JAVA Training Index

Day 1: 05-Aug-2024:

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

CoreJAVA-Workspace :

CoreJAVA-Development

CoreJAVA-Application

Day 2: 06-Aug-2024:

1. Logical Programming
2. Patterns
3. Switch Case
4. Arrays
    a. 1D Array's
    b. 2D Array's
5. Enum's
6. Event Management System

Day 3: 07-Aug-2024:

1. OOPS

a) Encapsulation:

i. Binding attributes and methods together inside a class and object creation is called Encapsulation.

ii. We cannot access attributes and methods without object creation. But we can access attributes and methods through object creation with reference.

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

public class Person {
    String name ="Abhi";
    int age=22;
    String address="Mahabubnagar";
    public void details() {
        System.out.println("her name is "+name+" "
        + "with age "+age+ " stays in "+address);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Person info=new Person();
        info.details();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Person [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-0

```
her name is Abhi with age 22 stays in Mahabubnagar
```

```java
package com.evergent.corejava.oops;
public class CalculationDemo {
    int a=2,b=4,c;
    public void addition(){
        c=a+b;
        System.out.println("a+b"+c);
    }
    public void sub(){
        c=a-b;
        System.out.println("a-b"+c);
    }
    public void mul(){
        c=a*b;
        System.out.println("a*b"+c);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        CalculationDemo cal = new CalculationDemo();
        cal.addition();
        cal.sub();
        cal.mul();
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> CalculationDemo [Java Application] C:\Users\abhinaya.valemo
a+b6
a-b-2
a*b8
```

b) Inheritance:

i. Re-usability of existing functionalities from super class to its subclass.

ii. Java won't support multiple inheritance through classes, but it will support through Interfaces.

```java
package com.evergent.corejava.oops;
class myPerson{
    public void personInfo() {
        String name = "Ram";
        System.out.println(name);
    }
}
class personDetails extends myPerson{
    public void personData () {
        System.out.println("With age 22");
    }
}
public class multiLevelInheritance extends personDetails  {
    public void show() {
        System.out.println("hello");
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        multiLevelInheritance mi =new multiLevelInheritance();
        mi.personInfo();
        mi.personData();
        mi.show();

    }

}
```

Problems Javadoc Declaration Console

```
<terminated> multiLevelInheritance (1) [Java Application] C:\Users\abhinaya.valemoni\D
Ram
With age 22
hello
```

c) Polymorphism:

i. Overloading:

1. Method names are same, parameters should be different, return type may or may not be same, It happens in same class or different class.

```
package com.evergent.corejava.oops;
public class Userlogin {
    public void  loginDetails() {
        System.out.println("login Details");
    }
    public void  loginDetails(String Username, String pass) {
        System.out.println("Username: "+Username);
        System.out.println("Password "+ pass);
    }
    public void  loginDetails(int mobile, String pass) {
        System.out.println("number: "+mobile);
        System.out.println("Password "+ pass);
    }
    public void  loginDetails(int mobile, String pass,String captche) {
        System.out.println("number: "+mobile);
        System.out.println("Password "+ pass);
        System.out.println("captche "+ captche);
    }
    public void show() {
        System.out.println("hii");
    }
    public static void main(String[] args) {
        Userlogin user=new Userlogin();
        user.loginDetails();
        user.loginDetails("Abhi","123");
        user.loginDetails(1234556,"123");
        user.loginDetails(1234556,"123","abv");
        user.show();
    }
```

Problems Javadoc Declaration Console

```
<terminated> Userlogin [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclip
login Details
Username: Abhi
Password 123
number: 1234556
Password 123
number: 1234556
Password 123
captche abv
hii
```

ii. Overriding:

1. Method names are same, parameters same, return type also same, It will happen in two different classes through Inheritance.

```
package com.evergent.corejava.oops;
class myBigData{
    public void myData() {
        System.out.println("Bigdata");
    }
    public void myData1() {
        System.out.println("Bigdata");
    }
}
public class methodOverriding extends myBigData {
    public void myData() {
        System.out.println("mydata");
    }
    public static void main(String[] args) {
        methodOverriding my=new methodOverriding();
        my.myData();
        my.myData1();
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> methodOverriding [Java Application] C:\Users\abhinaya.vale
mydata
Bigdata
```

d) Abstraction:

i. Hiding the irrelevant data and showing the relevant data to the end user.

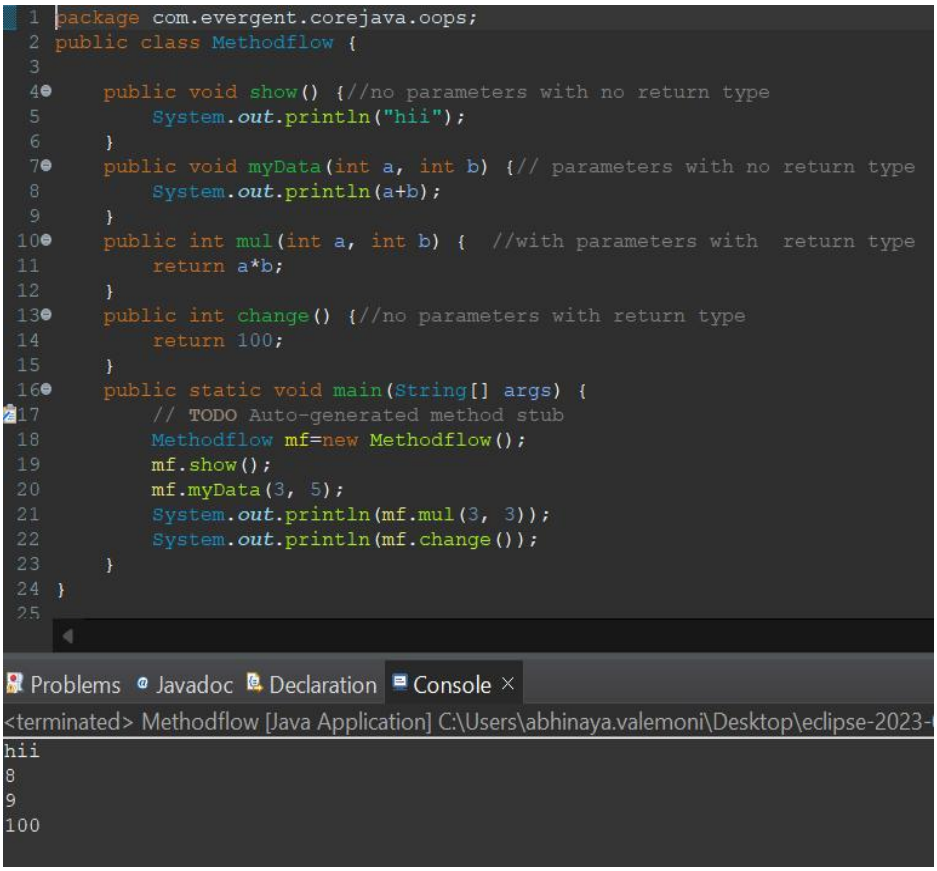
e) Method Flows:

i. With parameters, With return values.

ii. With parameters, Without return values.

iii. Without parameters, With return values.

iv. Without parameters, Without return values.



```java
package com.evergent.corejava.oops;
public class Methodflow {

    public void show() {//no parameters with no return type
        System.out.println("hii");
    }
    public void myData(int a, int b) {// parameters with no return type
        System.out.println(a+b);
    }
    public int mul(int a, int b) {  //with parameters with  return type
        return a*b;
    }
    public int change() {//no parameters with return type
        return 100;
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Methodflow mf=new Methodflow();
        mf.show();
        mf.myData(3, 5);
        System.out.println(mf.mul(3, 3));
        System.out.println(mf.change());
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Methodflow [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-

```
hii
8
9
100
```

f) IS-A and HAS-A relationship

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

public class HAS_A {
    public void mydata() {
        System.out.println("has_a");
    }
    public static void main(String[] agrs) {
        Person person=new Person();
        person.details();
        HAS_A mi=new HAS_A();
        mi.mydata();
    }

}
```

Problems Javadoc Declaration Console

```
<terminated> HAS_A [Java Application] C:\Users\abhinaya.valemoni\De
her name is Abhi with age 22 stays in Mahabubnagar
has_a
```

2. System class

a) System - is a class

b) Out - is reference of PrintStream

c) Println - method

Day 4: 08-Aug-2024:

1. Constructor:

a) Constructor is mainly used for initialization of object.

