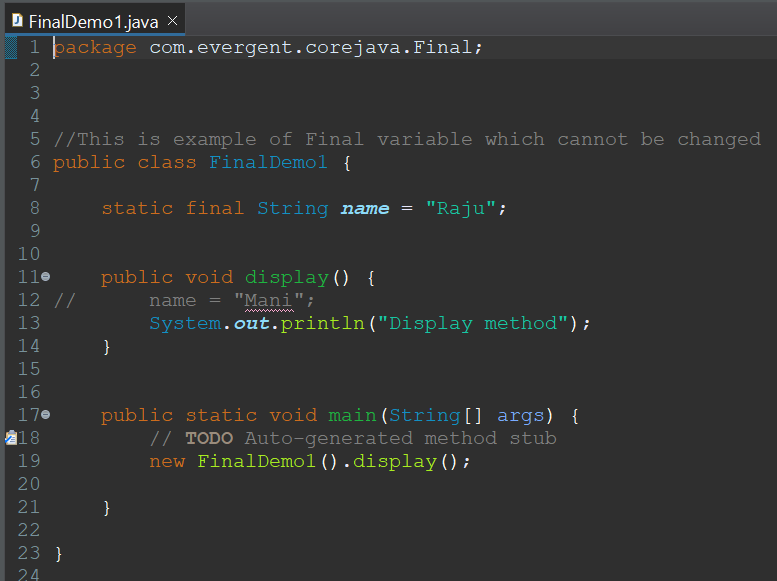
* 1. Non static method can access Static variables and Static methods



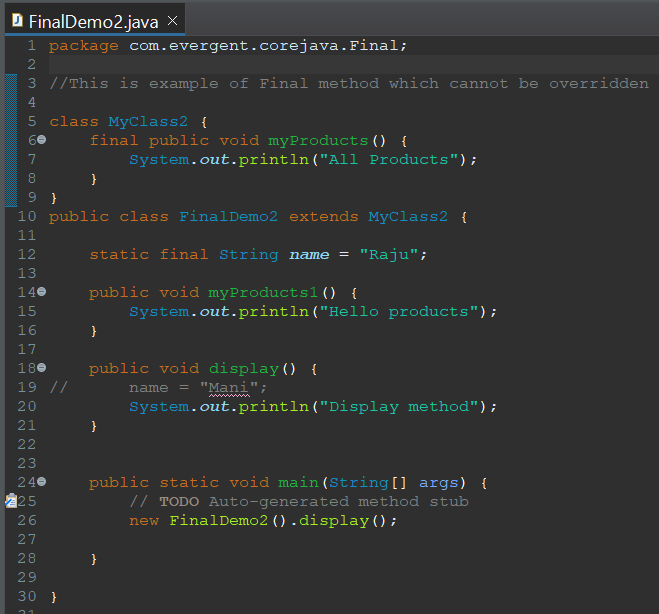
* 1. Static block is initialized first during class loading



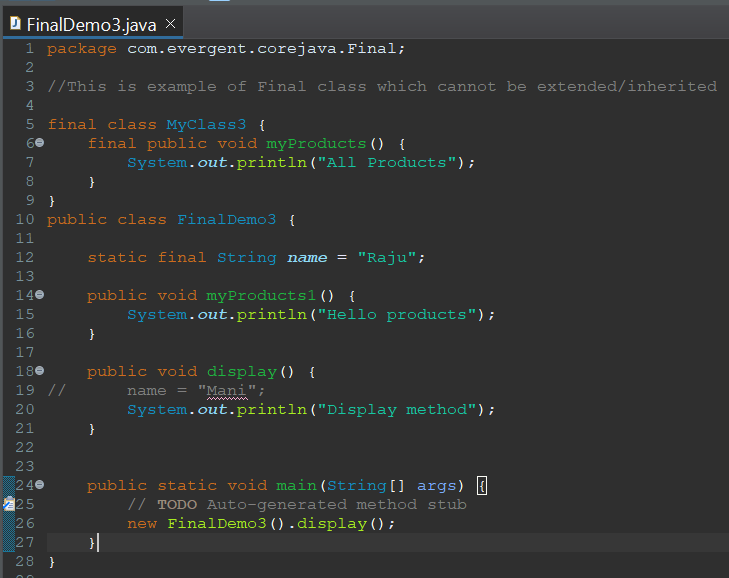
1. **Final**
   1. Final is a keyword
   2. We can declare Variables, Methods, Classes as final
   3. Final variables cannot be changed



* 1. Final methods cannot be overridden



* 1. Final classes cannot be extended



* 1. Final variables can be initialized through constructor or object creation
  2. Final variable can be accessed through HAS-A relation ship

**12-08-2024**

1. **String**
2. String Class
3. String Buffer
4. String Builder
5. String Class
6. String class is final
7. String is immutable
8. All string class methods are non synchronized
9. String Buffer
10. String Buffer class is final
11. String Buffer is mutable
12. All string buffer class methods are synchronized
13. String Builder
14. String Builder class is final
15. String Builder is mutable
16. All string builder class methods are non synchronized

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **String** | **String Buffer** | **String Builder** |
| 1. | Final class | Final class | Final class |
| 2. | Immutable | Mutable | Mutable |
| 3. | Non Synchronized(Not Thread safe) | Synchronized(Thread safe) | Non Synchronized(Not Thread safe) |
| 4. | String constant pool area | Heap area | Heap area |

String str1 = new String(“JAVA”)  
String str2 = new String(“JAVA”)

