

Java

[app2022 and javacode folder]

Taking input from user by object of scanner class.

Scanner :- Scanner is a built-in class which is available in **java.util package**.

Package : package is a container of classes, interfaces and sub-packages.

```
// Write a java code to print a message.
// Class is a blueprint of object.
// Class is a container of variables and methods.

class p1{
public static void main(String[] args)
{
System.out.println("Hello world");
}
}
```

Sol :-

```
Microsoft Windows [Version 10.0.22000.1219]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>D:

D:\>md app2022

D:\>cd app2022

D:\app2022>notepad p1.java

D:\app2022>javac p1.java

D:\app2022>java p1
Hello world
```

6-feb-2023

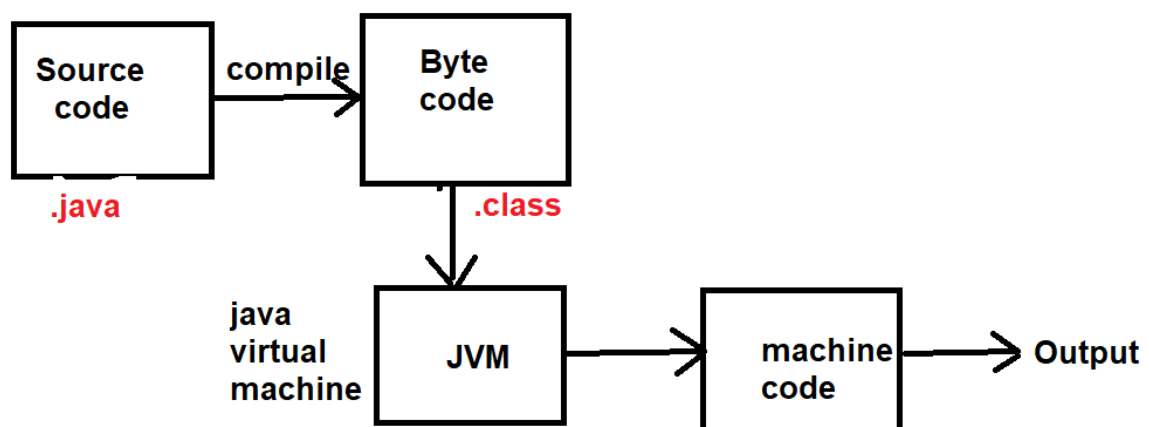
Revision JAVA

Java Programming

Author	:	James Gosling
Vender	:	SUN Microsoft
Old Name	:	OAK
Present Name	:	Java
Symbol	:	Coffee Cup with Saucer
Slogan	:	WORA (Write once run anywhere)

Language Type	:	Open Source
Operating System	:	Any Operating System
Developed From	:	C and C++
Extensions	:	.java , .class , .jar

Execution Scenario Of JAVA



Features of Java :

1. **Simple** :- Java is Simple for C and C++ learner , because all complexities of C and C++ reduced in java .
2. **Open Source** :- Java is open source that means source code of java is available for user and user can modify source code .
3. **Platform Independent** :- Java is Platform Independent , that means you can run java program on any Operating System .
4. **Object Oriented** :- Java Programming Language is an Object Oriented programming language .
5. **High Performance** :- The performance of java programming is better than C and C++ .
6. **Write Once Run Anywhere** .
7. **Architecture Neutral** :- Java program is not executed under operating system. Java program is executed under java runtime environment (JRE).
8. **Multithreaded** :- Java programming Language supports concept of multithreading .
9. **Web Application Development** : You can develop web application by using java programming language .

Applications Developed Using Java :-

1. **Desktop Application** :
e.g. Media Player , Anti-Virus etc .
2. **Web Application** :
e.g. irctc.co.in.
3. **ERP (Enterprise Resource Planning) like banking solution**
e.g. Finacle .
4. **Mobile Application Development** .
5. **Embedded System and Robotics** .
6. **Game Development**

JDK (Java Development Kit)

Commands :

Javac :- For checking JDK is available in our system .

```
Microsoft Windows [Version 10.0.22621.1105]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>javac
Usage: javac <options> <source files>
where possible options include:
    @<filename>                Read options and filename
    -Akey[=value]              Options to pass to annotation
    --add-modules <module>(<module>)*
                               Root modules to resolve in addition to the initial
modules
                               on the module path if <module> is ALL-MODULE-PATH
    --boot-class-path <path>, -bootclasspath <path>
                               Override location of bootstrap class files
    --class-path <path>, -classpath <path>, -cp <path>
                               Specify where to find user class files and annotations
    -d <directory>             Specify where to place class files
    -deprecation                Show deprecation warnings
```

Cd :- It is used for that Directory .

```
C:\Users\Lenovo>cd\
```

E : For change the Directory .

Md : Create new folder .

Cd : Usi folder me rahane ke liye.

```
C:\javacode>E:
E:\>md javacode
E:\>cd javacode
```

Code file :-

```
E:\javacode> notepad p1.java
```

Program

Compile the program : filename se compile hoti hai .

javac filename

javac p1.java

```
E:\javacode>javac p1.java
```

For running the program : program me jo class bnate tym classname dete hai
usse run hota hai .

java className

Java p1

```
E:\javacode>java p1  
Hello World
```

dir → It is used for know the, how much files in the folder .

Class : Class is a blueprint of object.

Example :

```
class p1
{
    public static void main(String [] args)
    {
        System.out.println("Hello World");
    }
}
```

→ public is a keyword , that is work as Access modifier .

Main method can make public .

→ Static is a keyword , that is a Modifier .

Static method ko call karne ke liye Object ki jarurt nahi parti .

→ Main is a method , our program start from a main method .

→ void is a keyword , that is work as Return type .

(agr kisi method me , void use kar lete hai to vo koi value return nahi karta hai.) void use kiya gaya hai to vo

→ String[] args ---- It is use to take input from command line .

```
System.out.println("Hello World");
```

→ System is a class

→ out is a Object

→ println is a Method .

7-feb-2023

How to take input from USER in java ?

For input in java we use **Scanner class** object. Scanner is a class available in **java.util** package.

Package : Package is a collection of classes , interfaces and sub-packages.

Code :-

```
import java.util.Scanner;
```

|

Object

```
// ClassName objectName=new ClassName();
```

```
Scanner scanner=new Scanner(System.in);
```

|

Input

For int input :-

Scanner class me nextInt() naam ka method hai . Jiska use input lene ke liye krte hai .

```
Int a;
```

```
a=scanner.nextInt();
```

For float input :-

```
Float b;
```

```
b=s.nextFloat();
```

For **Double** input :-

double c;

c=s.**nextDouble()**;

For **String** input :-

String name;

name=s.**nextLine()**;

Build-in class ka first letter capital hota hai.

Decision Controls in java :- Decision Controls are used for decision making .

In java programming language there are following types of decision controls :-

1. If Statement
2. If-else Statement
3. Ladder if-else Statement
4. Switch Statement

If-Statement :-

If is a keyword , which works as decision control .

We attach a condition with if statement, if given statement is true , then if block will executed otherwise no code will executed .

If(Condition)

{

// code

}

If-else Statement :-

If-else is the variation of if statement, We attach a condition with if statement, if given condition is true then if block code will executed and if given condition is false then else block code will executed.

if(Condition)

{

//code

}

else

{

//code

}

Ternary Operator (?) :-

Ternary Operator is a alternate of if-else .

(expression1) ? (expression2) : (expression3)

➔ If expression1 is true then expression2 will executed and if expression1 is false then expression3 will executed .

Question :

1. WAP to a find area and perimeter of rectangle .
2. WAP to check given year is leap year or not .
3. WAP to find roots of Quadratic equation .

08/feb/2023

Ladder if-else :-

If you have many conditions and you want to execute code based on those conditions, then you can use ladder if-else .

If(condition1)

{

// code1

}

else if(condition)

{

//code2

}

else

{

//code3

}

Switch Statement:

Switch is a keyword, which work as case control. It is used to create a menu based program.

// In switch we cannot passed double or float value.

Switch, case, break, default they are a keyword.

Switch(expression) //int or char or String

{

case 1:

```
//code1;
```

```
break;
```

```
case2:
```

```
//code2;
```

```
break;
```

```
default:
```

```
//code
```

```
}
```

Loop Control:

If you have a block of code which you want to execute repeatedly up to given condition is true, then you can use a loop control.

In java programming language there are Four types –

1. While loop
2. For loop
3. Do-while loop
4. For each loop

While loop:

While is a keyword which works as loop control.

Initialization of loop counter,

While(condition)

```
{
```

```
//code
```

Updation of loop counter

```
}
```

Question:

1. Input coordinates of point and check quadrants.
2. Temperature Convertor.

For loop:

For is a keyword, which works as loop control.

Working of for loop is same as while but syntax is different.

For (Initialization; Condition; Updation)

```
{  
// code  
}
```

Do-while loop:

Initialization of loop counter;

do

```
{
```

```
// Code;
```

Updation of code;

```
} while(Condition);
```

10/feb/2023

Question:

1. WAP to find compound interest.
2. WAP to find area of perimeter of circle.
3. WAP to check given number is Armstrong or not.

For each loop:

For each loop in java is used to traverse elements of a collection like array.

ARRAY in java:

Array is a collection of similar data types, that means an array can store multiple values of similar data types.

How to create Array in java?

datatype [] arrayname=new datatype[size];

For example:

```
int [] list=new int[10];
```

How to initialization of Array:

```
int [] x = {10,20,30,40,50} ;
```

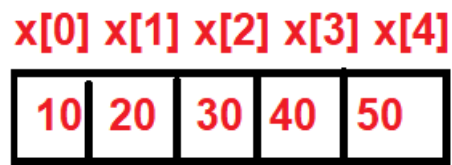
x[0] = 10

x[1] = 20

x[2] = 30

x[3] = 40

x[4] = 50



How to take input from user for an array?

Code Segment:-

```
int [] x=new int[5];
int i;
Scanner s=new Scanner(System.in);
System.out.println("Enter five number");
for(i=0; i<5; i++)
{
    x[i]=s.nextInt();
}
```

STRING in java:

String is a class in java. The object of String class is used to store **sequence of characters**.

```
String str="Softpro India";
```

toUpperCase() :- toUpperCase() method of String class is used to convert String into Upper Case (Capital letters).

toLowerCase() :- toLowerCase() method of String class is used to convert String into Lower Case (Small letters).

length() :- length() method of String Class is used to find length of Sting.

equals() :- equals() method of String Class is used to compare two Strings for equality. This method return Boolean value.

equalsIgnoreCase() :- This method of String class compare two

split() :- split() method of String class split String into substrings.
This method return array of String .

Eg :

```
String str= "He is a good boy."
```

```
String [] words = str.split(" ");
```

```
words[0] = "He";
```

```
words[1] = "is";
```

```
words[2] = "a";
```

```
words[3] = "good";
```

```
words[4] = "boy";
```

replace() :-- replace() method of String class replace one string with another string in given string.

```
replace(findWhat,replaceWith);
```

Method in java :-

Method is a named block of code, which perform specific task.

If you have a block of code which required different locations of program, then you can create a method of that code and call it from desired locations.

By using method you can avoid to write same code over and over.
Method is used to achieve modularity.

How to create a method in java?

```
<modifier> <return_type> method_name(parameters)
{
//code
}
```

Ex:-

```
Public int add(int x, int y)
{
return(x+y);
}
```

Types of Methods in java:-

In java programming language there are two types of methods –

1. Static methods
2. Non-static methods

Static method –

Static methods are created by using static, modifier.

These methods are also called **class methods**.

There is no need of object to call static methods.

// Static method ko with object or without object call kiya ja skta hai.

Non-Static Method –

Non-static methods are created without using static modifier.
There is a need Object to call non-static methods.

Note:- Method are created within class and outside of main() method.

OOPS

OOPS stands for **Object Oriented programming System**. It is a mechanism of software development.

OOPS has four pillars –

- 1. Abstraction**
- 2. Encapsulation**
- 3. Inheritance**
- 4. Polymorphism**

Abstraction –

Abstraction is a mechanism to **hide functionalities** of an object.

Encapsulation –

Encapsulation is a mechanism to wrap properties and functionalities in a single unit.

That single unit is called object.

Inheritance –

Inheritance is a mechanism to **create new product** by using **existing product**.

Polymorphism –

Term Polymorphism means **one thing many forms**.

Note :- Any programming language which follows these four concepts is known as Object Oriented Language.