b) Class name & Constructor name should be same.

c) There are two types of constructors

i. Default constructor

ii. Parameterized constructor

d) We can access constructor while creation of object

e) Constructor doesn't have any return type not even void, if we declare as void it will consider it as a method.

f) Every class has a default constructor.

g) this, super

h) Copy constructor

i) Always constructors are overloaded

Programs:

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

public class Employee1 {
    Employee1(){
        System.out.println("Default constructor");
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Employee1();
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Employee1 [Java Application] C:\Users\abhinaya.valemoni\
Default constructor
```

```java
package com.evergent.corejava.constructor;
public class Employee2 {
    int eno;
    String ename;
    double sal;
    Employee2(){
        System.out.println("Default constructor");
    }
    Employee2(int eno1, String ename1, double sal1){
        eno=eno1;
        ename=ename1;
        sal=sal1;
    }
    public void display() {
        System.out.println(eno);
        System.out.println(ename);
        System.out.println(sal);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Employee2();
        Employee2 emp = new Employee2(730,"Abhi",500000);
        emp.display();
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Employee2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\e
Default constructor
730
Abhi
500000.0
```

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

public class Employee3 {
    int eno;
    String ename;
    double sal;
    Employee3(){
        System.out.println("Default constructor");
    }
    Employee3(int eno, String ename, double sal){
        this.eno=eno;
        this.ename=ename;
        sal=sal;
    }
    public void display() {
        System.out.println(eno);
        System.out.println(ename);
        System.out.println(sal);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Employee3();
        Employee3 emp = new Employee3(730,"Abhi",500000);
        emp.display();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> Employee3 [Java Application] C:\Users\abhinaya.valemoni\Desktop\
Default constructor
730
Abhi
0.0
```

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

public class Employee5 {
    int eno;
    String ename;
    double sal;
    Employee5(){
        System.out.println("Default constructor");
    }
    Employee5(int eno){
        this.eno=eno;
    }
    Employee5(int eno, String ename1, double sal1){
        this(eno);//call one constructor to other constructor with in class
        ename=ename1;
        sal=sal1;
    }
    public void display() {
        System.out.println(eno);
        System.out.println(ename);
        System.out.println(sal);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Employee5();
        Employee5 emp = new Employee5(730,"Abhi",500000);
        emp.display();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> Employee5 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\ecl
Default constructor
730
Abhi
500000.0
```

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

public class Employee4 {
    void Employee4(){//implicit default constructor
        System.out.println("Default constructor");
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Employee4 emp = new Employee4();
        emp.Employee4();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> Employee4 [Java Application] C:\Users\abhinaya.valemoni\
Default constructor
```

```java
package com.evergent.corejava.constructor;
//SUPER: WE'LL CALL SUPER CLASS
class MyEmployee{
    int eno;
    public MyEmployee() {
    }
    MyEmployee(int eno){
        System.out.println(eno);
    }
}
public class Employee6 extends MyEmployee {
    int eno;
    String ename;
    double sal;
    Employee6(){
        System.out.println("Default constructor");
    }
    Employee6(int eno){
        this.eno=eno;
    }
    Employee6(int eno, String ename){
        super(eno);//will send eno number to MyEmployee class
        this.ename=ename;
    }
    public void display() {
                System.out.println(ename);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Employee6();
        Employee6 emp = new Employee6(730,"Abhi");
        emp.display();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> Employee6 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclip
Default constructor
730
Abhi
```

```java
package com.evergent.corejava.constructor;
class Animal
{
    private String name;
    private int age;
    public Animal(String name, int age)//constructor
    {
        this.name=name;
        this.age=age;
    }
    public void displayInfo() //method
    {
        System.out.println("name:"+name);
        System.out.println("age:"+age);
    }
}
class Dog extends Animal //subclass inheritance
{
    private String breed;
    public Dog(String name,int age, String breed)
    {
        super(name,age);//call to super class constructor
        this.breed=breed;
    }
    public void displayInfo() //method overriding
    {
        super.displayInfo();
        System.out.println("Breed:"+breed);
    }
}
public class Inheritance_Overriding {
    public static void main(String[] args) {
        Dog dog=new Dog("Buddy",6,"Golden Retriever");
        dog.displayInfo();
    }
```

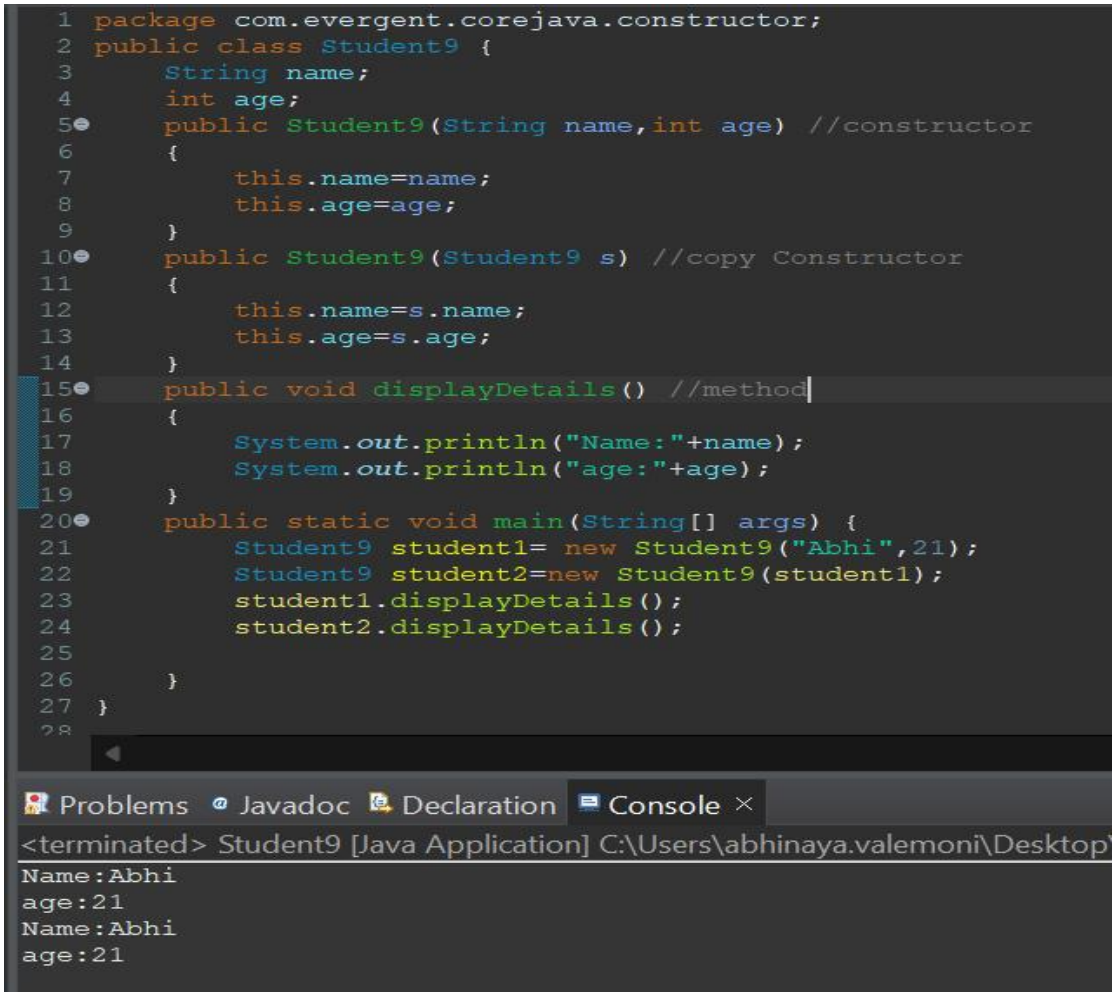
Problems Javadoc Declaration Console

```
<terminated> Inheritance_Overriding (1) [Java Application] C:\Users\abhinaya.valen
name:Buddy
age:6
Breed:Golden Retriever
```

```java
package com.evergent.corejava.constructor;
class Car{
    String color;
    int maxSpeed;
    Car(){//constructor initializing
        color="white";
        maxSpeed=120;
    }
    Car(String color, int maxSpeed){
        this.color=color;
        this.maxSpeed=maxSpeed;
    }
    void display() {
        System.out.println("color:"+color);
        System.out.println("maxSpeed:"+maxSpeed);
    }
}
public class MyCars {
    public static void main(String[] args) {
        Car car1=new Car();
        Car car2=new Car();
        car1.display();
        car2.display();
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> MyCars [Java Application] C:\Users\abhinaya.valemoni\D
color:white
maxSpeed:120
color:white
maxSpeed:120
```



```java
package com.evergent.corejava.constructor;
public class Student9 {
    String name;
    int age;
    public Student9(String name,int age) //constructor
    {
        this.name=name;
        this.age=age;
    }
    public Student9(Student9 s) //copy Constructor
    {
        this.name=s.name;
        this.age=s.age;
    }
    public void displayDetails() //method
    {
        System.out.println("Name:"+name);
        System.out.println("age:"+age);
    }
    public static void main(String[] args) {
        Student9 student1= new Student9("Abhi",21);
        Student9 student2=new Student9(student1);
        student1.displayDetails();
        student2.displayDetails();

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Student9 [Java Application] C:\Users\abhinaya.valemoni\Desktop
Name:Abhi
age:21
Name:Abhi
age:21
```