**HEAP AREA**

**“JAVA”**

STR1

**“JAVA”**

**String Object**

**“JAVA”**

STR2

**STRING POOL AREA**

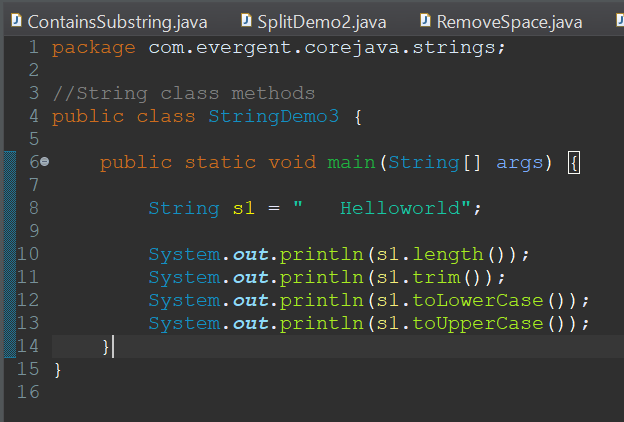
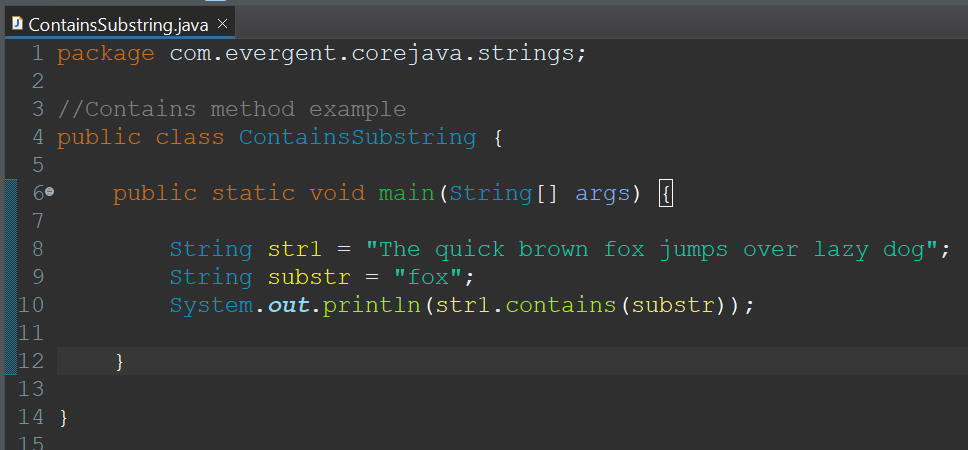
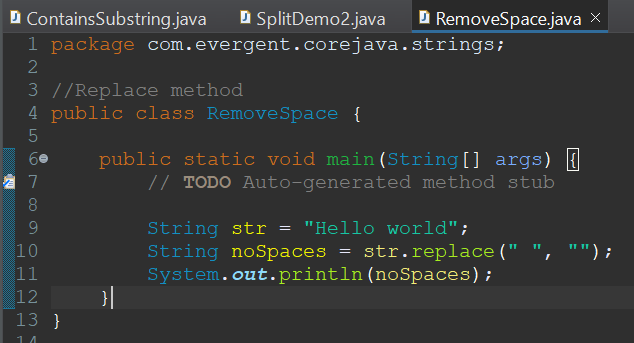
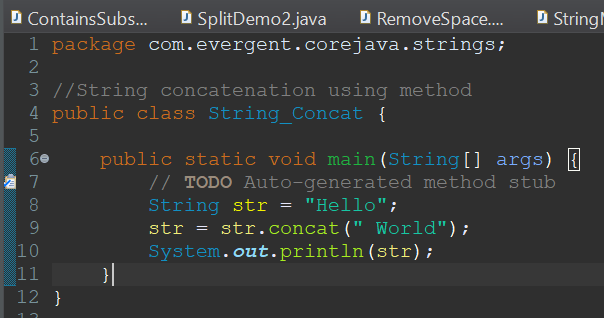
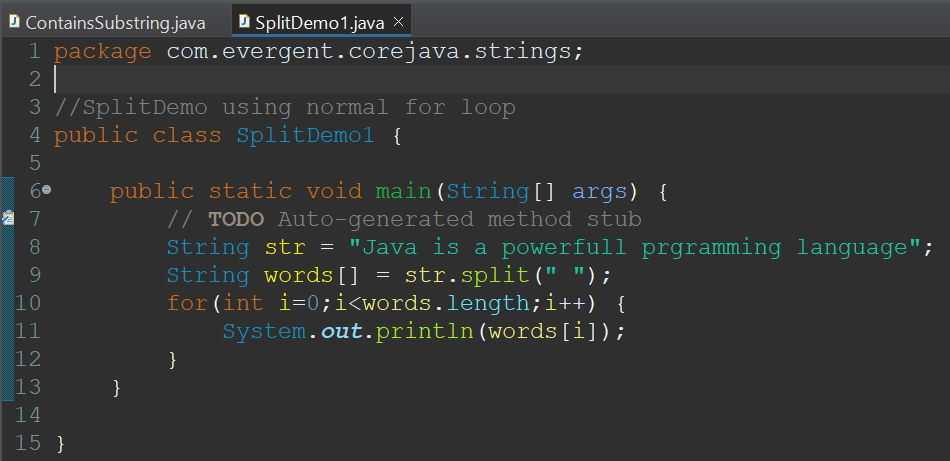
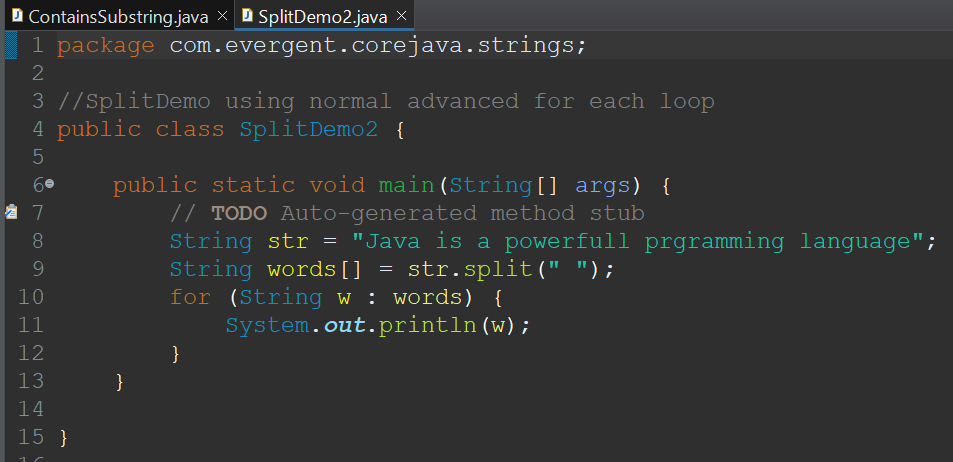
S1

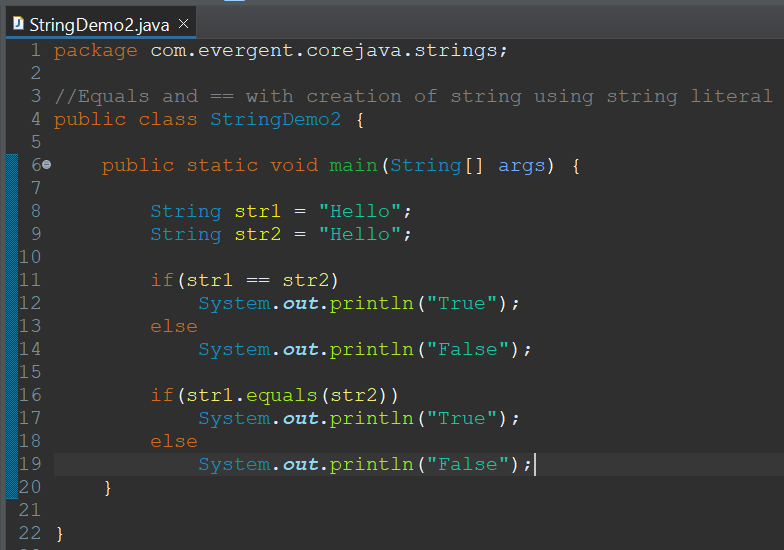
S2

**String Literal**

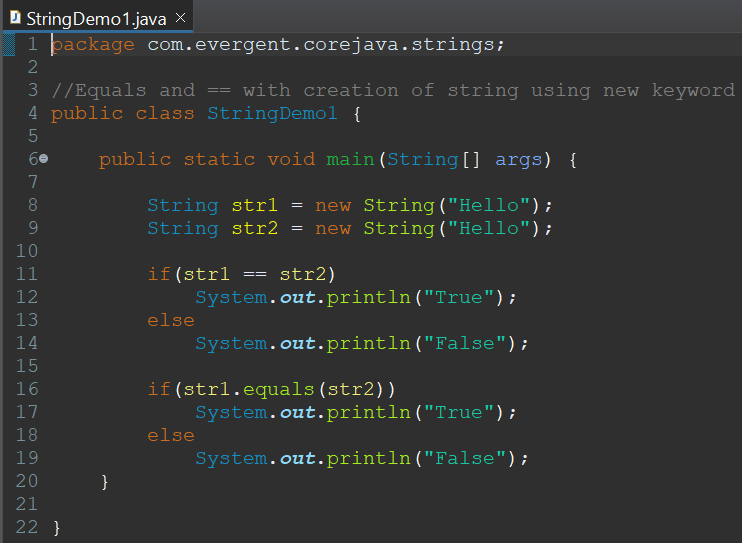
String str1 = “JAVA”  
String str2 = “JAVA”