16/Feb/2023

CLASS:-

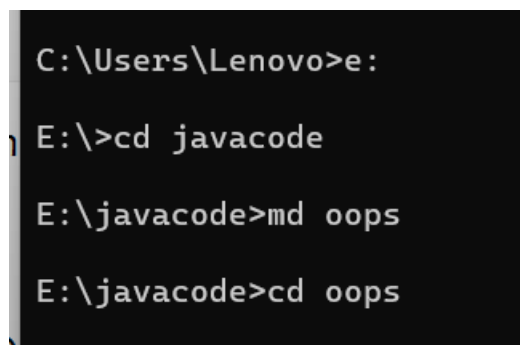
Class is a blueprint of object. Class is a container of variables and methods.

Class is created by using “**Class**” keyword followed by classname.

The body of class is enclosed within braces.

Class Employee

```
{  
// Variables and Methods  
}
```



```
C:\Users\Lenovo>e:  
E:\>cd javacode  
E:\javacode>md oops  
E:\javacode>cd oops
```

Constructor:-

Constructor is a special method, which is used to initialize variables.

The constructor has following properties –

1. Constructor name is same as Class Name.
2. Constructor has no return type.
3. Constructor Call automatically as soon as object is created.

Inheritance In Java:-

In Inheritance you can create a new class by using existing class. Existing class is called base class and new created class is called derived class.



➤ **The concept of Inheritance is also called “Reusability” .**

```
Class A {  
//Variable and methods  
}  
Class B extends A {  
//Variable and methods  
}
```

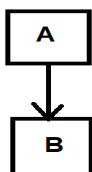
Types of Inheritance in java:-

In java programming language there are three types of inheritance are available:-

1. Single Inheritance
2. Hierarchical Inheritance
3. Multi-level Inheritance

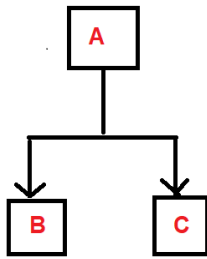
Single Inheritance:-

In Single Inheritance there is a single class and single derived class.

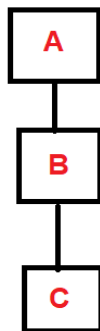


Heirarchical Inheritance:-

In Hierarchical Inheritance there is a single base class and multiple derived classes.



Multi-level Inheritance:-



Class A

```
{  
// code  
}
```

Class B extends A

```
{  
// code 2  
}
```

Class C extends B

```
{  
//code 3  
}
```

18/feb/2023

Polymorphism:-

The term 'Polymorphism' means one thing many forms.

In java there are two types of Polymorphism—

1. *Compile time polymorphism (Method Overloading)*
2. *Run time polymorphism (Method Overriding)*

➤ **Method Overloading—**

In java you can create multiple methods with same name in same class, but their parameters should be different.

Based on Method parameter it is decided at compilation time that which method call from where, This is called Method Overloading.

Method parameters can be different in two types—

1. Number of parameter can be different
2. Types of parameter can be different

➤ **Method Overriding:-**

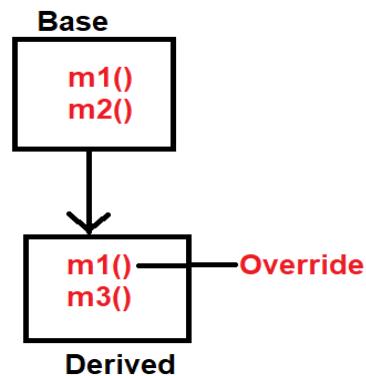
Re-writing of based class method into derived class is called method Overriding.

For Ex:-

```
class Connection
{
    void connect()
    {
        // Connect java code with oracle database
    }
}
class NewConnection extends Connection
{
    void connect()
    {
        // Connect java code with oracle database
    }
}
```

There are following rules to perform method overriding—

1. **Class must be inherited.**
2. **Base class method name and derived class method name must be same.**
3. **Base class method parameters and derived class method parameters must be same.**
4. **Base class method return type and derived class method return type can be same.**



Exception Handling In Java:-

Exception —

The dictionary meaning of Exception is abnormal terminated ab normally and rest of code is not executed.

In java programming language there are Three types of Exceptions—

1. ***Checked Exceptions***
2. ***Unchecked Exceptions***
3. ***Errors***

➤ **Checked Exceptions —**

Checked Exceptions are those exceptions which are identified by compiler at **Compilation time**.

E.g. —

**ClassNotFoundException, IOException, SQLException,
FileNotFoundException, InterruptedException...etc.**

➤ **Unchecked Exceptions —**

Unchecked Exceptions are those exceptions which are

identified at **Run time**.

E.g. –

**NullPointerException, ArithmeticExceptions,
InputMismatchedExceptions, ArrayIndexOutOfBoundsException...etc.**

➤ **Error –**

Error are occurred due to **lack of system resources**.

E.g. –

AwtError, FileNotFoundException, IOError,...etc.

21/Feb/2023

Concept of Interface, Abstract Class & Class –

Interface –

Interface in java is a container of abstract methods.

It is used to achieve full abstraction.

Abstract method ve methods hote hai jinme sirf methods ka declearation hota hai, use nhi hota hai.

E.g. –

Interface MyInterface

```
{  
void m1(); // public abstract void m1();  
void m2(); // public abstract void m2();  
void m3(); // public abstract void m3();  
}
```

```
C:\Users\Lenovo>e:
E:\>cd javacode
E:\javacode>cd oops
E:\javacode\oops>notepad interfaceDemo1.java
E:\javacode\oops>javac interfaceDemo1.java
E:\javacode\oops>javap MyInterface.class
Compiled from "interfaceDemo1.java"
interface MyInterface {
    public abstract void m1();
    public abstract void m2();
    public abstract void m3();
}
E:\javacode\oops>
```

```
E:\javacode\oops>javap MyInterface.class
```

Object –

Class ka object bnega alwage

Object sirf usi ka bnega jaha sari files implemented hogi. [like that→ class]

Interface –

If you have requirements but you don't know about its implementations, then you can use interface contain abstracts method only.

- ➔ **An interface can extends (Inherit) another Interface.**
- ➔ **You can't Create **Object of Interface**.**

Abstract Class –

Abstract class is a class which contain abstract methods and implemented method both.

- If you have requirements, you know about implementations but not complete implementations, then you can use abstract class.
- ➔ An abstract class can implement interface.
- ➔ An abstract class can extends another abstract class.
- ➔ You can't create object of abstract class.

Class –

Class is a container of implemented methods.

If you have requirements and you know about complete implementation then you can use class.

- ➔ A class can implements an Interface.
- ➔ A class can extends abstract class.
- ➔ A class can extends another class.
- ➔ You can create object of class.

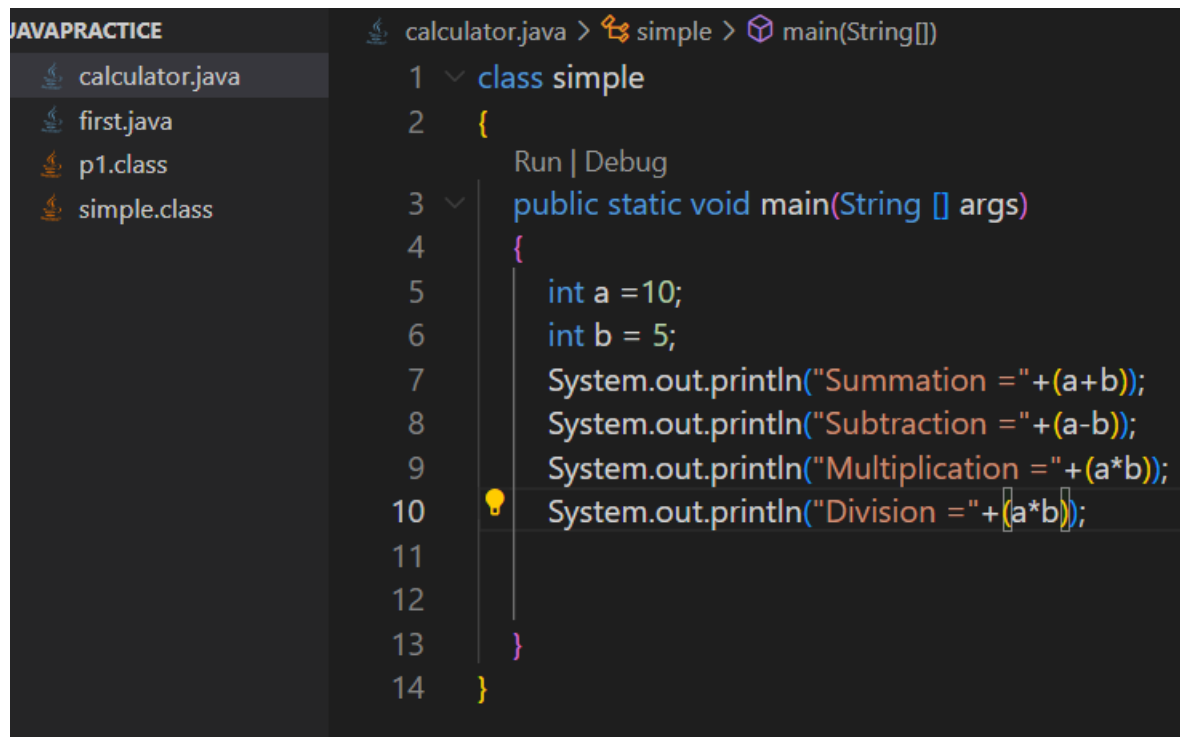
Examples – (javacode ➔ oops ➔ interfaceDemo2.java)

JAVAPRACTICE –

Programs –

1. Simple Calculator –

Without take input from user.



The screenshot shows an IDE with a file explorer on the left and a code editor on the right. The file explorer lists 'calculator.java', 'first.java', 'p1.class', and 'simple.class'. The code editor shows the following Java code:

```
calculator.java > simple > main(String[])
1  class simple
2  {
3      Run | Debug
4      public static void main(String [] args)
5      {
6          int a =10;
7          int b = 5;
8          System.out.println("Summation =" +(a+b));
9          System.out.println("Subtraction =" +(a-b));
10         System.out.println("Multiplication =" +(a*b));
11         System.out.println("Division =" +(a/b));
12     }
13 }
14 }
```

Sol –

```
E:\javacode\JAVAPRACTICE>java calculator.java
Summation =15
Subtraction =5
Multiplication =50
Division =50

E:\javacode\JAVAPRACTICE>javac calculator.java

E:\javacode\JAVAPRACTICE>java calculator.java
Summation =15
Subtraction =5
Multiplication =50
Division =50
```

INPUT FROM USER –

2. WAP a program to make a Simple Calculator.

SOL :-

```
Calculator2.java > Calculator2 > main(String[])
1  import java.util.*;
2
3  class Calculator2 {
   Run | Debug
4  public static void main(String [] args)
5  {
6      int a,b;
7      Scanner sc=new Scanner(System.in);
8      System.out.println(x: "Enter a two number :");
9      a=sc.nextInt();
10     b=sc.nextInt();
11     System.out.println("Sum = "+(a+b));
12     System.out.println("Subtraction = "+(a-b));
13     System.out.println("Multiplication = "+(a*b));
14     System.out.println("Division = "+(a/b));
15 }
16 }
17
```