Day 5: 09-Aug-2024:

1. Static:

a) Static is a keyword.
b) We can declare static as variables, methods.
c) We can access static variables and methods directly by classname.variablename, classname.methodname.
d) Static methods can access static variables & methods.
e) Static variables cannot access non-static variables & methods.
f) Non-static methods can access static variables and methods.

```java
package com.evergent.corejava.static1;
//we can access static variable and method without obj creation
public class StaticDemo1 {
    static String name="India";
    static public void myData() {
        System.out.println("MyData");
    }
    public static void main(String[] args) {
        System.out.println(StaticDemo1.name);
        StaticDemo1.myData();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> StaticDemo1 [Java Application] C:\Users\abhinaya.valemoni\Desktop\ecli

```
India
MyData
```

```java
package com.evergent.corejava.static1;
//static method cannot access static variables and static method only
public class StaticDemo2 {
    static String cname="India";
    String name="Abhi";
    static public void myData() {
        System.out.println("MyData");
    }
    public void show() {
        System.out.println("show is non static method");
    }
    public static void main(String[] args) {
        System.out.println(cname);
        myData();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> StaticDemo2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-202

```
India
MyData
```

```java
package com.evergent.corejava.static1;
//static method cannot access static variables and static method only
public class StaticDemo3 {
    static String cname="India";
    String name="Abhi";
    static public void myData() {
        //show(); will generate error
        System.out.println("MyData");
    }
    public void show() {
        System.out.println("show is non static method");
    }
    public static void main(String[] args) {
        System.out.println(cname);
        myData();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> StaticDemo3 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-202

```
India
MyData
```

```java
package com.evergent.corejava.static1;
//non static method can access static variables and static method only
public class StaticDemo4 {
    static String cname="India";
    String name="Abhi";
    static public void myData() {
        System.out.println("MyData");
    }
    public void show() {
        myData();
        System.out.println("show is non static method");
    }
    public static void main(String[] args) {
        StaticDemo4 sd = new StaticDemo4();
        sd.show();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> StaticDemo4 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-

```
MyData
show is non static method
```

```java
package com.evergent.corejava.static1;
//non static method can access static variables and static method only
public class StaticDemo5 {
    static String cname="India";
    String name="Abhi";
    static {
        System.out.println("Abhi");
    }
    static public void myData() {
        System.out.println("MyData");
    }
    public void show() {
        myData();
        System.out.println("show is non static method");
    }
    public static void main(String[] args) {
        StaticDemo5 sd = new StaticDemo5();
        sd.show();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> StaticDemo5 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-

```
Abhi
MyData
show is non static method
```

```java
package com.evergent.corejava.static1;
public class Person6 {
    static String name="Abhi";
    int age=22;
    String address="Hyd";
    public void display() {
        name="Welcome";//static in nonb static can be accessed
        System.out.println("Name:"+name);
        System.out.println("Age:"+age);
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Person6 p1=new Person6();
        System.out.println(p1.name);
        p1.display();
        Person6 p2=new Person6();
        System.out.println(p2.name);
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Person6 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-202

```
Abhi
Name:Welcome
Age:22
Welcome
```

2. Final:

a) Final is a keyword

b) We can declare final as variables, methods and classes.

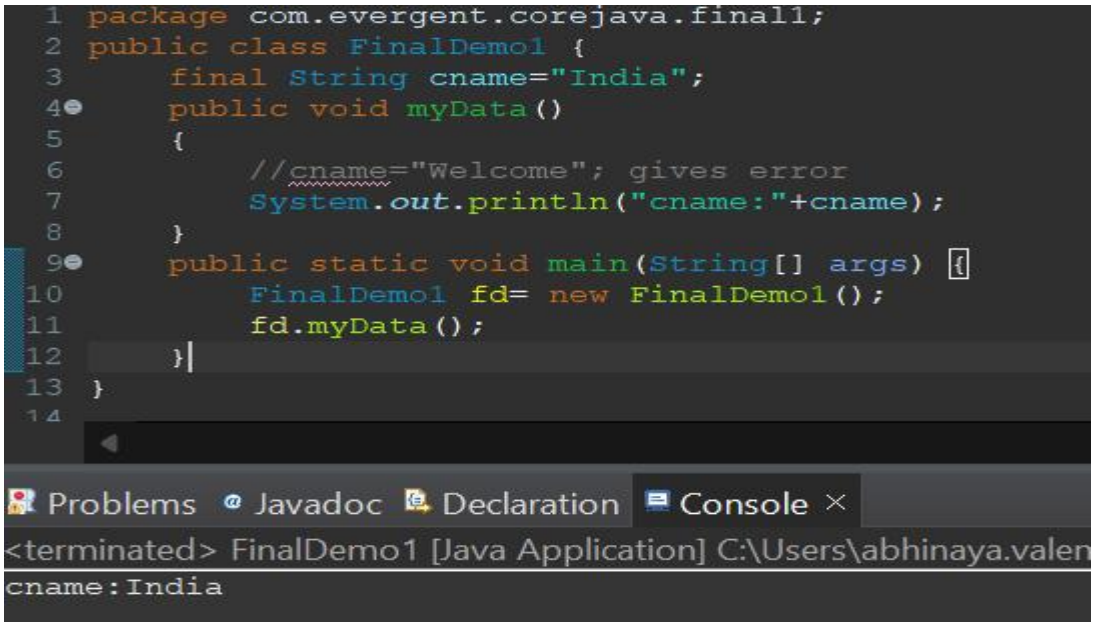
c) Final variable - we cannot modify.

d) Final methods - cannot be overridden.

e) Final class - cannot be inherited.

f) We can initialize final variables in constructors.

g) Final methods can access through HAS-A relation.



```java
package com.evergent.corejava.final1;
//we cant override final method and inherit
class MyClass
{
    final public void myProducts()
    {
        System.out.println("All products");
    }
}
public class FinalDemo2 extends MyClass{
    final String cname="India";
    //cannot overide the final method from MyClass
    public void myProducts1()
    {
        System.out.println("Hello Products");
    }
    public void myData()
    {
        System.out.println("cname:"+cname);
    }
    public static void main(String[] args) {
        FinalDemo2 fd =new FinalDemo2();
        fd.myData();
        fd.myProducts();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> FinalDemo2 [Java Application] C:\Users\abhinaya.valemor

```
cname:India
All products
```

```java
package com.evergent.corejava.final1;
//we cant override final method and inheit
final class Hello
{
    final public void myProducts()
    {
        System.out.println("All products");
    }
}
public class FinalDemo3{
    final String cname="India";
    //cannot overide the final method from MyClass
    public void myProducts1()
    {
        System.out.println("Hello Products");
    }
    public void myData()
    {
        System.out.println("cname:"+cname);
    }
    public static void main(String[] args) {
        FinalDemo3 fd =new FinalDemo3();
        Hello mc=new Hello();
        fd.myData();
        mc.myProducts();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> FinalDemo3 [Java Application] C:\Users\abhinaya.valemo

```
cname:India
All products
```

Day 6: 12-Aug-2024:

Strings

1. String class:

a) String is a sequence of characters, often used to represent text.

b) Strings are objects in java and are instances of the string class, which is part of the java.lang package.

c) Immutable: Once string object is created, it cannot be changed. Any modification to string creates a new string object.

d) Java optimizes memory usage by storing strings in a special area of memory known as string constant pool.

e) If two strings have the same value and are created without using new keyword, they will reference the same object in the string pool.