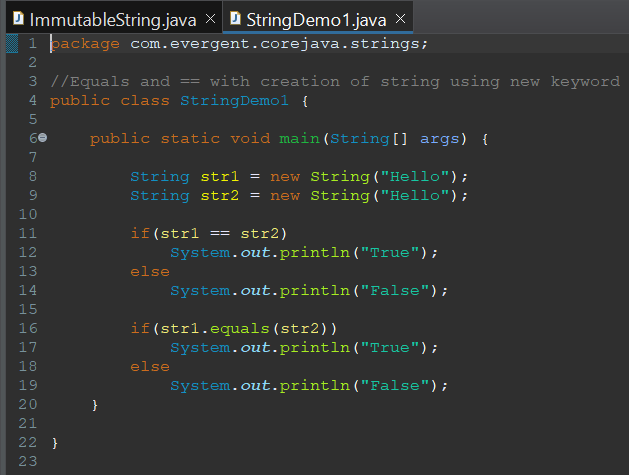
**String class points**

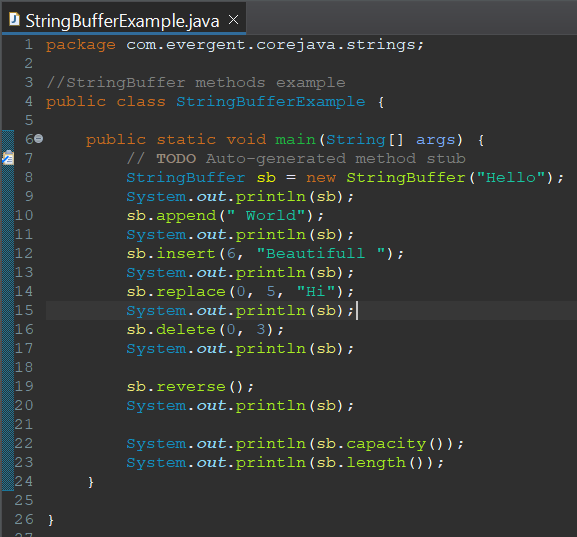
1. In java a string is a sequence of characters, often used to represent text.
2. String are objects in java & instance of string class which is part of java.lang.package.
3. Key features of string in java
   1. Immutable : Once a string object is created it cannot be changed.
   2. Any modification to string creates a new string object
4. Java optimizes memory usage by storing strings in a special way area of memory called “String Constant Pool”.
5. If two strings have same value and are created without using new keyword they will refer to same object in string pool.
6. String methods are :   
   length()  
   trim()  
   toLowerCase()  
   toUpperCase()  
   append()  
   indexOf()  
   strip()  
     
   
7. String contains() method is used to check whether substring is present or not.  
   
8. String replace() method is used to replace other values  
   
9. String reverse() method is used to reverse the string
10. String concat() method is used to reverse the string  
    
11. Split method is used to split the sentence into words.  
    I)Using for loop  
      
    II)Using for each loop  
    
12. We can create string using in two ways :
    1. Using String literals



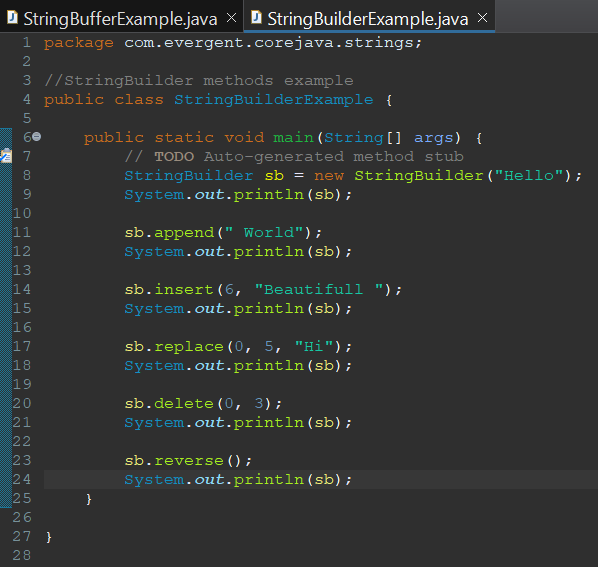
* 1. Using new keyword



1. In string
   1. equals() is used to compare the content
   2. == is used to compare the memory of the two objects  
      

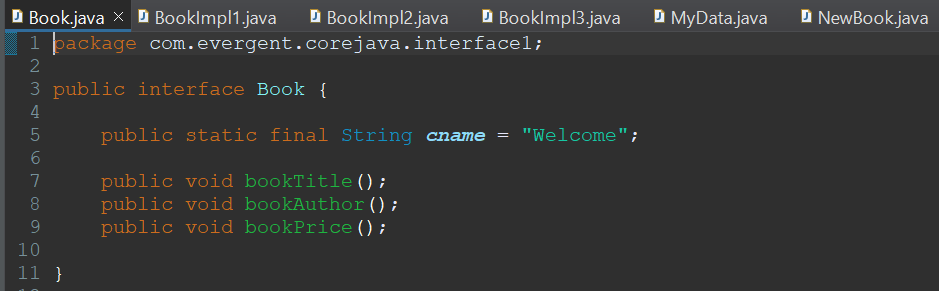
xiii) StringBuffer  
 StringBuffer is a final class.  
 StringBuffer are mutable.  
 String class consists of methods.  
 Methods in StringBuffer class are synchronized.  
1) StringBuffer methods(append()-used to combine the two strings into one String,  
Insert-to insert the string into the present string based on the index value,reverse-to reverse the string ,replace-replace the string with other string based on starting index and ending index,delete-to delete the sub string or string from the existing string based on starting index and ending index.)  


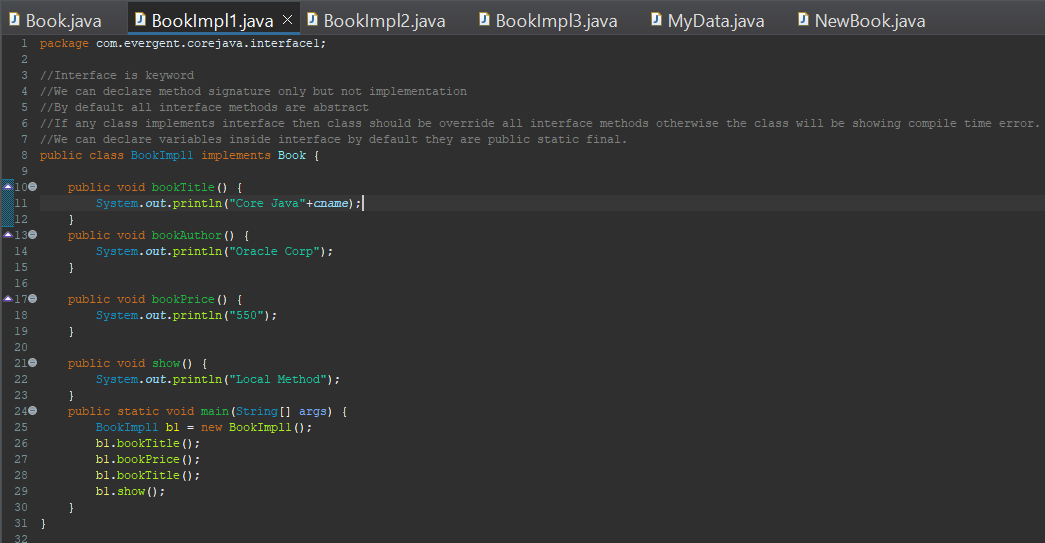
xiv) StringBuilder  
 String Builder is a final class.  
 StringBuilder are mutable.  
 StringBuilder class consists of methods.  
 Methods in String class are non-synchronized.  
2) 1)StringBuilder methods(append()-used to combine the two strings into one String,Insert-to insert the string into the present string based on the index value,reverse-to reverse the string ,replace-replace the string with other string based on starting index and ending Index,delete-to delete the substring or string from the existing string based on starting index and ending index.)