After RUN –

```
E:\javacode\JAVAPRACTICE>javac calculator2.java

E:\javacode\JAVAPRACTICE>java calculator2.java
Enter a two number :
4
5
Sum = 9
Subtraction = -1
Multiplication = 20
Division = 0

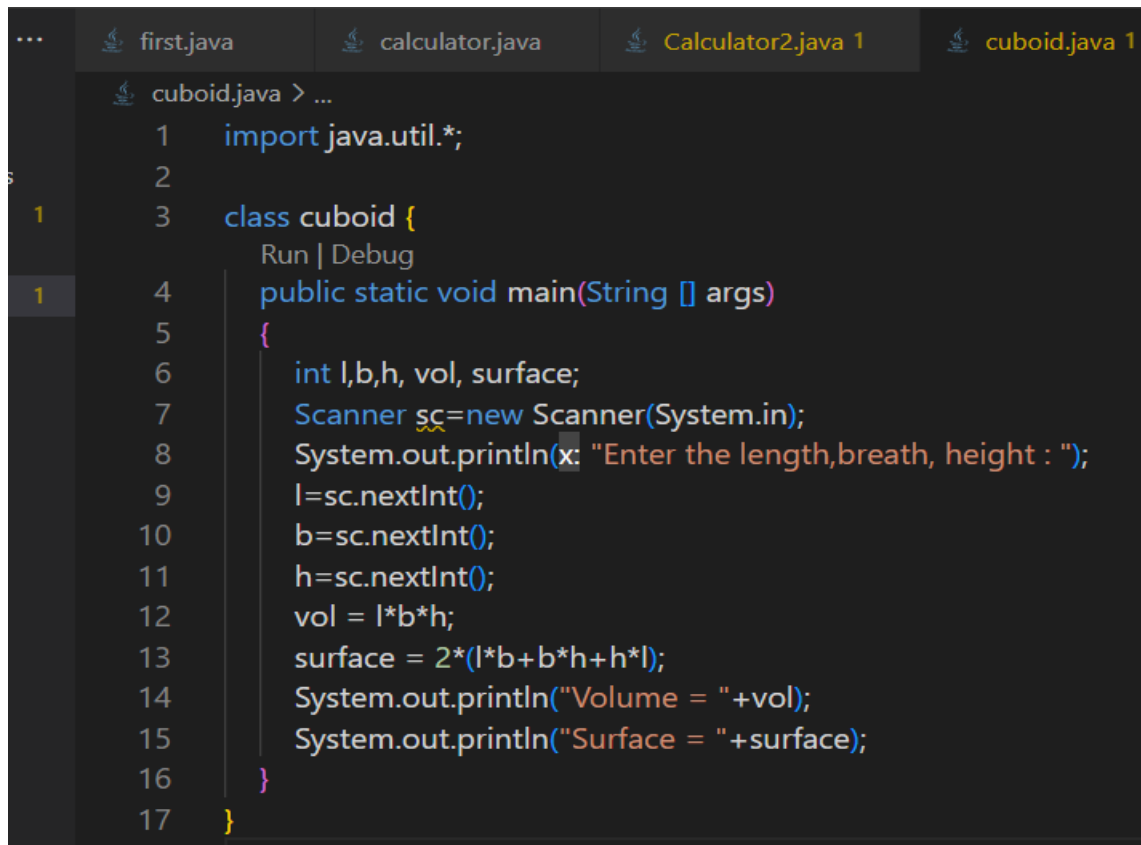
E:\javacode\JAVAPRACTICE>
```

3. Write a program to find the surface area of cuboid .

Volume = $l*b*h$

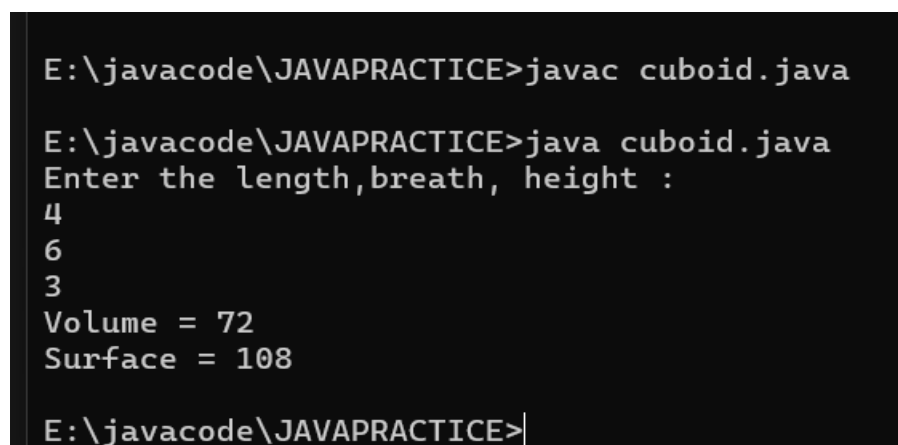
Surface = $2(l*b+b*h+h*l)$

Sol –

A screenshot of an IDE with a dark theme. The top bar shows four tabs: 'first.java', 'calculator.java', 'Calculator2.java 1', and 'cuboid.java 1'. The 'cuboid.java' tab is active, showing the following Java code:

```
1  import java.util.*;
2
3  class cuboid {
4      public static void main(String [] args)
5      {
6          int l,b,h, vol, surface;
7          Scanner sc=new Scanner(System.in);
8          System.out.println(x: "Enter the length,breath, height : ");
9          l=sc.nextInt();
10         b=sc.nextInt();
11         h=sc.nextInt();
12         vol = l*b*h;
13         surface = 2*(l*b+b*h+h*l);
14         System.out.println("Volume = "+vol);
15         System.out.println("Surface = "+surface);
16     }
17 }
```

Running –

A screenshot of a Windows command prompt with a black background and white text. The following commands and output are shown:

```
E:\javacode\JAVAPRACTICE>javac cuboid.java

E:\javacode\JAVAPRACTICE>java cuboid.java
Enter the length,breath, height :
4
6
3
Volume = 72
Surface = 108

E:\javacode\JAVAPRACTICE>|
```

4. WAP to take a number as input, if number is 1 then print "Hi" .

Sol –

```
if.java > ...
1  import java.util.*;
2
3  class condition
4  {
5
6      Run | Debug
7      public static void main(String [] args)
8      {
9          int n;
10         Scanner s=new Scanner(System.in);
11         System.out.print(s: "Enter a number : ");
12         n=s.nextInt();
13
14         if(n==1)
15         {
16             System.out.print(s: "hi");
17         }
18     }
19 }
```

Running –

```
E:\javacode\JAVAPRACTICE>javac if.java

E:\javacode\JAVAPRACTICE>java if.java
Enter a number : 5

E:\javacode\JAVAPRACTICE>java if.java
Enter a number : 1
hi
E:\javacode\JAVAPRACTICE> java condition
Enter a number : 5

E:\javacode\JAVAPRACTICE> java condition
Enter a number : 1
hi
E:\javacode\JAVAPRACTICE>|
```

5. WAP to check given number is Even or Odd .

Sol –

```
odd_even.java > Even
1  import java.util.*;
2
3  class Even
4  {
5      Run | Debug
6      public static void main(String [] args)
7      {
8          int n;
9          Scanner sc=new Scanner(System.in);
10         System.out.print(s: "Enter a number :");
11         n=sc.nextInt();
12
13         if(n%2==0)
14         {
15             System.out.print(s: "Number is Even ");
16         }
17         else
18         {
19             System.out.print(s: "Number is Odd");
20         }
21     }
```

```
E:\javacode\JAVAPRACTICE>javac odd_even.java
E:\javacode\JAVAPRACTICE>java Even
Enter a number :4
Number is Even
E:\javacode\JAVAPRACTICE>
```

FOR MAKE Folder –

```
C:\javacode>E:
E:\>md javacode
E:\>cd javacode
```

1. First Program in JAVA.

```
Command Prompt
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>e:

E:\>cd javacode

E:\javacode>cd javapractice

E:\javacode\JAVAPRACTICE>|
```

Compile –

Javac filename.java

Javac demo1.java

Running –

Java CLASSNAME

Java p1

```
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>e:

E:\>cd javacode

E:\javacode>cd javapractice

E:\javacode\JAVAPRACTICE>notepad demo1.java

E:\javacode\JAVAPRACTICE>javac demo1.java

E:\javacode\JAVAPRACTICE>java p1
Hello world

E:\javacode\JAVAPRACTICE>|
```

2. Calculator program.

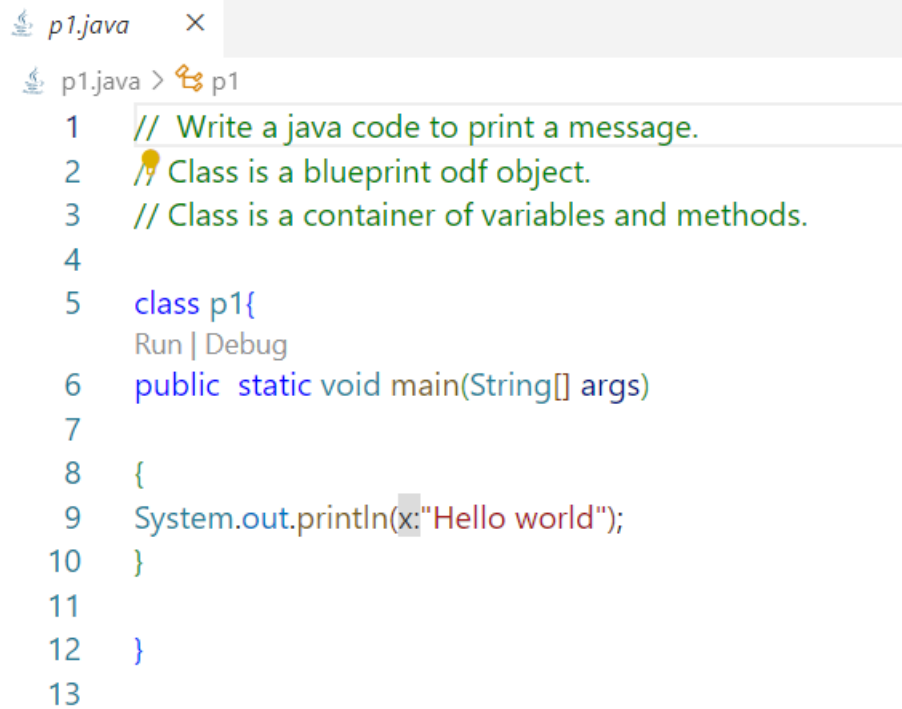
```
Calculator2.java > Calculator2 > main(String[])
1  import java.util.*;
2
3  class Calculator2 {
   Run | Debug
4  public static void main(String [] args)
5  {
6      int a,b;
7      Scanner sc=new Scanner(System.in);
8      System.out.println(x: "Enter a two number :");
9      a=sc.nextInt();
10     b=sc.nextInt();
11     System.out.println("Sum = "+(a+b));
12     System.out.println("Subtraction = "+(a-b));
13     System.out.println("Multiplication = "+(a*b));
14     System.out.println("Division = "+(a/b));
15 }
16 }
17
```

```
E:\javacode\JAVAPRACTICE>javac calculator2.java

E:\javacode\JAVAPRACTICE>java calculator2.java
Enter a two number :
4
5
Sum = 9
Subtraction = -1
Multiplication = 20
Division = 0

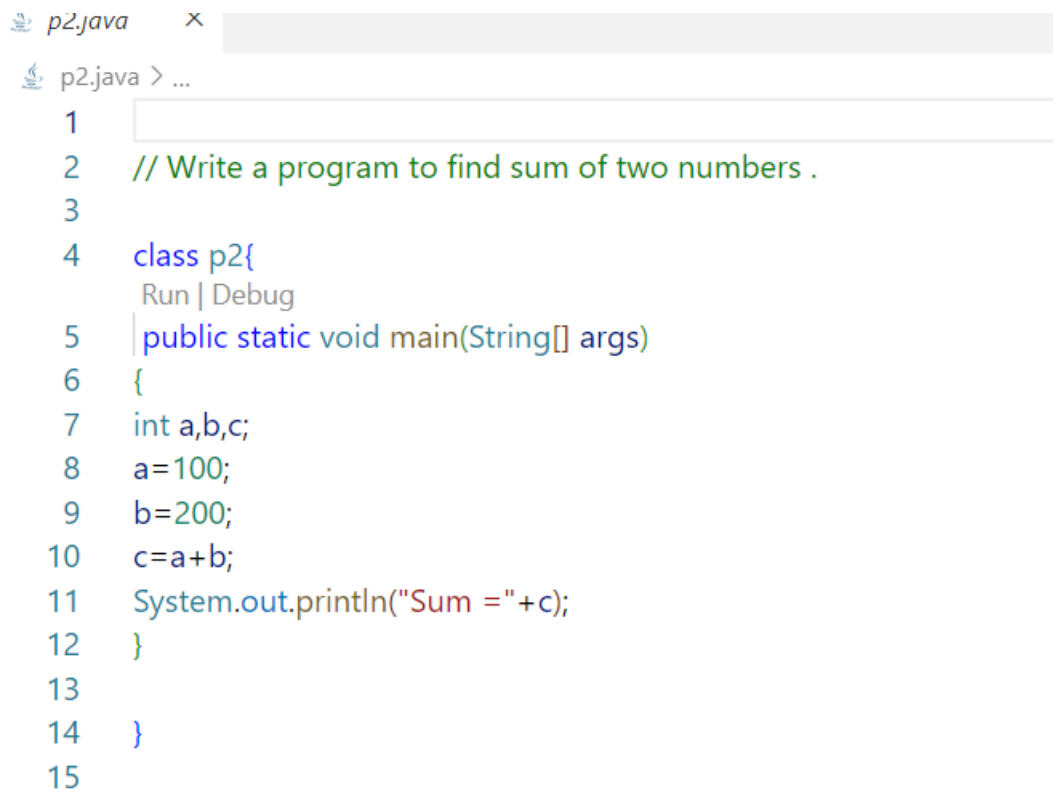
E:\javacode\JAVAPRACTICE>
```