f) We can create a string in java in multiple ways:

g) Using literals:

   i. String str = “hello world”;

h) Using the new keyword:

   i. String str = new String(“hello world”);

i) String class methods:

   i. length();

   ii. toUpperCase();

   iii. toLowerCase();

   iv. trim();

   v. contains();

   vi. concat();

   vii. split();

```java
package com.evergent.corejava.Strings;
public class StringsDemo1 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        String name= new String("Java");
        String name1= new String("Java");
        if (name==name1) //its checks for memory location
            System.out.println("True");
        else
            System.out.println("False");
        if (name.equals(name1)) //its checks for content
            System.out.println("True");
        else
            System.out.println("False");
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> StringsDemo1 [Java Application] C:\Users\abhinaya.valemoni\Desk
False
True
```

```java
package com.evergent.corejava.Strings;
public class StringsDemo2 {
    public static void main(String[] args) {
        String name= "java";
        String name1= "java";
        if (name==name1) //its checks for memory location it will true with string pool constant
            System.out.println("True");
        else
            System.out.println("False");
        if (name.equals(name1)) //its checks for content
            System.out.println("True");
        else
            System.out.println("False");
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> StringsDemo2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org
True
True
```

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

public class StringDemo_methods {
    public static void main(String[] args) {
        String name=new String("  Hello world");
        System.out.println(name.length());
        System.out.println(name.toLowerCase());
        System.out.println(name.toUpperCase());
        System.out.println(name.trim());
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> StringDemo_methods [Java Application] C:\Users\abhir
13
  hello world
  HELLO WORLD
Hello world
```

```java
package com.evergent.corejava.Strings;
public class StringDemo_methods2 {
    public static void main(String[] args) {
        String str="The quick brown fffox jumps over the lazy dogs";
        String substr="fox";
        Boolean contains=str.contains(substr);
        System.out.println("contains "+substr+contains);
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> StringDemo_methods2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\
contains foxtrue
```

```java
package com.evergent.corejava.Strings;
public class StringDemo_methods3 {
    public static void main(String[] args) {
        String str="The quick brown fffoix jumps over the lazy dogs";
        String substr="fox";
        Boolean contains=str.contains(substr);
        System.out.println("contains "+substr+contains);
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> StringDemo_methods3 [Java Application] C:\Users\abhinaya.valemoni\Desktop\e
contains foxfalse
```

```java
package com.evergent.corejava.Strings;
public class StringDemo_methods4 {
    public static void main(String[] args) {
        String str="The quick brown fffoix jumps over the lazy dogs";
        String nospace=str.replace(" ","");
        System.out.println(nospace);
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> StringDemo_methods4 [Java Application] C:\Users\abhinaya.valemoni\Desktop\e
Thequickbrownfffoixjumpsoverthelazydogs
```

```java
package com.evergent.corejava.Strings;
public class StringDemo_methods5 {
    public static void main(String[] args) {
        String str="The quick brown fffoix jumps over the lazy dogs";
        StringBuilder reversed=new StringBuilder(str).reverse();
        System.out.println(reversed);
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> StringDemo_methods5 [Java Application] C:\Users\abhinaya.valemoni\Desktop\e
sgod yzal eht revo spmuj xiofff nworb kciuq ehT
```

2. StringBuffer:

a) StringBuffer is a final class.

b) StringBuffer is Mutable.

c) All methods in StringBuffer are synchronized(thread safe).

d) It is legacy API(not recommended to use).

e) StringBuffer methods are:

i. append();

ii. length();

iii. insert();

iv. replace();

v. delete();

vi. reverse();

vii. capacity();

```
1  package com.evergent.corejava.Strings;
2  public class StringBuffer_methods {
3      public static void main(String[] args) {
4          StringBuffer sb=new StringBuffer("Hello");
5          System.out.println("original String:"+sb);
6          System.out.println("append String:"+sb.append("World!"));
7          System.out.println("insert String:"+sb.insert(7,"beautiful"));
8          System.out.println("replace String:"+sb.replace(0,5,"hi"));
9          System.out.println("delete String:"+sb.delete(0,3));
10         System.out.println("reverse String:"+sb.reverse());
11         System.out.println("capacity String:"+sb.capacity());
12         System.out.println("length String:"+sb.length());
13     }
14 }
15
```

Problems Javadoc Declaration Console ×

<terminated> StringBuffer_methods [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2

```
original String:Hello
append String:HelloWorld!
insert String:HelloWobeautifulrld!
replace String:hiWobeautifulrld!
delete String:obeautifulrld!
reverse String:!dlrlufituaebo
capacity String:21
length String:14
```

3. StringBuilder:

a) StringBuilder is final class.

b) StringBuilder is Mutable.

c) All methods in StringBuffer are non-synchronized.

d) StringBuffer methods are:

i. append();

ii. length();

iii. insert();

iv. replace();

v. delete();

vi. reverse();

vii. capacity();

```
1  package com.evergent.corejava.Strings;
2  public class StringBuilder_methods {
3      public static void main(String[] args) {
4          StringBuffer sb=new StringBuffer("Hello");
5          System.out.println("original String:"+sb);
6          System.out.println("append String:"+sb.append("World!"));
7          System.out.println("insert String:"+sb.insert(7,"beautiful"));
8          System.out.println("replace String:"+sb.replace(0,5,"hi"));
9          System.out.println("delete String:"+sb.delete(0,3));
10         System.out.println("reverse String:"+sb.reverse());
11     }
12 }
```

Problems Javadoc Declaration Console

```
<terminated> StringBuilder_methods [Java Application] C:\Users\abhinaya.valemoni\Desktop\ecli
original String:Hello
append String:HelloWorld!
insert String:HelloWobeautifulrld!
replace String:hiWobeautifulrld!
delete String:obeautifulrld!
reverse String:!dlrlufituaebo
```

Day 7: 13-Aug-2024:

1. Interfaces:

a) Interface is a keyword.
b) We can declare method signature only but not implementation.
c) By default all interface methods are abstract.
d) If any class implements the interface then that class should be override all interface method, otherwise that class will be showing compile time error.
e) We cannot create object to interface but we can create reference to interface.
f) We can declare variables inside interface - by default (public static final) variables.
g) JAVA will support multiple inheritance through interface.
h) One class can implement interfaces.
i) One interface extends other interfaces.
j) We can declare interfaces without methods is called Marker Interfaces.
k) Example:
    i. clonable
    ii. Serializable

```java
package com.evergent.corejava.interfaces;
//methods signature
public interface Book {
    public void booktitle();
    public void bookauthor();
    public void bookprice();
}
```

```java
package com.evergent.corejava.interfaces;
public interface NewBook extends MyData {
    public void addnewBook();
}
```

```java
package com.evergent.corejava.interfaces;
public interface MyData {
    public void myData();
}

```

```java
package com.evergent.corejava.interfaces;
//If nay class implements interface that class should ne override all interface methods
public class Bookimpl implements Book {
    public void booktitle() {
        System.out.println("Java Core");
    }
    public void bookauthor() {
        System.out.println("Oracle Crop");
    }
    public void bookprice() {
        System.out.println("550");
    }
    public void show() {
        System.out.println("hiii");
    }
    public static void main(String[] args) {
        Bookimpl bi=new Bookimpl();
        bi.bookauthor();
        bi.bookprice();
        bi.booktitle();
        bi.show();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Bookimpl (1) [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\pl