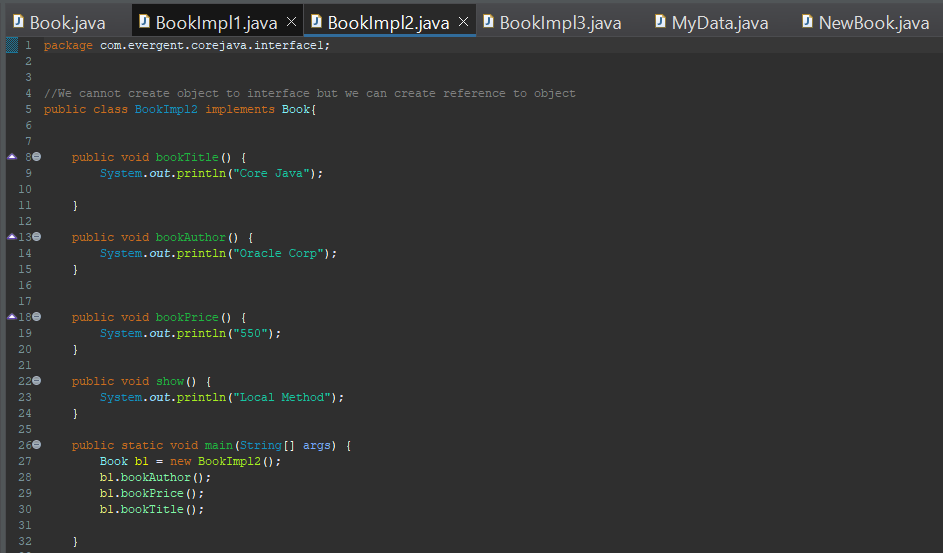


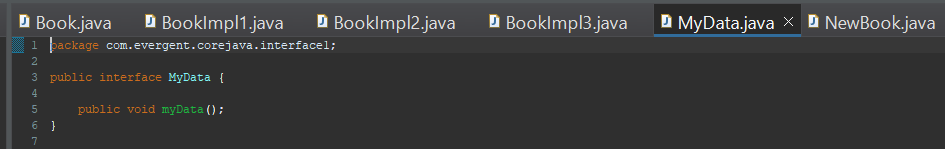
**13/08/2024**

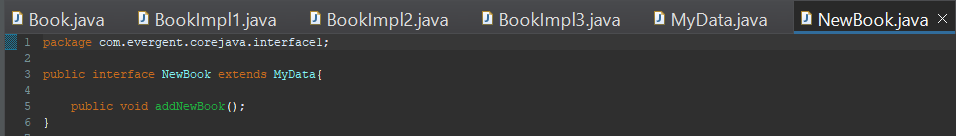
1. **Interface**
2. Interface is keyword
3. We can declare method signature only but not implementation
4. By default all interface methods are abstract
5. If any class implements interface then class should be override all interface methods otherwise the class will be showing compile time error
6. We cannot create object to interface but we can create reference to it
7. We can declare variables inside interface by default they are public static final
8. Java will support multiple inheritance through interfaces
9. One class can implement interfaces
10. One interface can extend interfaces
11. Interface can be created with out methods, these are called as Marker Interface
12. Examples of Marker Interface are cloneable, serializable