Programs → 1.)



```
p1.java ×
p1.java > p1
1 // Write a java code to print a message.
2 // Class is a blueprint of object.
3 // Class is a container of variables and methods.
4
5 class p1{
  Run | Debug
6 public static void main(String[] args)
7
8 {
9 System.out.println("Hello world");
10 }
11
12 }
13
```

2.)



```
p2.java ×
p2.java > ...
1
2 // Write a program to find sum of two numbers .
3
4 class p2{
  Run | Debug
5 public static void main(String[] args)
6 {
7 int a,b,c;
8 a=100;
9 b=200;
10 c=a+b;
11 System.out.println("Sum =" +c);
12 }
13
14 }
15
```


3.)

```
p3.java
File Edit View

/*
    WAP to find area and perimeter of rectangle .
    Aera = l*b;
    P = 2(l+b);
*/

import java.util.Scanner;
class p3
{
public static void main(String[] args)
{
int l,b,a,p;
Scanner s=new Scanner(System.in);
l = s.nextInt();
b = s.nextInt();
System.out.println("Enter the value of L and b");
a=l*b;
p=2*(l+b);

System.out.println("Area =" +a);
}
}
```

4.)

```
File Edit View

// WAP to find the volume and surface area of the cuboid.
// v = l*b*h;
// sa = 2*(l*b+b*h+h*l);

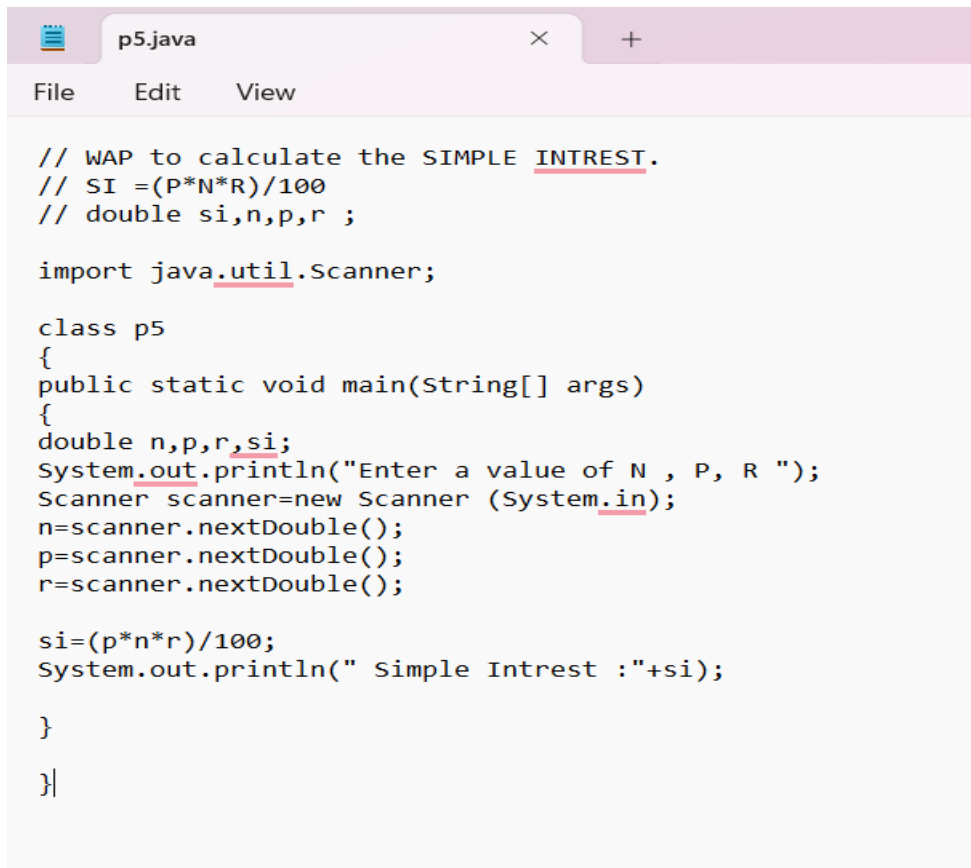
import java.util.Scanner;

class p4
{

public static void main(String [] args)
{
int l, b,h,v, sa;
System.out.println("Enter the value of l ,b and h");
Scanner sc=new Scanner(System.in);
l=sc.nextInt();
b=sc.nextInt();
h=sc.nextInt();
v=l*b*h;
sa = 2*(l*b+b*h+h*l);
System.out.println(" Area :"+v);
System.out.println("Cuboid :"+sa);

}
}
|
```

5.)



```
p5.java
File Edit View

// WAP to calculate the SIMPLE INTREST.
// SI =(P*N*R)/100
// double si,n,p,r ;

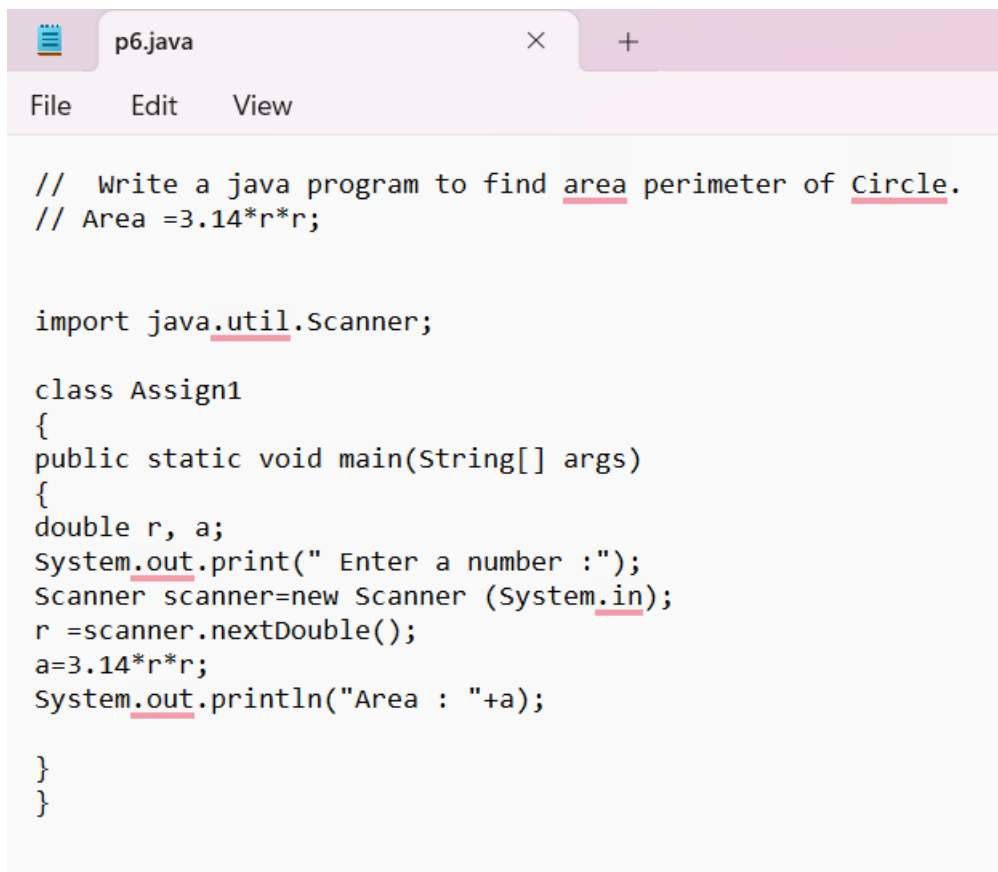
import java.util.Scanner;

class p5
{
    public static void main(String[] args)
    {
        double n,p,r,si;
        System.out.println("Enter a value of N , P, R ");
        Scanner scanner=new Scanner (System.in);
        n=scanner.nextDouble();
        p=scanner.nextDouble();
        r=scanner.nextDouble();

        si=(p*n*r)/100;
        System.out.println(" Simple Intrest :"+si);

    }
}
```

6.)



```
p6.java
File Edit View

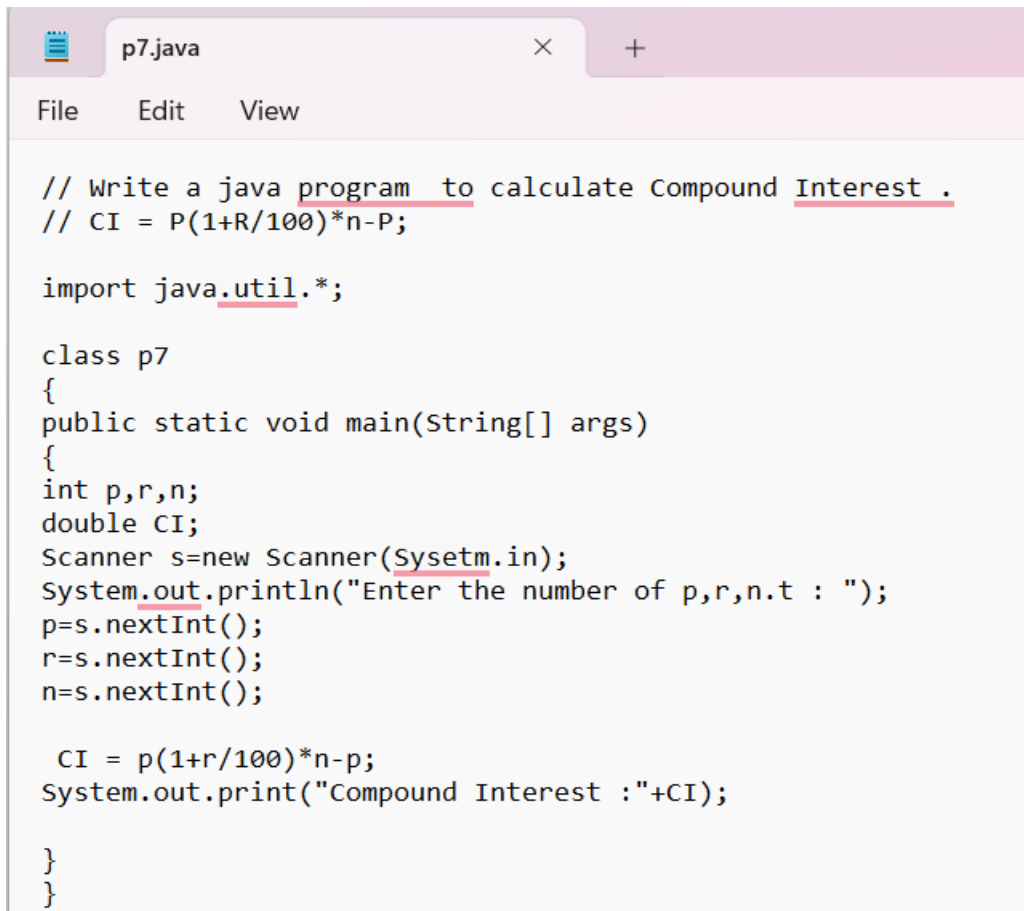
// Write a java program to find area perimeter of Circle.
// Area =3.14*r*r;

import java.util.Scanner;

class Assign1
{
    public static void main(String[] args)
    {
        double r, a;
        System.out.print(" Enter a number :");
        Scanner scanner=new Scanner (System.in);
        r =scanner.nextDouble();
        a=3.14*r*r;
        System.out.println("Area : "+a);

    }
}
```

7.)