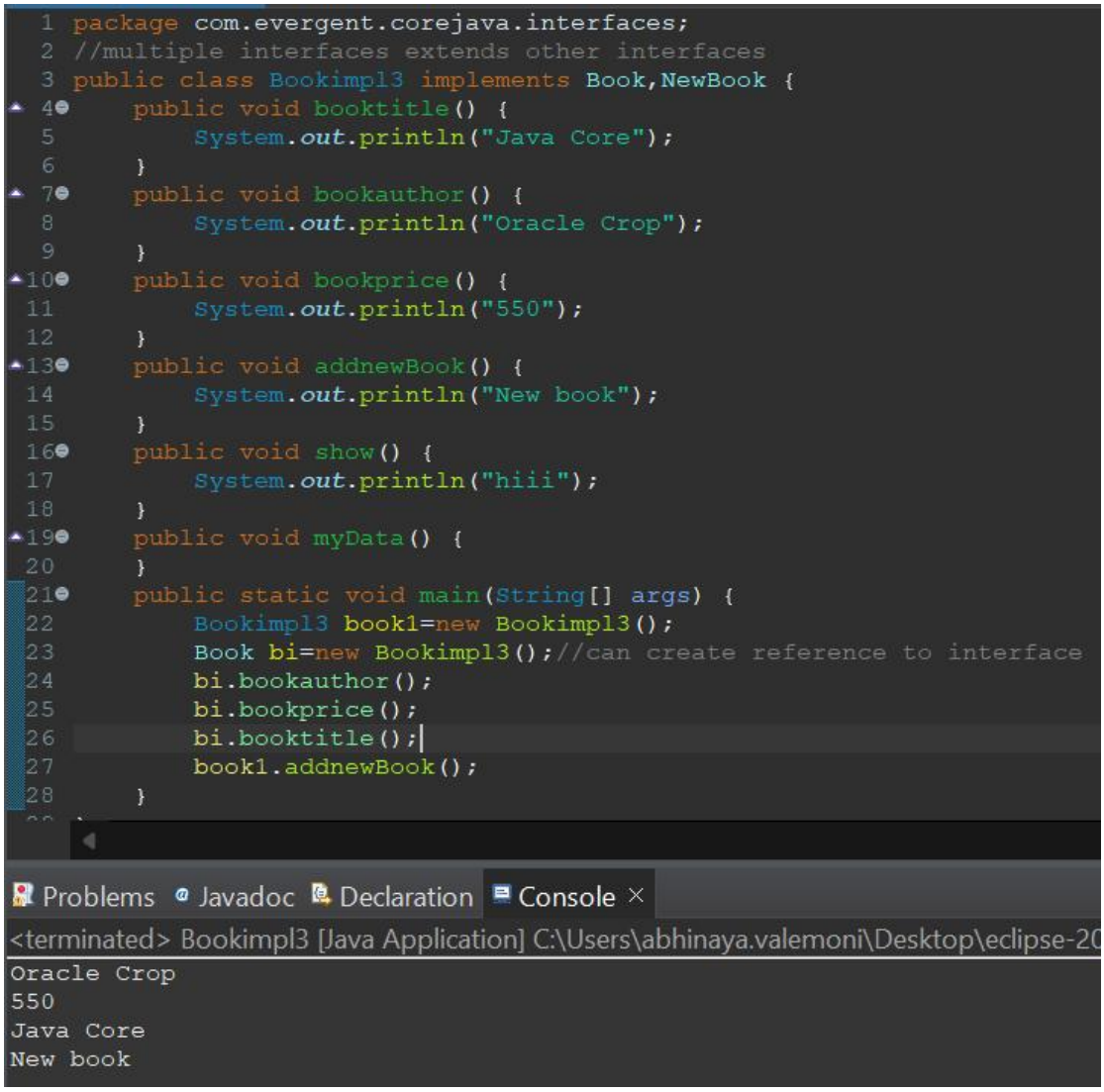
```
Oracle Crop
550
Java Core
hiii
```

```java
package com.evergent.corejava.interfaces;
//If nay class implements interface that class should ne override all interface methods
public class Bookimpl2 implements Book {
    public void booktitle() {
        System.out.println("Java Core");
    }
    public void bookauthor() {
        System.out.println("Oracle Crop");
    }
    public void bookprice() {
        System.out.println("550");
    }
    public void show() {
        System.out.println("hiii");
    }
    public static void main(String[] args) {
        //we cannot create object to interface
        //Book bi=new Book();
        Book bi=new Bookimpl2();//can create reference to interface
        bi.bookauthor();
        bi.bookprice();
        bi.booktitle();
        //interface reference cannot visible for subclass
        //bi.show();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Bookimpl2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\p

```
Oracle Crop
550
Java Core
```



```java
package com.evergent.corejava.interfaces;
//multiple interfaces extends other interfaces
public class Bookimpl3 implements Book,NewBook {
    public void booktitle() {
        System.out.println("Java Core");
    }
    public void bookauthor() {
        System.out.println("Oracle Crop");
    }
    public void bookprice() {
        System.out.println("550");
    }
    public void addnewBook() {
        System.out.println("New book");
    }
    public void show() {
        System.out.println("hiii");
    }
    public void myData() {
    }
    public static void main(String[] args) {
        Bookimpl3 book1=new Bookimpl3();
        Book bi=new Bookimpl3();//can create reference to interface
        bi.bookauthor();
        bi.bookprice();
        bi.booktitle();
        book1.addnewBook();
    }
```

Problems Javadoc Declaration Console ×

<terminated> Bookimpl3 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2

```
Oracle Crop
550
Java Core
New book
```

Day 8: 14-Aug-2024:

1. Abstract class:

a) Abstract is a keyword.
b) Abstract class can have abstract methods and concrete methods.
c) If any class having one abstract method that class should be declared as a abstract keyword, otherwise that class will be showing compile time error.
d) If any class extends abstract class that class should override all abstract methods, otherwise the class will be showing compile time error.

e) We cannot create object to abstract class but we can create reference to abstract class.
f) We have to declare abstract methods explicitly.
g) We can create constructor to abstract class.
h) We can access abstract class constructor through the sub-class object creation.

```java
package com.evergent.corejava.abtractclass;
abstract public class Product {//The type Product must be an abstract class to define abstract methods
    abstract public void NewProduct();
    //public void NewProduct(); This method requires a body instead of a semicolon
    public void allproducts() {
        System.out.println("All Products");
    }
}

```

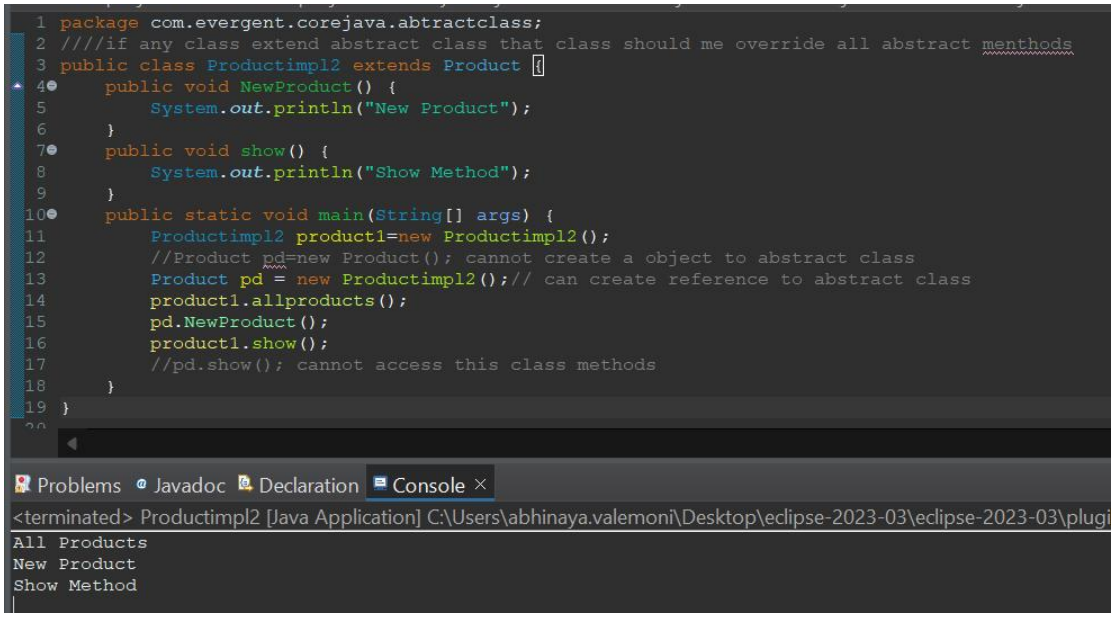
```java
package com.evergent.corejava.abtractclass;
abstract public class Product2 {//The type Product must be an abstract class to define abstract methods
    public Product2() { //constructor
        System.out.println("hello");
    }
    abstract public void NewProduct();
    //public void NewProduct(); This method requires a body instead of a semicolon
    public void allproducts() {
        System.out.println("All Products");
    }
}

```