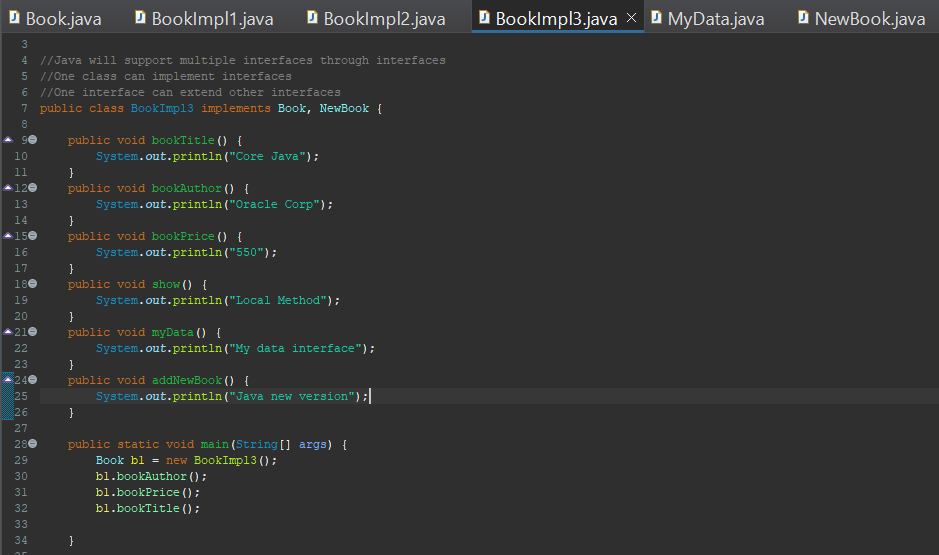






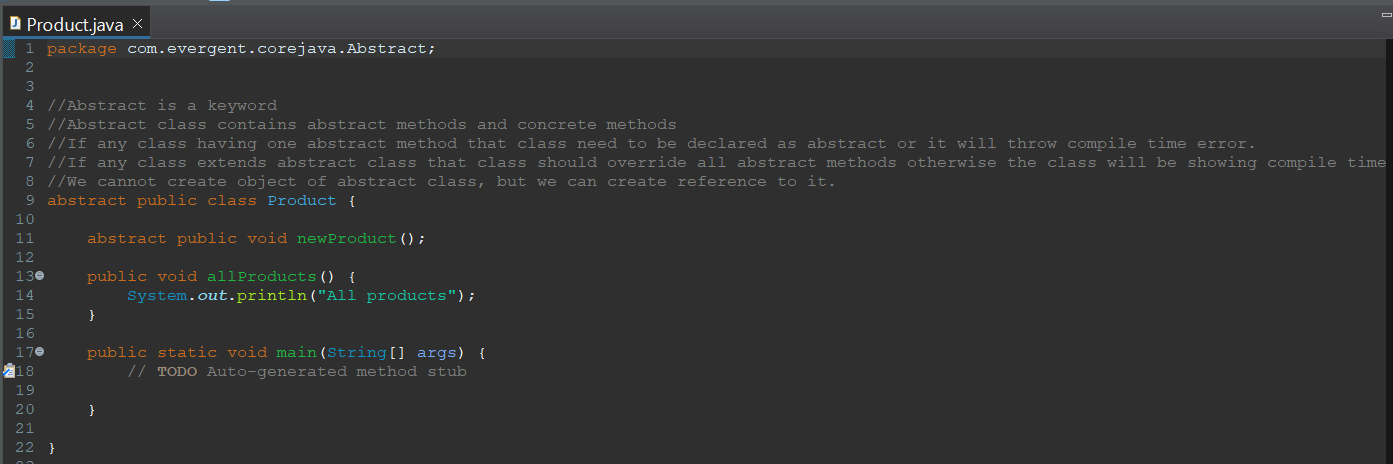




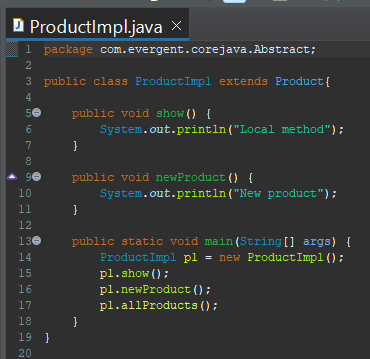


**14/08/2024**

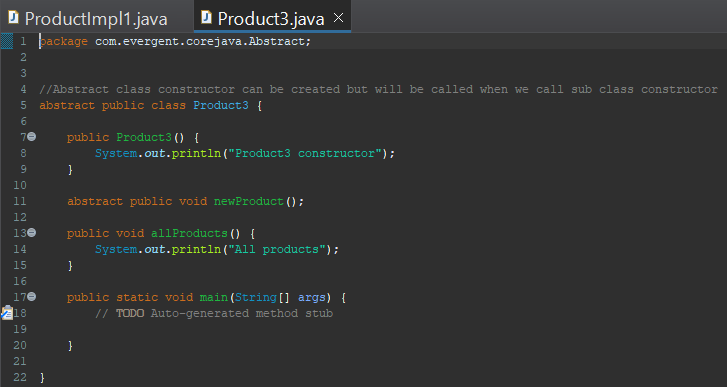
1. **Abstract**
2. Abstract is a keyword
3. Abstract class having abstract methods and concrete(implementation) methods
4. If any class having one abstract method that class should be declared as a abstract otherwise that class will be showing compile time error

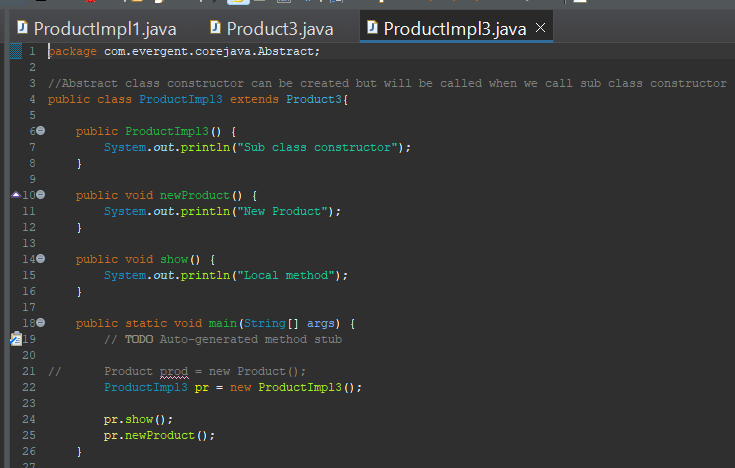


1. If any class extends abstract class that class should be override all abstract methods otherwise the class will be showing compile time error
2. We cannot create object to abstract class, but we can create reference to it.