A screenshot of a Java IDE window titled 'p7.java'. The window has a menu bar with 'File', 'Edit', and 'View'. The code inside is a Java program to calculate Compound Interest. It starts with a comment: '// Write a java program to calculate Compound Interest .'. Below this is the formula '// CI = P(1+R/100)*n-P;'. The code then imports 'java.util.*;'. It defines a class 'p7' with a 'main' method. Inside the 'main' method, it declares variables 'p, r, n' as integers and 'CI' as a double. It creates a 'Scanner' object 's' using 'Sysetm.in' (a typo for 'System.in'). It prompts the user to 'Enter the number of p,r,n,t : ' and reads three integers 'p', 'r', and 'n'. It then calculates 'CI = p(1+r/100)*n-p;' and prints 'Compound Interest :'+CI;'. The code ends with closing braces for the 'main' method and the class.

```
// Write a java program to calculate Compound Interest .
// CI = P(1+R/100)*n-P;

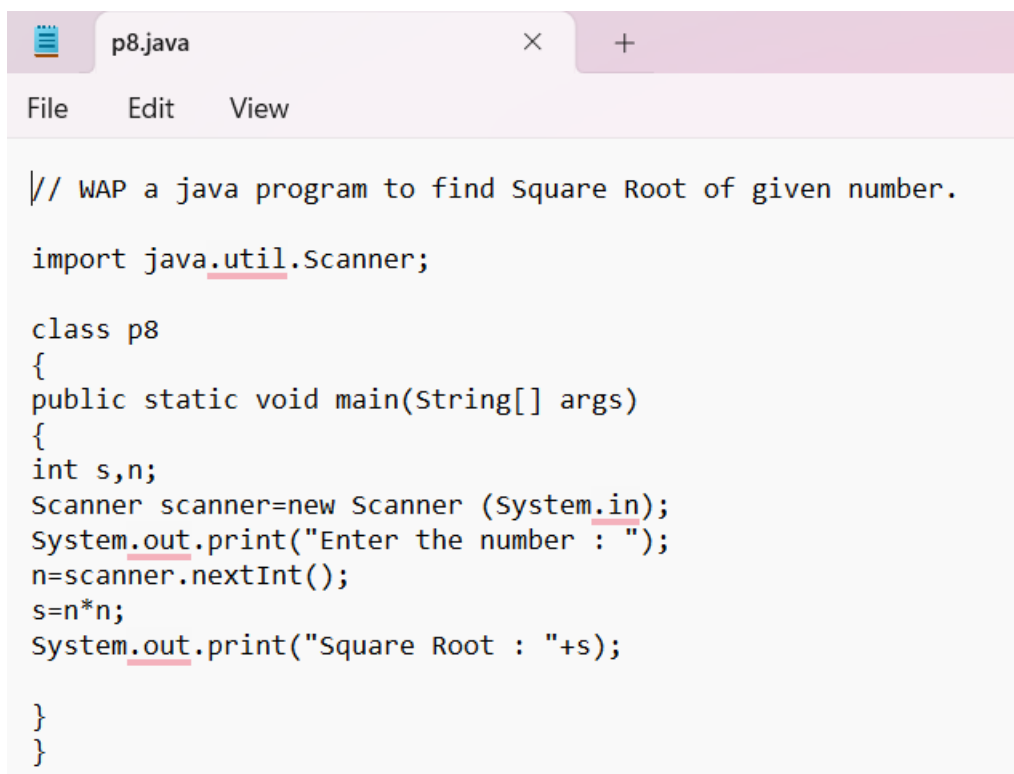
import java.util.*;

class p7
{
    public static void main(String[] args)
    {
        int p,r,n;
        double CI;
        Scanner s=new Scanner(Sysetm.in);
        System.out.println("Enter the number of p,r,n,t : ");
        p=s.nextInt();
        r=s.nextInt();
        n=s.nextInt();

        CI = p(1+r/100)*n-p;
        System.out.print("Compound Interest :"+CI);

    }
}
```

8.)

A screenshot of a Java IDE window titled 'p8.java'. The window has a menu bar with 'File', 'Edit', and 'View'. The code inside is a Java program to find the Square Root of a given number. It starts with a comment: '// WAP a java program to find Square Root of given number.'. It imports 'java.util.Scanner;'. It defines a class 'p8' with a 'main' method. Inside the 'main' method, it declares variables 's, n' as integers. It creates a 'Scanner' object 'scanner' using 'System.in'. It prompts the user to 'Enter the number : ' and reads an integer 'n'. It calculates 's=n*n;' and prints 'Square Root : '+s;. The code ends with closing braces for the 'main' method and the class.

```
// WAP a java program to find Square Root of given number.

import java.util.Scanner;

class p8
{
    public static void main(String[] args)
    {
        int s,n;
        Scanner scanner=new Scanner (System.in);
        System.out.print("Enter the number : ");
        n=scanner.nextInt();
        s=n*n;
        System.out.print("Square Root : "+s);

    }
}
```

9.)

```
// ARRAY

// How to take input from user or an array .
// code segment :-
// import java.util.*;
// Scanner sc=new Scanner(System.in);
// int[]x = new int[5];
// int i; |--> take input / for store / use array value we use LOOP CONTROLS .

// WAP to find sum and average of 10 numbers using Array.

import java.util.*;
class p10
{
    public static void main (String [] args)
    {
        int [] x=new int [10] ; // Declaration of array
        int i , sum=0;
        double avg;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter ten numbers");

        for (i=0;i<10;i++)
        {
            x[i] = sc.nextInt();
            sum=sum+x[i];
        }

        //avg=sum/10;
        //--> int/int = (Result Quesent / bragfal ) for solve this proble we use TYPEcasting.

        avg =(float) sum/10;
        System.out.println("Sum =" +sum);
        System.out.println("Avg =" +avg);

    }
}
```

10.)

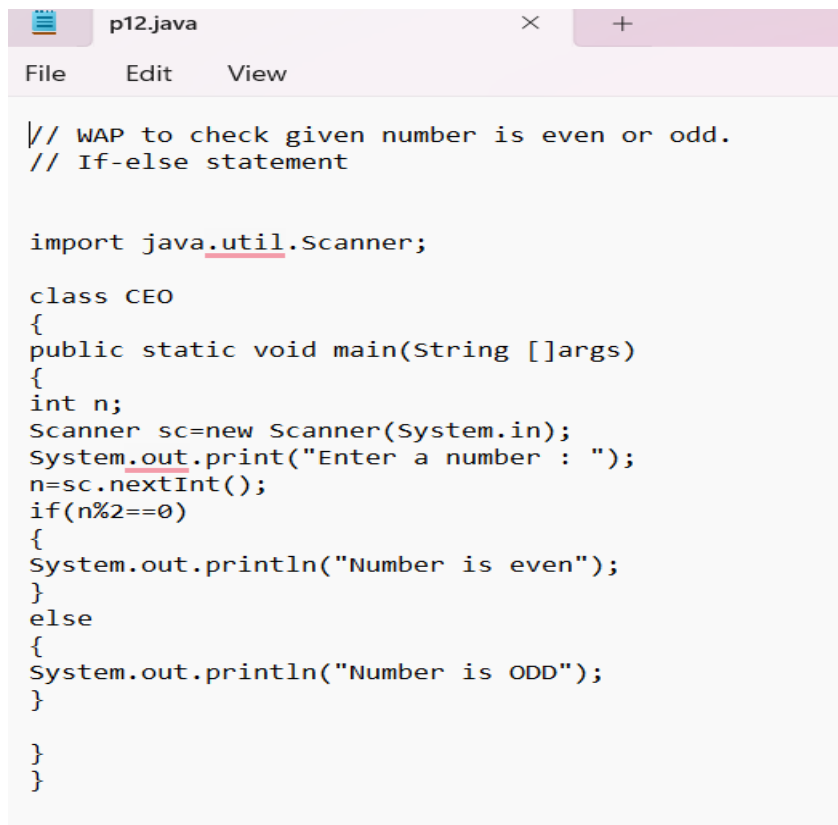
```
//WAP
// IF STATEMENT :--

import java.util.Scanner;

class soni
{
    public static void main(String [] args)
    {
        int n;
        Scanner s=new Scanner(System.in);
        System.out.print("Enter the Number :");
        n=s.nextInt();
        if (n==1)
        {
            System.out.println("hi.....");
        }
        System.out.println("by by.....");

    }
}
```

11.)



```
p12.java
File Edit View

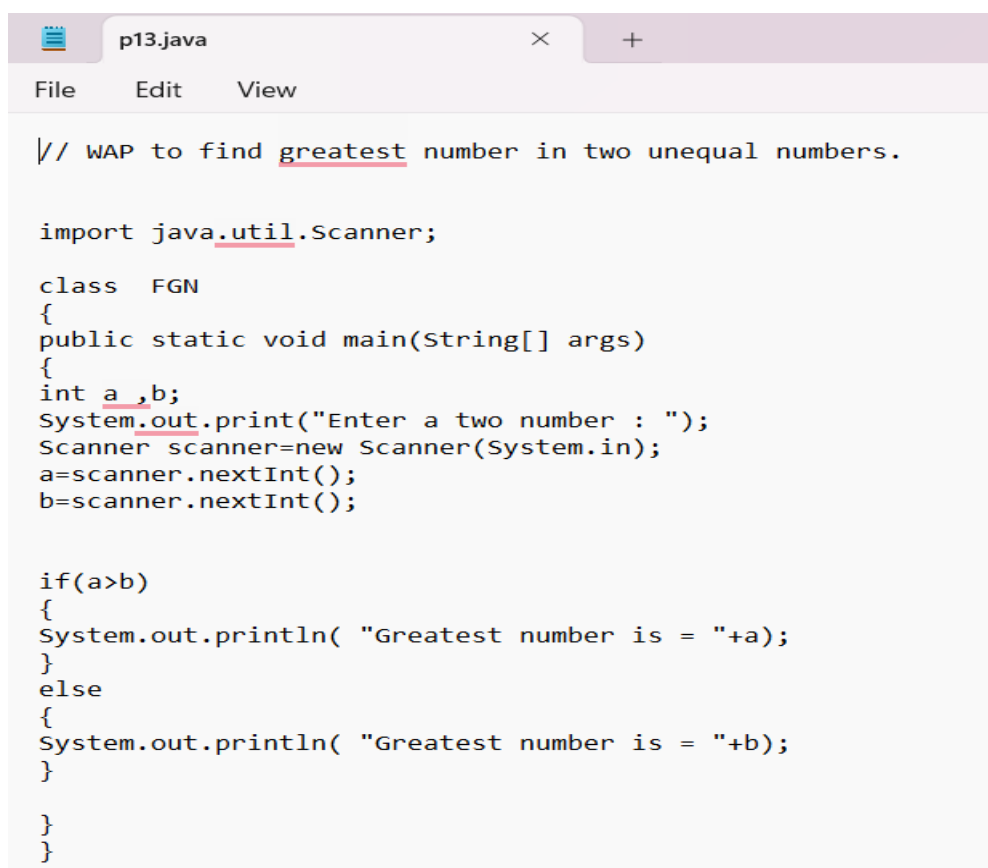
// WAP to check given number is even or odd.
// If-else statement

import java.util.Scanner;

class CEO
{
public static void main(String []args)
{
int n;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number : ");
n=sc.nextInt();
if(n%2==0)
{
System.out.println("Number is even");
}
else
{
System.out.println("Number is ODD");
}

}
}
```

12.)



```
p13.java
File Edit View

// WAP to find greatest number in two unequal numbers.

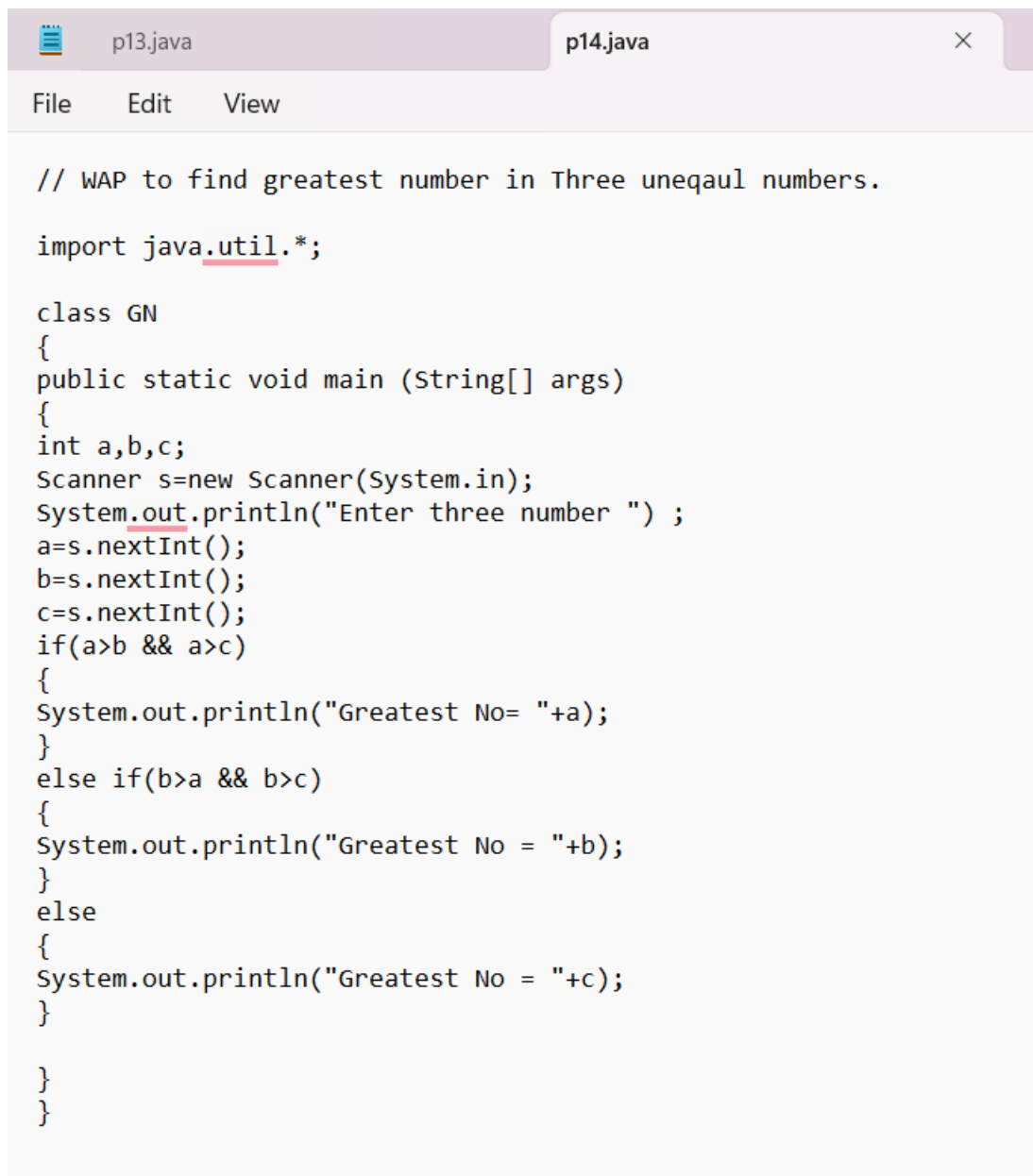
import java.util.Scanner;

class FGN
{
public static void main(String[] args)
{
int a,b;
System.out.print("Enter a two number : ");
Scanner scanner=new Scanner(System.in);
a=scanner.nextInt();
b=scanner.nextInt();

if(a>b)
{
System.out.println( "Greatest number is = "+a);
}
else
{
System.out.println( "Greatest number is = "+b);
}

}
}
```

13.)