```java
package com.evergent.corejava.abtractclass;
////if any class extend abstract class that class should me override all abstract menthods
public class Productimpl extends Product2 {
    public Productimpl() {
        System.out.println("hru");
    }
    public void NewProduct() {
        System.out.println("New Product");
    }
    public void show() {
        System.out.println("Show Method");
    }
    public static void main(String[] args) {
        Productimpl product1=new Productimpl();
        //Product pd=new Product(); cannot create a object to abstract class
        Product2 pd = new Productimpl();// can create reference to abstract class
        product1.allproducts();
        pd.NewProduct();
        product1.show();
        //pd.show(); cannot access this class methods
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Productimpl (1) [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\pl

```
hello
hru
hello
hru
All Products
New Product
Show Method
```



```
package com.evergent.corejava.abtractclass;
////if any class extend abstract class that class should me override all abstract menthods
public class Productimpl2 extends Product {
    public void NewProduct() {
        System.out.println("New Product");
    }
    public void show() {
        System.out.println("Show Method");
    }
    public static void main(String[] args) {
        Productimpl2 product1=new Productimpl2();
        //Product pd=new Product(); cannot create a object to abstract class
        Product pd = new Productimpl2();// can create reference to abstract class
        product1.allproducts();
        pd.NewProduct();
        product1.show();
        //pd.show(); cannot access this class methods
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Productimpl2 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plugi
All Products
New Product
Show Method
```

Day 9: 16-Aug-2024:

1. Java Concept Test
2. Application design and code

Day 10: 19-Aug-2024 and Day 11: 20-Aug-2024:

1. Exception Handling:

2. Exception handling is mechanism
3. Exceptions are inbuilt mechanism of JAVA
4. All exceptions are executed while abnormal conditions
5. Normal flow it won't execute any exception
6. Once any exceptions are occurring in java then remaining lines of code are unreachable.
7. Java.lang.Throwable is super class for exception and error
8. There are two types of exceptions in JAVA:

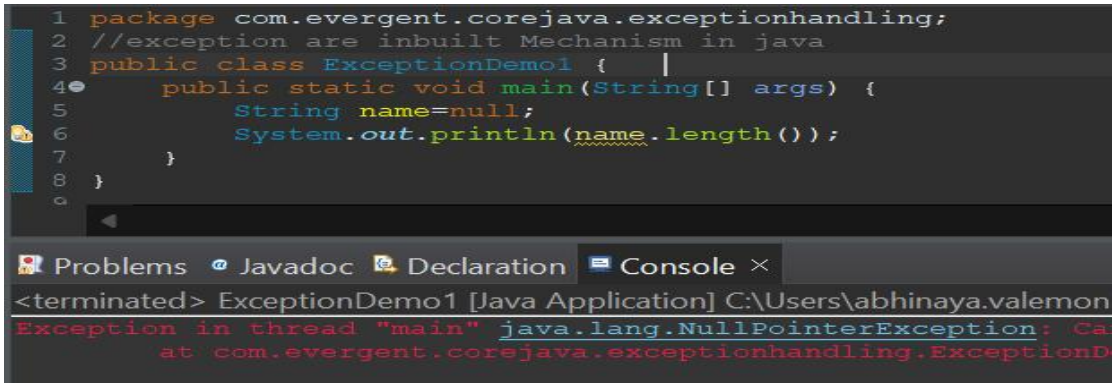
Checked exception
Unchecked exception

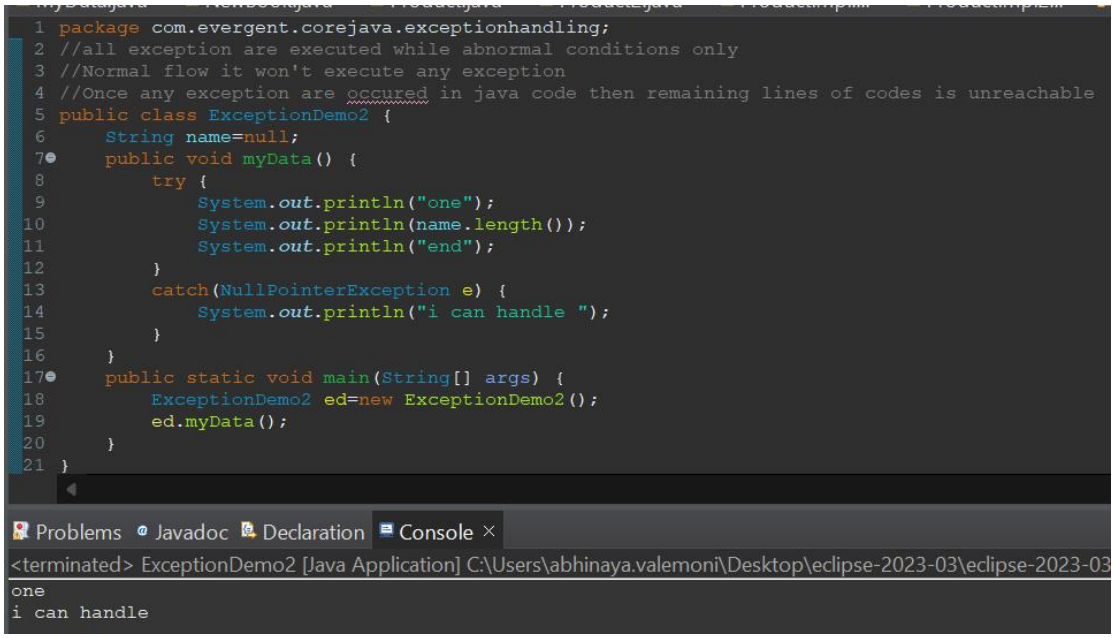
9. All checked exceptions are compile time exceptions
10. All unchecked exceptions are run time exceptions
11. There are 5 keywords in exception handling:

try
catch()
finally()
throws

throw

12. Try is for business logic
13. Catch is for handling exceptions
14. Finally block, is executed if exception occurs or not
15. Throws an exception will be executed method by method
16. Throw is for run time exceptions & will call predefined exceptions
17. Try followed by either catch or finally block
18. We should follow exception hierarchical
19. We can create our own (user-defined) exceptions
20. Our own exceptions extends exception or run time exception
21. All exception classes are in to java.lang package
22. There is two exceptions in class, developer should be handle 1st exception then after 2nd exception will be handled.
23. Errors cannot be controlled





```java
package com.evergent.corejava.exceptionhandling;
//multiple exception
//if 2 exception occurs then 1st exception will comes at prior and stops executing
public class ExceptionDemo3 {
    String name=null;
    int k=0;
    public void myData()
    {
        try
        {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            System.out.println("End");
        }
        catch(ArithmeticException e)
        {
            System.out.println("I can handle:"+e);
        }
            catch(NullPointerException e)
            {
                System.out.println("I can handle:"+e);
            }
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        ExceptionDemo3 ed3=new ExceptionDemo3();
        ed3.myData();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> ExceptionDemo3 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.
one
I can handle:java.lang.NullPointerException: Cannot invoke "String.length()" because "this.name" is null
```

```java
package com.evergent.corejava.exceptionhandling;
//should follow exception hierarchial
public class ExceptionDemo4 {
    String name=null;
    int k=0;
    public void myData()
    {
        try
        {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            System.out.println("End");
        }
        /*catch(exception e) {
            System.out.println(e);
        }*/
            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(e);
        }
    }
    public static void main(String[] args) {
        ExceptionDemo4 ed3=new ExceptionDemo4();
        ed3.myData();
    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> ExceptionDemo4 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plugins\org.
one
I can handle:java.lang.NullPointerException: Cannot invoke "String.length()" because "this.name" is null
```

```java
package com.evergent.corejava.exceptionhandling;
//exception occur or not finally block will get executed
public class ExceptionDemo5finally {
    String name=null;
    int k=0;
    public void myData()
    {
        try
        {
            System.out.println("one");
            System.out.println(name.length());
            int t=10/k;
            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(e);
        }
        finally {
            System.out.println("finally block");
        }

    }
    public static void main(String[] args) {
        ExceptionDemo5finally ed3=new ExceptionDemo5finally();
        ed3.myData();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> ExceptionDemo5finally [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plugi

```
one
I can handle:java.lang.NullPointerException: Cannot invoke "String.length()" because "this.name" is null
finally block
```