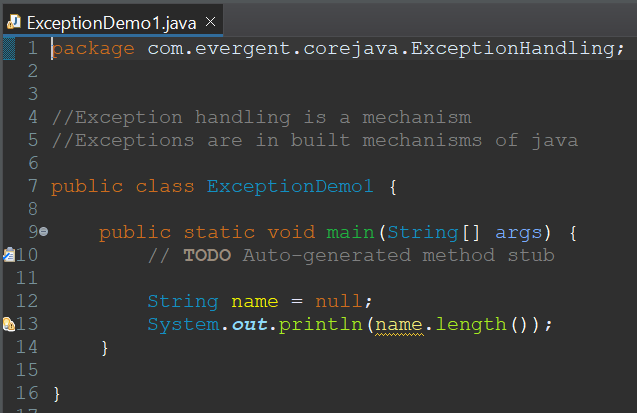
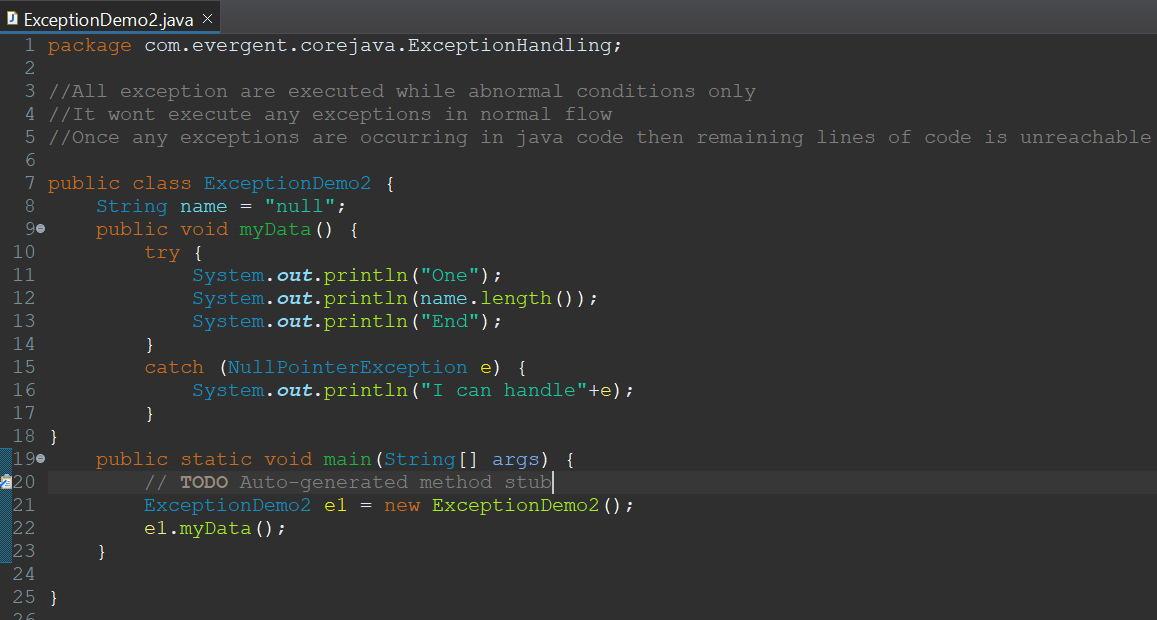
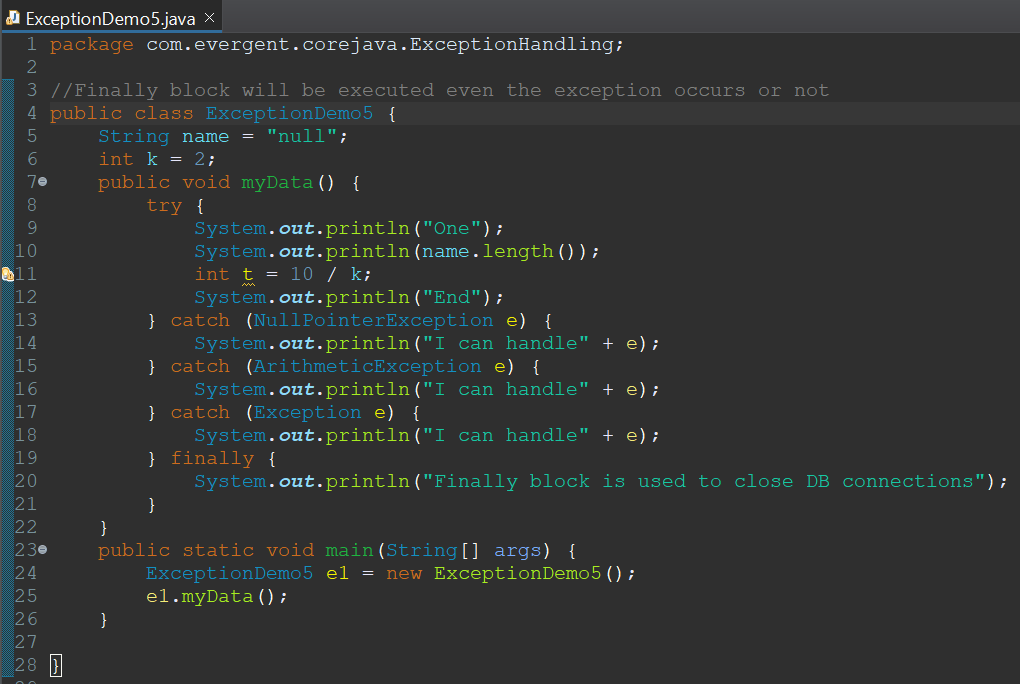
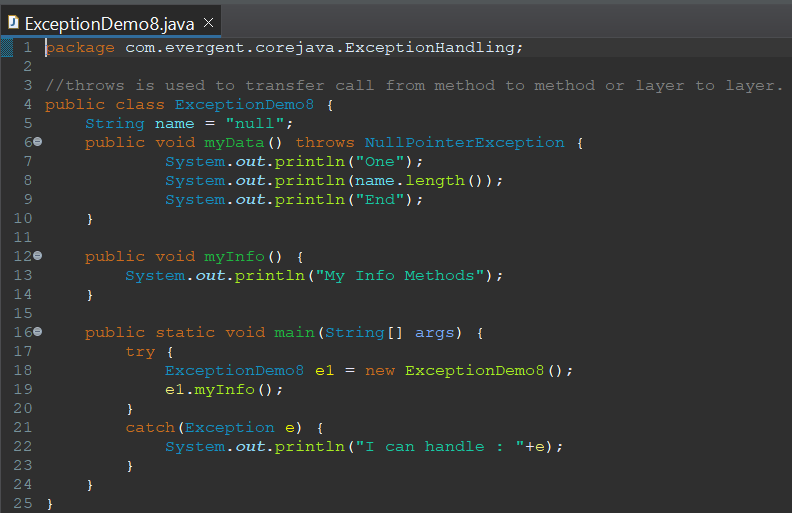
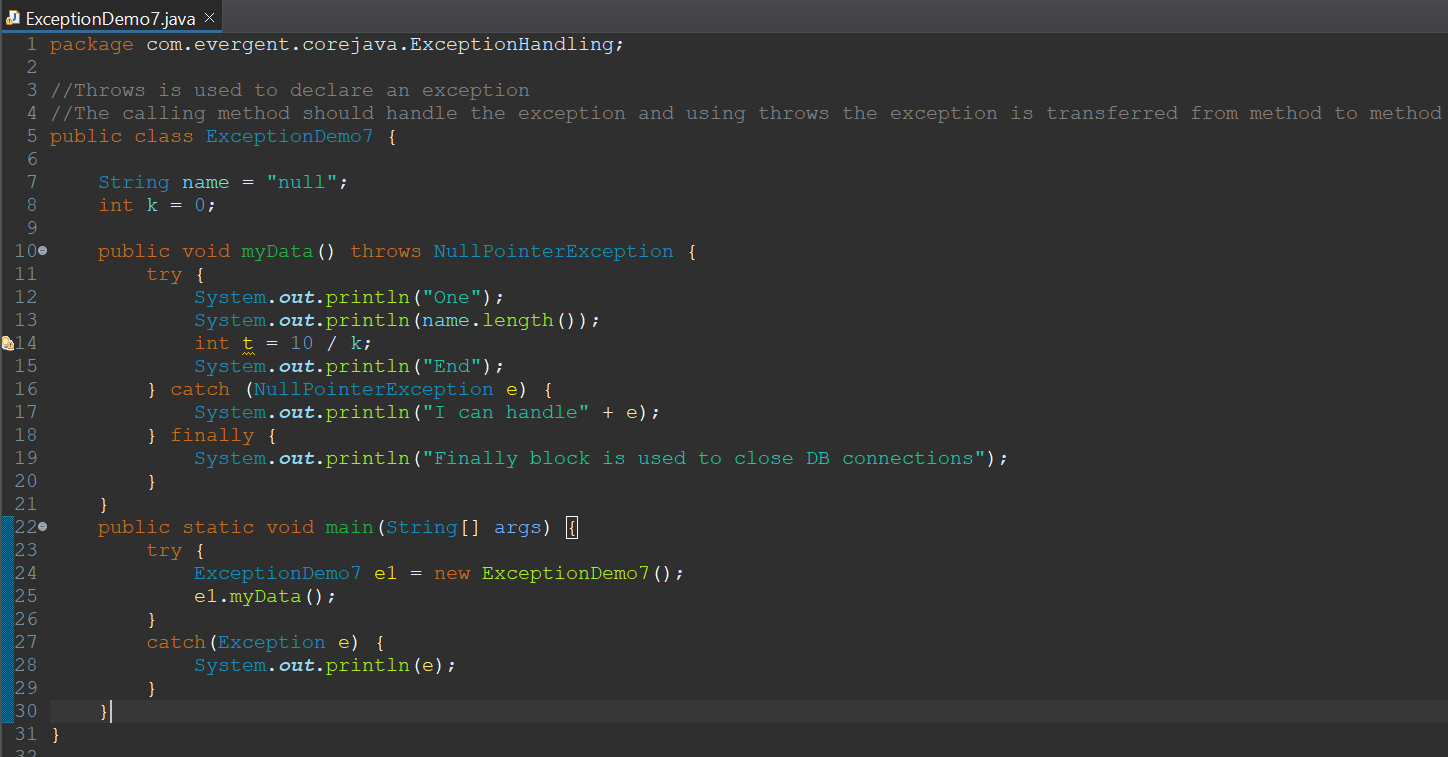
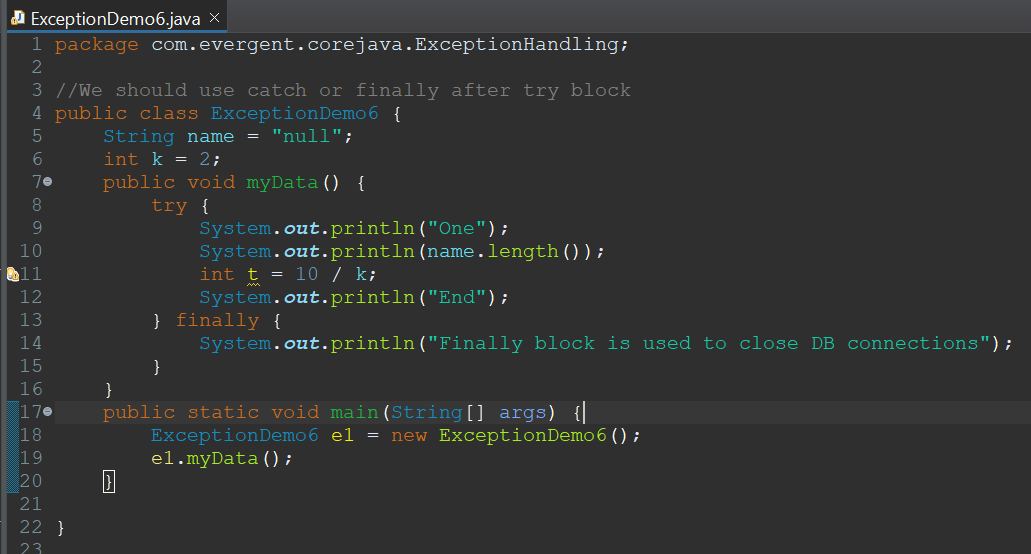