The screenshot shows a Java IDE with two tabs: 'p13.java' and 'p14.java'. The 'p14.java' tab is active, displaying the following code:

```
// WAP to find greatest number in Three unequal numbers.

import java.util.*;

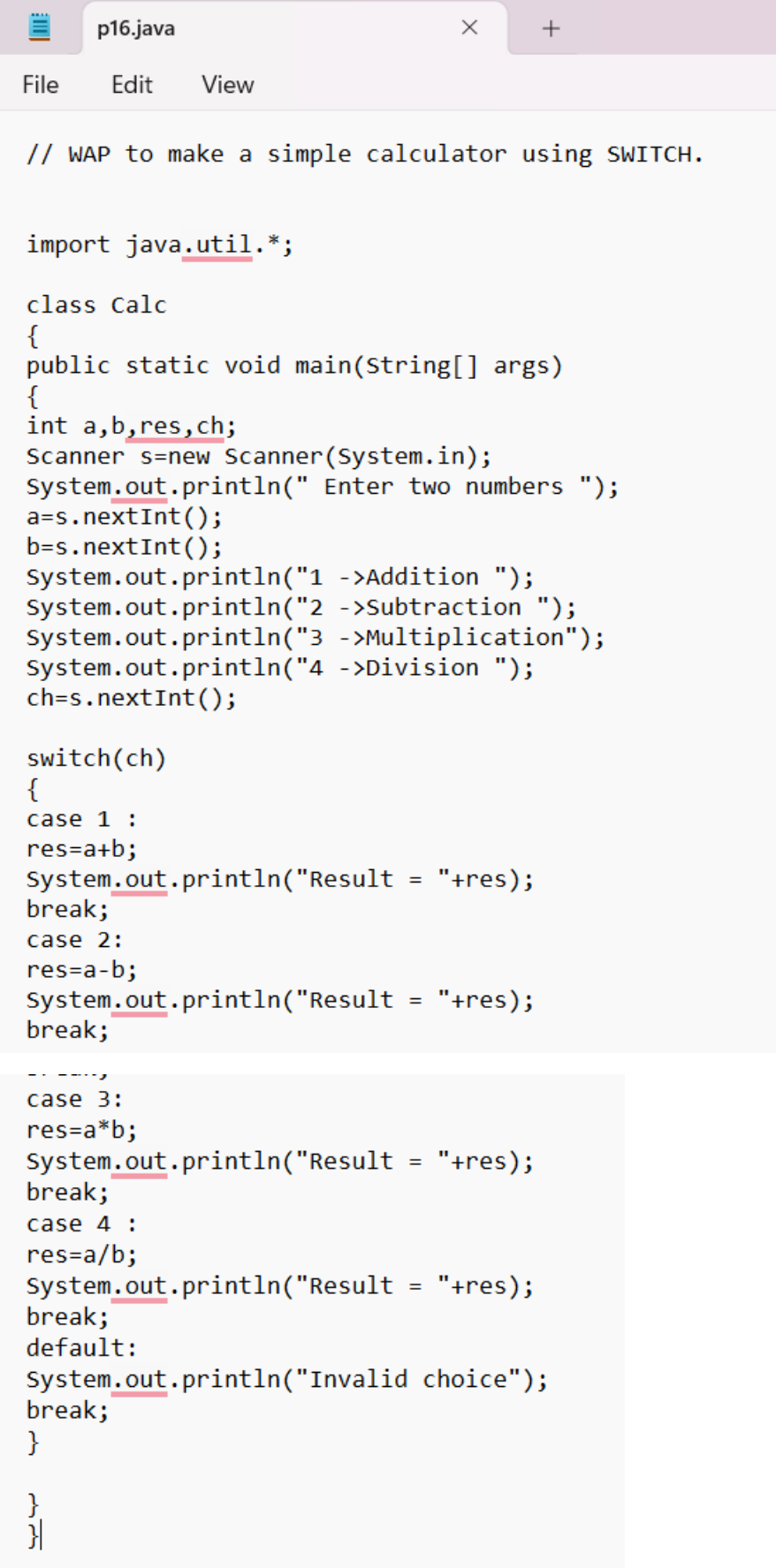
class GN
{
    public static void main (String[] args)
    {
        int a,b,c;
        Scanner s=new Scanner(System.in);
        System.out.println("Enter three number ") ;
        a=s.nextInt();
        b=s.nextInt();
        c=s.nextInt();
        if(a>b && a>c)
        {
            System.out.println("Greatest No= "+a);
        }
        else if(b>a && b>c)
        {
            System.out.println("Greatest No = "+b);
        }
        else
        {
            System.out.println("Greatest No = "+c);
        }
    }
}
```

14.)

14.)

```
p15.java 1 ●
p15.java > p15 > main(String[])
1 // WAP to input coordinates of a point and check the point is in which Quadrant.
2 import java.util.*;
3
4 class p15
5 {
6     public static void main(String[] args)
7     {
8         int x,y;
9         Scanner s = new Scanner(System.in);
10        System.out.print(s:"Enter value for x : ");
11        x=s.nextInt();
12        System.out.println (x:"Enter value for y : ");
13        y=s.nextInt();
14
15        if(x>0 && y>0)
16        {
17            System.out.printf(format:"The point ( %d,%d) is in first quadrant \n",x,y);
18        }
19        else if (x<0 && y>0)
20        {
21            System.out.printf(format:"The point ( %d,%d) is in Second quadrant \n",x,y);
22        }
23        else if (x<0 && y<0)
24        {
25            System.out.printf(format:"The point ( %d,%d) is in Third quadrant \n",x,y);
26        }
27        else if (x>0 && y<0)
28        {
29            System.out.printf(format:"The point ( %d,%d) is in Fourth quadrant \n",x,y);
30        }
31        else if (x>0 && y<0)
32        {
33            System.out.printf(format:"The point ( %d,%d) is at origin \n",x,y);
34        }
35        else {
36            System.out.println(x:"The point is located either X-axis or Y-axis");
37        }
38    } }
```

15.)



```
p16.java × +
File Edit View

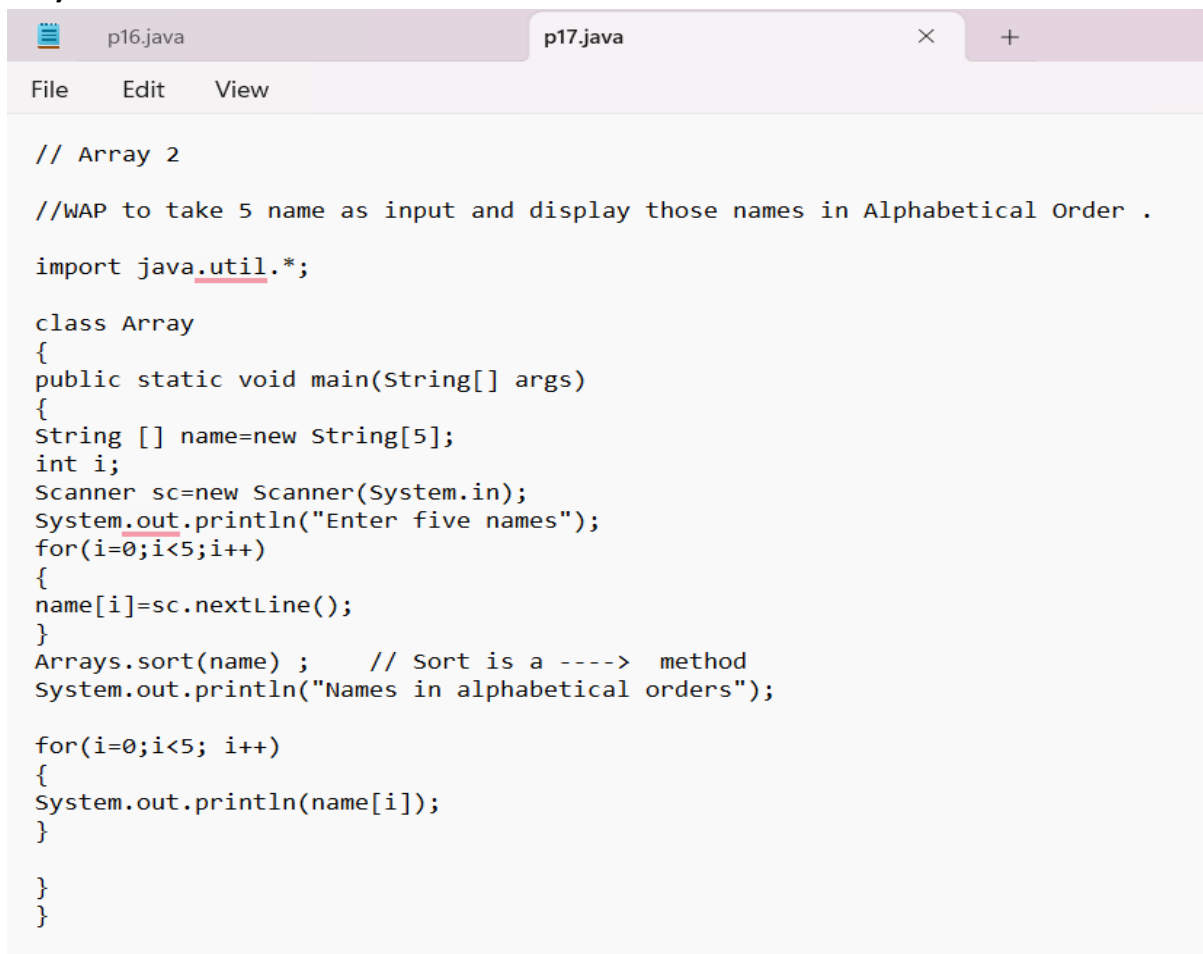
// WAP to make a simple calculator using SWITCH.

import java.util.*;

class Calc
{
    public static void main(String[] args)
    {
        int a,b,res,ch;
        Scanner s=new Scanner(System.in);
        System.out.println(" Enter two numbers ");
        a=s.nextInt();
        b=s.nextInt();
        System.out.println("1 ->Addition ");
        System.out.println("2 ->Subtraction ");
        System.out.println("3 ->Multiplication");
        System.out.println("4 ->Division ");
        ch=s.nextInt();

        switch(ch)
        {
            case 1 :
                res=a+b;
                System.out.println("Result = "+res);
                break;
            case 2:
                res=a-b;
                System.out.println("Result = "+res);
                break;
            case 3:
                res=a*b;
                System.out.println("Result = "+res);
                break;
            case 4 :
                res=a/b;
                System.out.println("Result = "+res);
                break;
            default:
                System.out.println("Invalid choice");
                break;
        }
    }
}
```


16.)