```java
package com.evergent.corejava.exceptionhandling;
//try should be followed by either catch or finally block
public class ExceptionDemo6 {
    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("End");
        }
        finally {
            System.out.println("finally block");
        }
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        ExceptionDemo6 ed3=new ExceptionDemo6();
        ed3.myData();
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> ExceptionDemo6 [Java Application] C:\Users\abhinaya.valemoni\D

```
one
4
End
finally block
```

```java
package com.evergent.corejava.exceptionhandling;
//try should be followed by either catch or finally block
public class ExceptionDemo7 {
    String name=null;
    int k=0;

    public void myInfo() throws NullPointerException
    {
        System.out.println(name.length());
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        try {
            ExceptionDemo7 ed3=new ExceptionDemo7();

            ed3.myInfo();
        }
        catch(NullPointerException e) {
            System.out.println(e);
        }

    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> ExceptionDemo7 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-0
java.lang.NullPointerException: Cannot invoke "String.length()" because "this.name" is null
```

```java
package com.evergent.corejava.exceptionhandling;
public class ExceptionDemo8 {
    String name="";
    int k=0;
    public void myInfo() throws NullPointerException
    {
        System.out.println(name.length());
    }
    public void myData() throws ArithmeticException
    {
            System.out.println("one");
            int t=10/k;
            System.out.println("End");
    }
    public static void main(String[] args) {
        try {
            ExceptionDemo8 ed3=new ExceptionDemo8();
            ed3.myData();
            ed3.myInfo();
        }
        catch(ArithmeticException e) {
            System.out.println(e);
        }
    }
```

Problems Javadoc Declaration Console ×

```
<terminated> ExceptionDemo8 [Java Application] C:\Users\abhinaya.valemo
one
java.lang.ArithmeticException: / by zero
```

```java
package com.evergent.corejava.exceptionhandling;
class ProductNotFound extends Exception{
    public ProductNotFound(String message) {
        System.out.println("hello "+ message);
    }
}
public class ProductImpl9 {
    public void myData() throws ProductNotFound{
        int k=105;
        if(k>100) {
            throw new ProductNotFound("hello");
        }
        else {
            System.out.println("hii");
        }
    }
    public static void main(String[] args) {
        try {
            ProductImpl9 pd =new ProductImpl9();
            pd.myData();
        }
        catch(Exception e) {
            System.out.println("i can handle"+e);
        }
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> ProductImpl9 [Java Application] C:\Users\abhinaya.valemoni\Desktop\ecl
hello hello
i can handlecom.evergent.corejava.exceptionhandling.ProductNotFound
```

```java
package com.evergent.corejava.exceptionhandling;
class AgeNotSupport extends Exception{
    public AgeNotSupport(String message) {
        System.out.println("hello "+ message);
    }
}
public class ProductImpl10 {
    public void myData() throws AgeNotSupport{
        int age=17;
        if(age<18) {
            throw new AgeNotSupport("U're age is not elegible to vote");
        }
        else {
            System.out.println("u can vote");
        }
    }
    public static void main(String[] args) {
        try {
            ProductImpl10 pd =new ProductImpl10();
            pd.myData();

        }
        catch(Exception e) {
            System.out.println("i can handle"+e);
        }
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> ProductImpl10 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-202
hello U're age is not elegible to vote
i can handlecom.evergent.corejava.exceptionhandling.AgeNotSupport
```

```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("Insufficient funds for withdrawl");
        }
        else{
            System.out.println("withdrawl succesful");
        }
    }
    public static void main(String[] args) {
        try{
            withDraw(600.00);
        }
        catch(InsufficientFundsException e){
            System.out.println("Caught the exception:"+e.getMessage());
            System.out.println(e);
        }
        System.out.println("Program  continues after handling the exception");
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> UserDefinedExceptionDemo11 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-20

```
Caught the exception:Insufficient funds for withdrawl
com.evergent.corejava.exceptionhandling.InsufficientFundsException: Insufficient funds for withdrawl
Program  continues after handling the exception
```

```java
package com.evergent.corejava.exceptionhandling;
class InvalidScoreException extends RuntimeException{
    public InvalidScoreException(String message){
        super(message);
    }
}
public class UserDefinedExceptionDemo12 {
    public static void validScore(int score) throws InvalidScoreException{
        if(score<0||score>100){
            throw new InvalidScoreException("Score must be b/w 0 and 100");
        }
        else{
            System.out.println("score is valid");
        }
    }
    public static void main(String[] args) {
        try {
            validScore(110);
        }
        catch(InvalidScoreException e){
            System.out.println("Caught the exception:"+e.getMessage());
            System.out.println(e);
        }
        System.out.println("Program  continues after handling the exception");
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> UserDefinedExceptionDemo12 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-

```
Caught the exception:Score must be b/w 0 and 100
com.evergent.corejava.exceptionhandling.InvalidScoreException: Score must be b/w 0 and 100
Program  continues after handling the exception
```

```java
package com.evergent.corejava.exceptionhandling;
public class ArrayOfIndex13 {
    public static void main(String[] args) {
        int[] numbers= {1,2,3,4,5};
        try {
            System.out.println("Accesing element at index 10: "+numbers[10]);
        }
        catch(ArrayIndexOutOfBoundsException e){
            System.out.println("Caught an exception "+e.getMessage());
        }
        System.out.println("Program continues after exception handling");

    }
}
```

Problems Javadoc Declaration Console ×

```
<terminated> ArrayOfIndex13 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\ecl
Caught an exception Index 10 out of bounds for length 5
Program continues after exception handling
```

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

public class CommandLineArgument14 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println(args[0]);
        System.out.println(args[1]);

    }

}
```

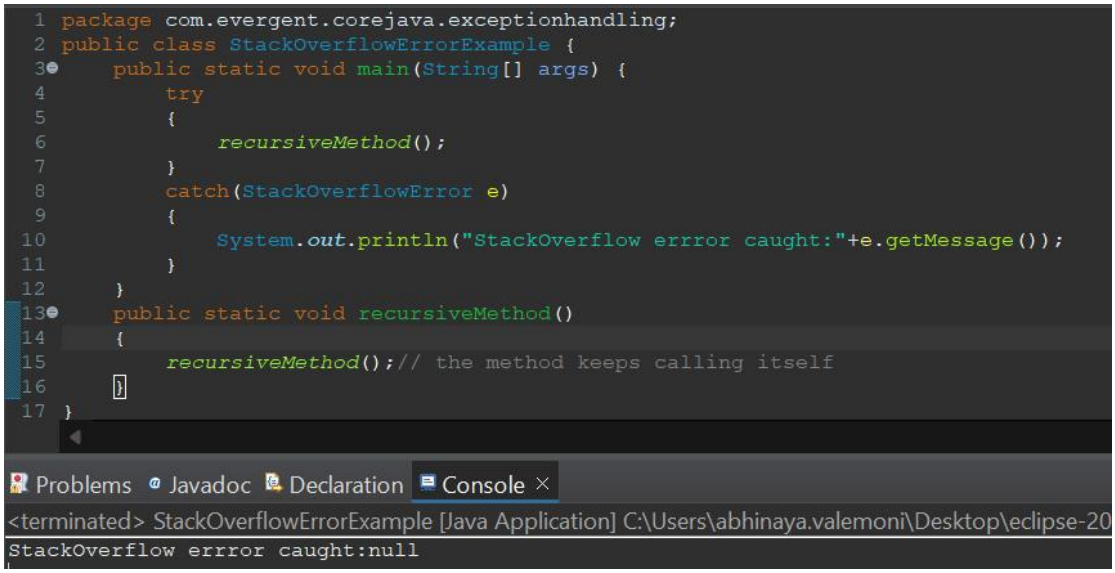
Problems Javadoc Declaration Console ×

```
<terminated> CommandLineArgument14 [Java Application] C:\Users\abhi
100
100
```