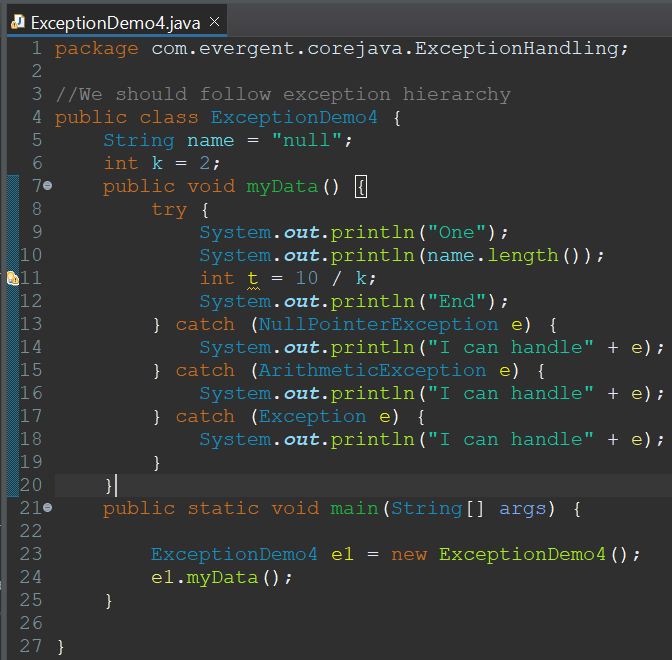
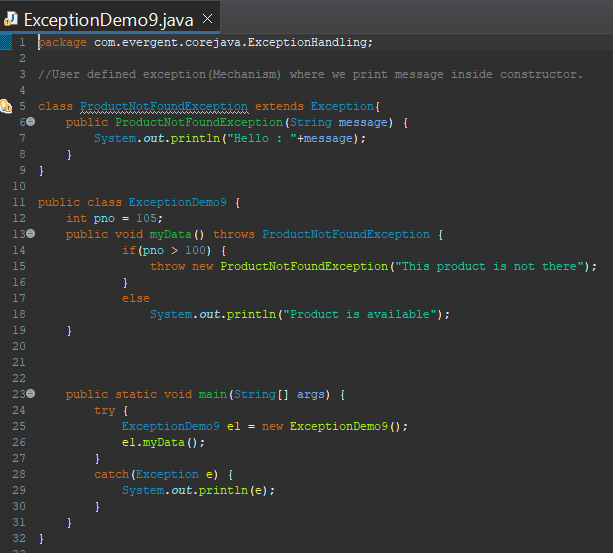
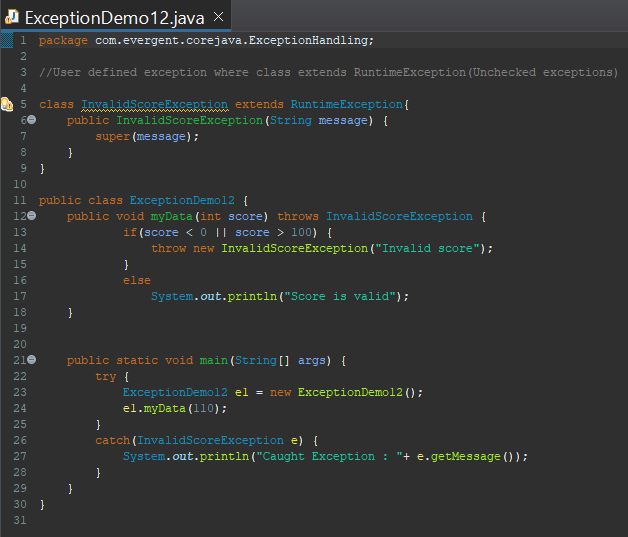
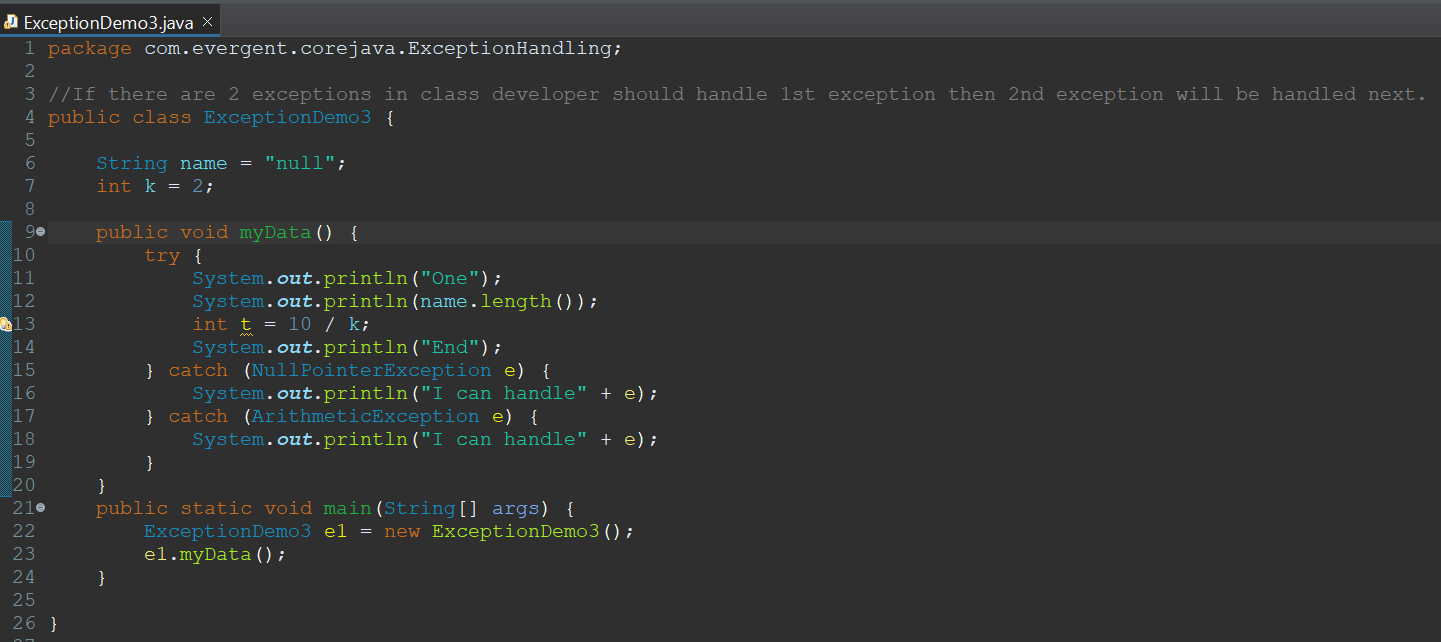
1. Abstract class constructor will be called automatically during object class creation