The screenshot shows a Java IDE with two tabs: 'p16.java' and 'p17.java'. The 'p16.java' tab is active. The menu bar includes 'File', 'Edit', and 'View'. The code in 'p16.java' is as follows:

```
// Array 2

//WAP to take 5 name as input and display those names in Alphabetical Order .

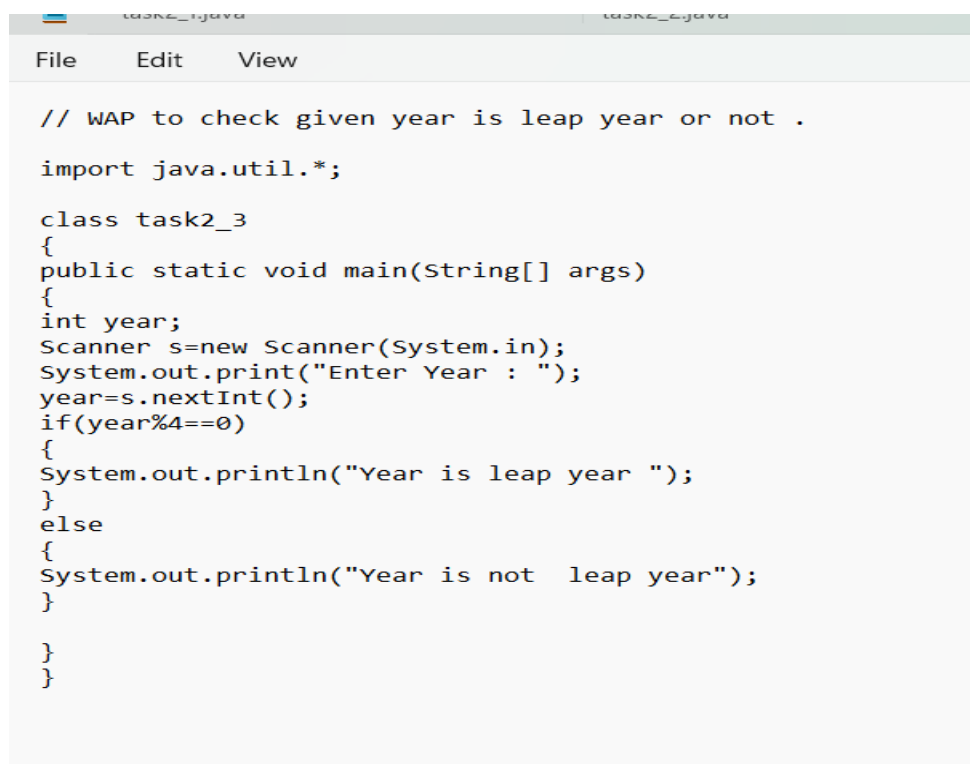
import java.util.*;

class Array
{
    public static void main(String[] args)
    {
        String [] name=new String[5];
        int i;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter five names");
        for(i=0;i<5;i++)
        {
            name[i]=sc.nextLine();
        }
        Arrays.sort(name) ;    // Sort is a ----> method
        System.out.println("Names in alphabetical orders");

        for(i=0;i<5; i++)
        {
            System.out.println(name[i]);
        }

    }
}
```

17.)



The screenshot shows a Java IDE with two tabs: 'task2_1.java' and 'task2_2.java'. The 'task2_2.java' tab is active. The menu bar includes 'File', 'Edit', and 'View'. The code in 'task2_2.java' is as follows:

```
// WAP to check given year is leap year or not .

import java.util.*;

class task2_3
{
    public static void main(String[] args)
    {
        int year;
        Scanner s=new Scanner(System.in);
        System.out.print("Enter Year : ");
        year=s.nextInt();
        if(year%4==0)
        {
            System.out.println("Year is leap year ");
        }
        else
        {
            System.out.println("Year is not leap year");
        }

    }
}
```

18.)

```
// Salary Calculate

import java.util.*;

class task2_1
{
    public static void main(String [] args)

    {
        double bs,hra,da,gs;
        Scanner s=new Scanner(System.in);
        System.out.print("Enter basic Salary : ") ;
        bs=s.nextDouble();

        if(bs<=4000)
        {
            hra=bs*10/100;
            da=bs*50/100;
        }

        else if(bs>4000 && bs<=8000)
        {
            hra=bs*20/100;
            da=bs*60/100;
        }

        else if(bs>8000 && bs<=12000)
        {
            hra=bs*25/100;
            da=bs*60/100;
        }

        else
        {
            hra=bs*30/100;
            da=bs*80/100;
        }
        gs=bs+hra+da;

        System.out.print("*****PAY SLIP***** " );
        System.out.println ("Basic Salary =" +bs);
        System.out.println("House Rent Allownces = "+hra);
        System.out.println ("Dearness Allownces = "+da);
        System.out.println ("Gross Salary = "+gs);

    }
}
```

19.)



The screenshot shows a Java IDE with two tabs: 'task2_1.java' and 'task2_2.java'. The 'task2_2.java' tab is active, displaying a Java program. The menu bar includes 'File', 'Edit', and 'View'. The code is as follows:

```
// Electricity Bill Calculate.....

import java.util.*;

class task2_2
{
    public static void main(String [] args )
    {
        int unit;
        double bill;
        Scanner s=new Scanner(System.in);
        System.out.print("Enter number of units cousumed : ");
        unit=s.nextInt();

        if(unit<=150)
        {
            bill=unit*2.40;
        }
        else if (unit>150 && unit<=300)
        {
            bill=(150*2.40)+(unit-150)*3.00;
        }
        else
        {
            bill=(150*2.40)+(150*3.00)+(unit-300)*3.20;
        }
        System.out.println( "Bill = "+bill);

    }
}
```

20.)

20.)

```
// Write a program to find Roots of Quadratic Equation  $ax^2+bx+c=0$ .  
//  
  
import java.util.*;  
  
class task2_4  
{  
    public static void main(String[] args)  
    {  
        double a,b,c,d,r1,r2;  
        Scanner s=new Scanner(System.in);  
        System.out.println("Enter value for a,b and c ");  
        a=s.nextDouble();  
        b=s.nextDouble();  
        c=s.nextDouble();  
        d=(b*b-4*a*c);  
  
        if(d<0)  
        {  
            System.out.println("Roots are imaginary");  
        }  
        else  
        {  
            r1=(-b+Math.sqrt(d))/(2*a);  
            r2=(-b-Math.sqrt(d))/(2*a);  
            System.out.println("Root1 = "+r1);  
            System.out.println("Root2= "+r2);  
        }  
  
    }  
}
```

OOPS

1.)

```
Converter.java 1 X
Converter.java > ...
1  // WAP to take a decimal number as input and convert it into binary ,
2  // octal and hexa-decimal equivalent .
3
4  import java.util.*;
5
6  class Converter
7  {
8      public static void main(String [] args )
9      {
10         int n;
11         Scanner s = new Scanner(System.in);
12         System.out.println("Enter a number : ");
13         n=s.nextInt();
14
15         // Int conver to string
16         System.out.print("Binary Format = "+Integer.toString(n,radix:2));
17         System.out.print(" Octal Format = "+Integer.toString(n,radix:8));
18         System.out.print(" Hexa-Decimal Format = "+Integer.toString(n,radix:16));
19     }
20 }
21
```

2.)

multiinheritence.java X

multiinheritence.java > A

```
1 // WAP to demonstrate concept of multi-level inheritance
2
3 class A
4 {
5     void showA()
6     {
7         System.out.println("This message from Class A");
8     }
9 }
10
11 class B extends A
12 {
13     void showB()
14     {
15         System.out.println("This message from class B");
16     }
17 }
18
19 class C extends B
20 {
21     void showC()
22     {
23         System.out.println("This message from class C");
24     }
25 }
26
27 class multiinheritence
28 {
29     Run | Debug
30     public static void main(String [] args)
31     {
32         C c = new C();
33         c.showA();
34         c.showB();
35         c.showC();
36     }
37 }
```

3.)

```
MyUtil.java 1 ×
MyUtil.java > MyUtil > greatest(int, int)
1 // In this we create a package with name mypack and inside this package, we create a class with name MyUtil.
2 // In MyUtil class we create two methods add() and greatest. add() method return sum of two numbers
3 // and greatest () method return greatest number in two numbers.
4
5
6 package mypack;
7 public class MyUtil
8 {
9     public int add(int x , int y)
10    {
11        return (x+y);
12    }
13    public int greatest (int x, int y)
14    {
15        if (x>y)
16        {
17            return x;
18        }
19        else
20        {
21            return y ;
22        }
23
24
25    }
26
```

4.)

4.)

```
OverLoadingDemo.java 1 ●
OverLoadingDemo.java > OverLoadingDemo > main(String[])
1 // Polimorphism -----> Over-loading
2 //
3
4 import java.util.*;
5
6 class Figure {
7     int area(int s) // Area of Square
8     {
9         return s * s;
10    }
11
12    int area(int l, int b) // Area of Rectangle
13    {
14        return l * b;
15    }
16
17    double area(double r) // Area of Circle
18    {
19        return 3.14 * r * r;
20    }
21 }
22
23 class OverLoadingDemo {
24     Run | Debug
25     public static void main(String[] args) {
26         int s, l, b, a1, a2;
27         double r, a3;
28         Scanner sc = new Scanner(System.in);
29         Figure fig = new Figure();
30         System.out.println(x:"Enter side of Square : ");
31         s = sc.nextInt();
32         System.out.println(x:"Enter length of Rectangle : ");
33         l = sc.nextInt();
34         System.out.println(x:"Enter breadth of Rectangle : ");
35         b = sc.nextInt();
36         System.out.println(x:"Enter radius of Circle : ");
37         r = sc.nextDouble();
38         a1 = fig.area(s);
39         a2 = fig.area(l, b);
40         a3 = fig.area(r);
41         System.out.println("Area of Square = " + a1);
42         System.out.println("Area of Rectangle = " + a2);
43         System.out.print("Area of Circle = " + a3);
44     }
```


5.)

```
OverRidingDemo.java > X
1  // This program demonstrative concept of menthod overriding
2
3  class X
4  {
5      void m1()
6      {
7          System.out.println("m1 of X");
8      }
9      void m2()
10     {
11         System.out.println("m2 of X");
12     }
13 }
14
15 class Y extends X
16 {
17     // Here overriding m1() method of class X
18     void m1()
19     {
20         System.out.println(" m1 of Y");           // -----> Over-write..
21     }
22     void m3()
23     {
24         System.out.println("m3 of Y");
25     }
26 }
27
28 class OverRidingDemo
29 {
30     Run | Debug
31     public static void main(String[] args)
32     {
33         X x1=new X();
34         x1.m1();           // m1 of X
35         x1.m2();           // m2 of X
36
37         Y y1=new Y();
38         y1.m1();           // m1 of Y
39         y1.m2();           // m2 of X
40         y1.m3();           // m3 of Y
41     }
```

6.)

Test.java > ...