```java
package com.evergent.corejava.exceptionhandling;
import java.io.File;
public class CompileTimeDemo15 {

    public static void main(String[] args) {
        try {
            File file=new File("c:/myadata/myinfo.txt");
            Scanner scanner=new Scanner(file);
            while(scanner.hasNextLine()) {
                System.out.println(scanner.nextLine());
            }
            scanner.close();
        }
        catch(FileNotFoundException e){
            e.printStackTrace();
        }
    }
```

Problems Javadoc Declaration Console ×

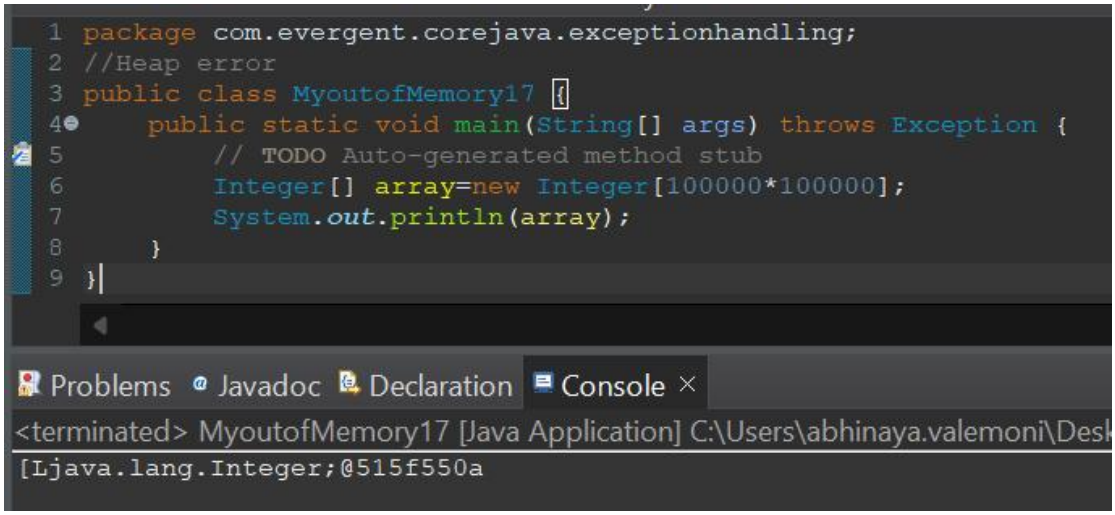
```
<terminated> CompileTimeDemo15 [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-2023-03\eclipse-2023-03\plu
java.io.FileNotFoundException: c:\myadata\myinfo.txt (The system cannot find the path specified)
        at java.base/java.io.FileInputStream.open0(Native Method)
        at java.base/java.io.FileInputStream.open(FileInputStream.java:216)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)
        at java.base/java.util.Scanner.<init>(Scanner.java:639)
        at com.evergent.corejava.exceptionhandling.CompileTimeDemo15.main(CompileTimeDemo15.java:10)
```



```
1  package com.evergent.corejava.exceptionhandling;
2  public class StackOverflowErrorExample {
3      public static void main(String[] args) {
4          try
5          {
6              recursiveMethod();
7          }
8          catch(StackOverflowError e)
9          {
10             System.out.println("StackOverflow errror caught:"+e.getMessage());
11         }
12     }
13     public static void recursiveMethod()
14     {
15         recursiveMethod();// the method keeps calling itself
16     }
17 }
```

Problems Javadoc Declaration Console ×

```
<terminated> StackOverflowErrorExample [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclipse-20
StackOverflow errror caught:null
```



```
1  package com.evergent.corejava.exceptionhandling;
2  //Heap error
3  public class MyoutofMemory17 {
4      public static void main(String[] args) throws Exception {
5          // TODO Auto-generated method stub
6          Integer[] array=new Integer[100000*100000];
7          System.out.println(array);
8      }
9  }
```

Problems Javadoc Declaration Console ×

```
<terminated> MyoutofMemory17 [Java Application] C:\Users\abhinaya.valemoni\Desk
[Ljava.lang.Integer;@515f550a
```

JAVA Beans:

1. Java bean is a mechanism
2. Java bean is light weight
3. All attributes are private
4. get/set methods are public
5. Implements java.io.Serializable interface
6. We can achieve tightly encapsulation through java beans

```java
package com.evergent.corejava.javabeans;
import java.io.Serializable;
public class Employee implements Serializable {
    private int eno;
    private String ename;
    private double sal;
    public void setEno(int eno) {
        this.eno=eno;
    }
    public void setEname(String ename) {
        this.ename=ename;
    }
    public void setSal(double sal) {
        this.sal=sal;
    }
    public int getEno() {
        return eno;
    }
    public String getEname() {
        return ename;
    }
    public double getSal() {
        return sal;
    }
}

```

```java
package com.evergent.corejava.javabeans;
//initializing and retriving using setter and getter respectively
public class EmployeeImpl {
    public static void main(String[] args) {
        Employee emp=new Employee();
        emp.setEno(100);
        emp.setEname("Abhi");
        emp.setSal(500.00);
        System.out.println("Employee no "+emp.getEno());
        System.out.println("Employee name "+emp.getEname());
        System.out.println("Employee sal "+emp.getSal());
    }
}
```

Problems Javadoc Declaration Console X

<terminated> EmployeeImpl [Java Application] C:\Users\abhinaya.valemoni\Desktop\eclips

```
Employee no 100
Employee name Abhi
Employee sal 500.0
```

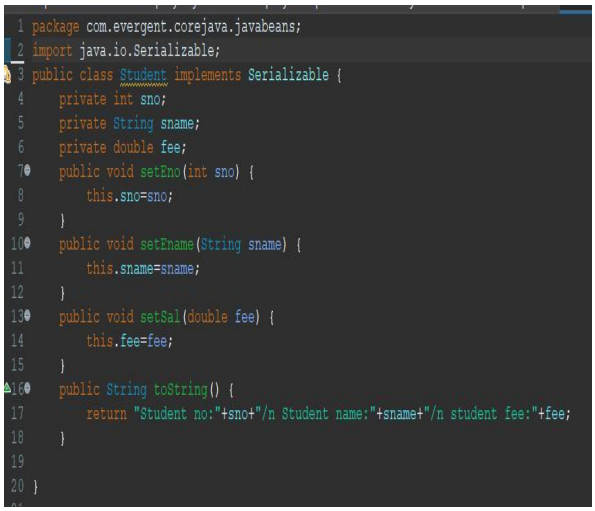
```java
package com.evergent.corejava.javabeans;
import java.io.Serializable;
public class Product implements Serializable {
    private int pno;
    private String pname;
    private double price;
    public Product(int pno, String pname, double price) {
        this.pno=pno;
        this.pname=pname;
        this.price=price;
    }
    public int getPno() {
        return pno;
    }
    public String getPname() {
        return pname;
    }
    public double getPrice() {
        return price;
    }
}
```

```java
package com.evergent.corejava.javabeans;
//initializing using constructor and retrving with getter
public class ProductImpl {
    public static void main(String[] args) {
        Product pd = new Product(100,"Abhi",500.10);
        System.out.println("Product no "+pd.getPno());
        System.out.println("Product name "+pd.getPname());
        System.out.println("price "+pd.getPrice());
    }
}
```

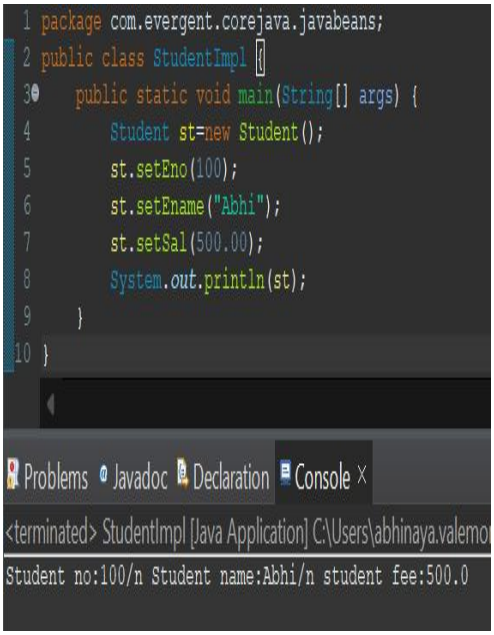
Problems Javadoc Declaration Console X

<terminated> ProductImpl (1) [Java Application] C:\Users\abhinaya.valemoni\De

```
Product no 100
Product name Abhi
price 500.1
```



```java
package com.evergent.corejava.javabeans;
import java.io.Serializable;
public class Student implements Serializable {
    private int sno;
    private String sname;
    private double fee;
    public void setEno(int sno) {
        this.sno=sno;
    }
    public void setEname(String sname) {
        this.sname=sname;
    }
    public void setSal(double fee) {
        this.fee=fee;
    }
    public String toString() {
        return "Student no:"+sno+"/n Student name:"+sname+"/n student fee:"+fee;
    }

}
```



```java
package com.evergent.corejava.javabeans;
public class StudentImpl {
    public static void main(String[] args) {
        Student st=new Student();
        st.setEno(100);
        st.setEname("Abhi");
        st.setSal(500.00);
        System.out.println(st);
    }
}
```

Problems Javadoc Declaration Console

<terminated> StudentImpl [Java Application] C:\Users\abhinaya.valemo

```
Student no:100/n Student name:Abhi/n student fee:500.0
```

Git and GitHub:

Git Commands:

1. Git init
2. Git status
3. Git add .
4. Git commit -m “Uploaded”
5. Git branch
6. Git remote add origin “ link”

7. Git push --force origin master