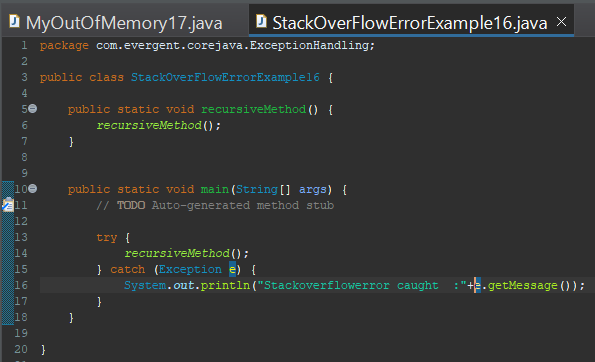


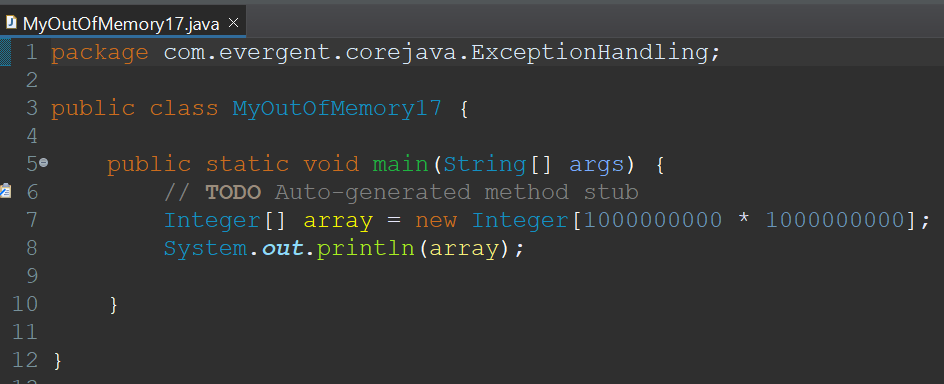


vii. Abstract class constructor can be created but it will be called when we create sub class object.

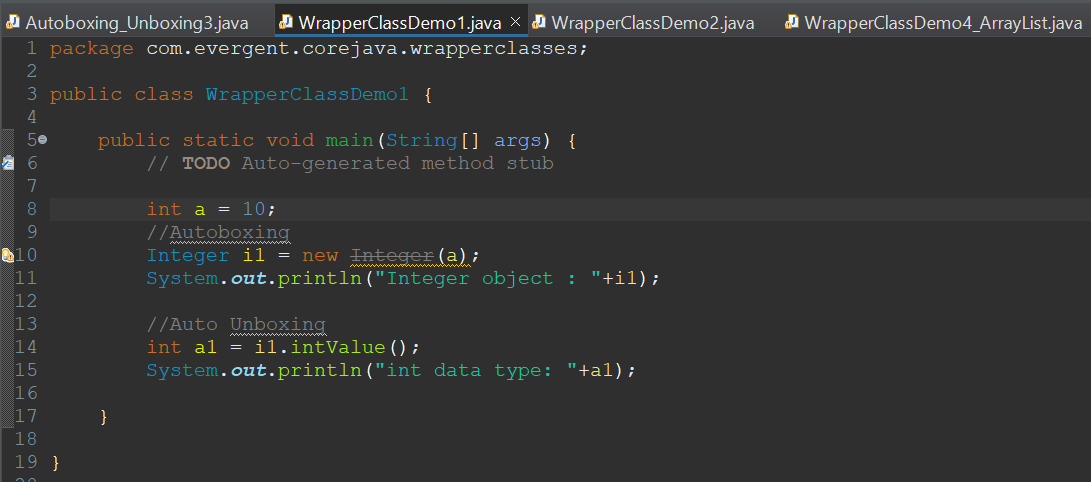
**19/08/2024**

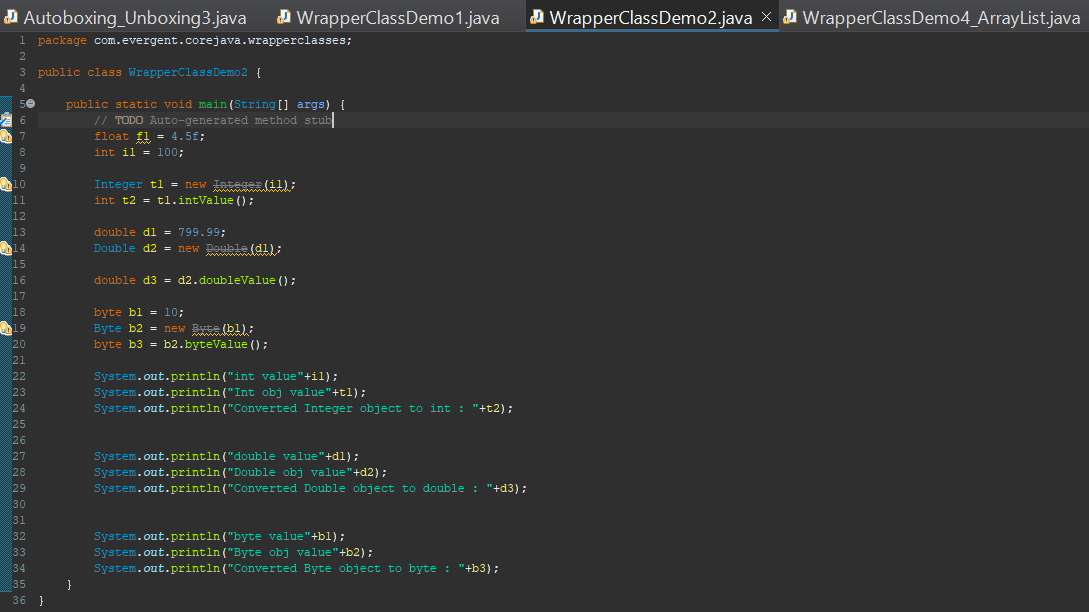
1. **Exception**
2. Exception handling is a mechanism  
   ii. Exceptions are in built mechanisms of java  
     
   iii. All exception are executed while abnormal conditions only  
   iv. It wont execute any exceptions in normal flow  
   v. Once any exceptions are occurring in java code then remaining lines of code is unreachable   
     
   vi. Java.lang.Throwable is super class for exceptions and errors  
   vii. There are 2 types of exceptions   
    I. Checked Exceptions  
    II. Unchecked Exceptions  
   viii. All checked exceptions are compile time exceptions  
   ix. All Unchecked exceptions are run time exceptions  
   x. There are 5 keywords in Exception handling  
    I. try  
    II. catch  
    III. finally  
    IV. throws  
    V. throw  
   xi. try is for business logic  
   xii. catch is for handling exceptions  
   xiii. finally is block if exception occurs or not finally block will be executed  
     
   xiv. throws an exception is transferred from method to method  
     
     
   xv. throw is for runtime exceptions will call predefined exception classes and user defined exceptions.  
   xvi. Try follwed by either catch or final block  
     
   xvii. We should follow exception hierarchy  
     
   xviii. We can create our own user defined exceptions  
     
   xix. Our own exceptions extend Exception or Runtime exceptions  
     
   xx. All exception classes into java.lang package  
   xxii. If there are 2 exceptions in class developer should handle 1st exception then 2nd exception will be handled next.  
     
   xxiii. Errors are not in developer control





1. **Wrapper Classes**Wrapping of primitive types to convert into objects  
   Collection framework only supports objects so we need wrapper classes  
   Wrapper class objects are synchronized and serializable  
   These class objects have their own methods





Auto boxing is conversion of primitive types into objects  
Unboxing is conversion of objects into primitive types