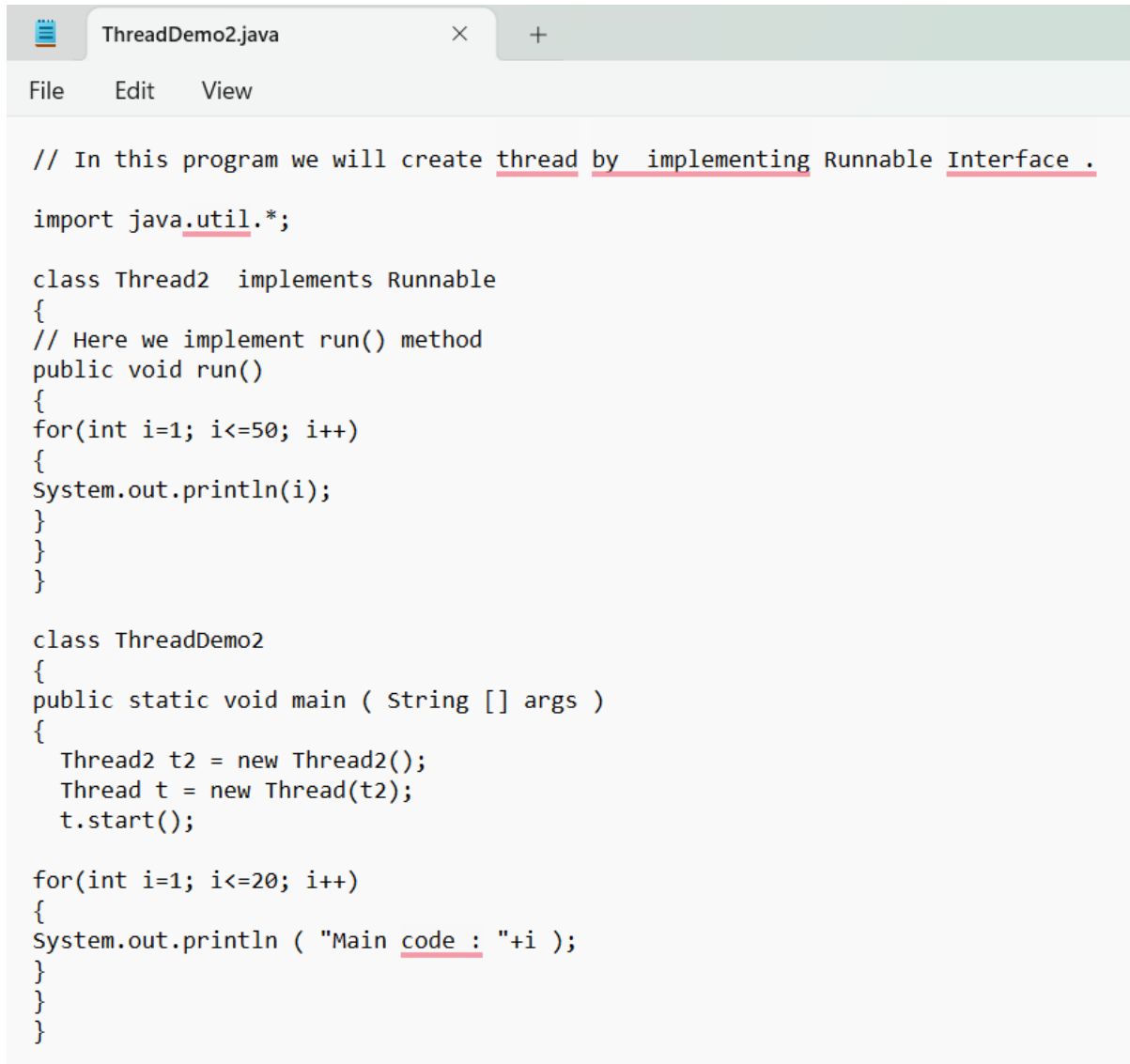
```
1 // In this program file we import mypack.MyUtil and test MyUtil class .
2 // and Test MyUtil class
3
4 import java.util.Scanner;
5 import mypack.MyUtil;
6
7 class Test
8 {
9     Run | Debug
10    public static void main (String [] args)
11    {
12        int a,b,s,g;
13        Scanner sc=new Scanner(System.in);
14        MyUtil mu=new MyUtil();
15        System.out.println(x:"Enter two number : ");
16        a=sc.nextInt();
17        b=sc.nextInt();
18        s=mu.add(a,b);
19        g=mu.greatest(a,b);
20        System.out.println("Sum = "+s);
21        System.out.println("Greatest No. = "+g);
22    }
23 }
```

7.)

7.)

```
ThreadDemo1.java > Thread1
1  // We create a thread in this program by extending Thread Class .
2  ⚡
3  class Thread1 extends Thread
4  {
5      // Here we override run() method of Thread class
6      public void run()
7      {
8          for(int i=1; i<=50; i++)
9          {
10             System.out.println(i);
11             try{
12                 Thread.sleep(millis:500);
13             }
14             catch(InterruptedException e)
15             {
16             }
17         }
18     }
19 }
20 }
21 }
22
23 class ThreadDemo1
24 {
25     Run | Debug
26     public static void main ( String [] args ) throws InterruptedException
27     {
28         Thread1 t=new Thread1();
29         t.start();
30         for(int i=1; i<=20; i++)
31         {
32             System.out.println("Main : " +i);
33             Thread.sleep(millis:1000);
34         }
35     }
```

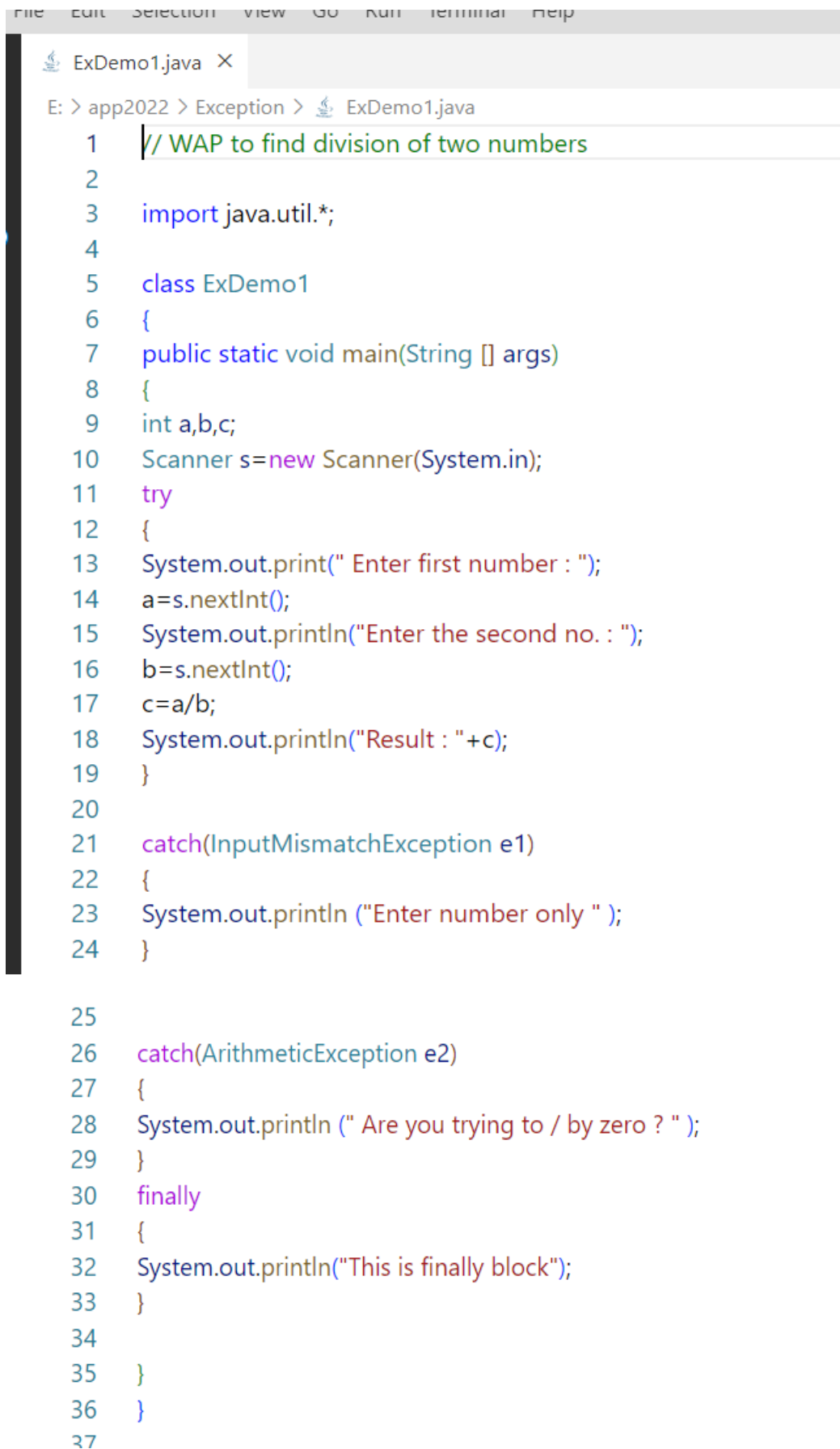
8.)

A screenshot of a Java IDE window titled 'ThreadDemo2.java'. The window has a menu bar with 'File', 'Edit', and 'View'. The code is as follows:

```
// In this program we will create thread by implementing Runnable Interface .  
import java.util.*;  
  
class Thread2 implements Runnable  
{  
    // Here we implement run() method  
    public void run()  
    {  
        for(int i=1; i<=50; i++)  
        {  
            System.out.println(i);  
        }  
    }  
}  
  
class ThreadDemo2  
{  
    public static void main ( String [] args )  
    {  
        Thread2 t2 = new Thread2();  
        Thread t = new Thread(t2);  
        t.start();  
  
        for(int i=1; i<=20; i++)  
        {  
            System.out.println ( "Main code : "+i );  
        }  
    }  
}
```

Exceptions

9.)



The screenshot shows an IDE window titled 'ExDemo1.java'. The code is a Java program designed to handle exceptions during division. It uses a try-catch-finally block to manage potential runtime errors. The try block contains the logic for reading two integers and performing division. The first catch block handles InputMismatchException, prompting the user to enter a number only. The second catch block handles ArithmeticException, prompting the user to avoid division by zero. The finally block ensures that a message is printed regardless of whether an exception occurred.

```
File Edit Selection View Go Run Terminal Help
ExDemo1.java X
E: > app2022 > Exception > ExDemo1.java
1  // WAP to find division of two numbers
2
3  import java.util.*;
4
5  class ExDemo1
6  {
7      public static void main(String [] args)
8      {
9          int a,b,c;
10         Scanner s=new Scanner(System.in);
11         try
12         {
13             System.out.print(" Enter first number : ");
14             a=s.nextInt();
15             System.out.println("Enter the second no. : ");
16             b=s.nextInt();
17             c=a/b;
18             System.out.println("Result : "+c);
19         }
20
21         catch(InputMismatchException e1)
22         {
23             System.out.println ("Enter number only " );
24         }
25
26         catch(ArithmeticException e2)
27         {
28             System.out.println (" Are you trying to / by zero ? " );
29         }
30         finally
31         {
32             System.out.println("This is finally block");
33         }
34
35     }
36 }
37
```

10.)

```
ExDemo2.java X ExDemo1.java ●
E: > app2022 > Exception > ExDemo2.java
1 //WAP to print numbers from 1-100 with delay of 1-1 second
2
3 class ExDemo2
4 {
5 public static void main (String [] args)
6 {
7 for (int i=1; i<=100; i++ )
8 {
9 System.out.println(i+"\t");
10 }
11 // try
12 // {
13 // Thread.sleep(100);
14 // }
15 // catch(InterrruptedException ex)
16 // {
17 //
18 // }
19
20
21 }
22 }
```

1.)

```
p12.java > p12
1  // print numbers 1 to 100.
2
3
4  class p12
5  {
6      Run | Debug
7      public static void main(String [] args)
8      {
9          int c;
10         c=1;
11         while(c<=100)
12         {
13             System.out.println(c+"");
14             c++;
15         }
16     }
```

2.)

```
OverloadingDemo.java 1  ClassDemo2.java 2  p17.java 1
p13.java > ...
1  // WAP a program to find factorial of given number .
2
3  import java.util.*;
4
5
6  class fact
7  {
8      Run | Debug
9      public static void main(String[] args)
10     {
11         int n , f=1;
12         Scanner scanner=new Scanner(System.in);
13         System.out.println(x:"Enter a number : ");
14         n=scanner.nextInt();
15         while(n>0)
16         {
17             f=f*n;
18             n--;
19         }
20         System.out.println("Factorial : "+f);
21     }
22 }
```

3.)

```
// In this program we will create thread by  
// implementing Runnable Interface .
```

```
import java.util.*;
```

```
class Thread2 implements Runnable  
{  
    // Here we implement run() method  
    public void run()  
    {  
        for(int i=1; i<=50; i++)  
        {  
            System.out.println(i);  
        }  
    }  
}
```

```
class ThreadDemo2  
{  
    public static void main ( String [] args )  
    {  
        Thread2 t2 = new Thread2();  
        Thread t = new Thread(t2);  
        t.start();
```

```
        for(int i=1; i<=20; i++)  
        {  
            System.out.println ( "Main code : "+i );  
        }  
    }  
}
```



```
D:\app2022\oops>javac ThreadDemo2.java
```

```
D:\app2022\oops>java ThreadDemo2
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36
```

```
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50
```

```
Main code : 1  
Main code : 2  
Main code : 3  
Main code : 4  
Main code : 5  
Main code : 6  
Main code : 7  
Main code : 8  
Main code : 9  
Main code : 10  
Main code : 11  
Main code : 12  
Main code : 13  
Main code : 14  
Main code : 15  
Main code : 16  
Main code : 17  
Main code : 18  
Main code : 19  
Main code : 20
```

4.)

```
// We create a thread in this program by extending Thread Class .

class Thread1 extends Thread
{
    // Here we override run() method of Thread class
    public void run()
    {
        for(int i=1; i<=50; i++)
        {
            System.out.println(i);
            try{
                Thread.sleep(500);
            }
            catch(InterruptedException e)
            {

            }

        }
    }
}

class ThreadDemo1
{
    public static void main ( String[] args ) throws InterruptedException
    {
        Thread1 t=new Thread1();
        t.start();
        for(int i=1; i<=20; i++)
        {
            System.out.println("Main : " +i);
            Thread.sleep(1000);
        }
    }
}
```

```
D:\app2022\oops>javac ThreadDemo1.java
```

```
D:\app2022\oops>java ThreadDemo1
```

```
1
Main : 1
Main : 2
Main : 3
Main : 4
Main : 5
Main : 6
Main : 7
Main : 8
Main : 9
Main : 10
Main : 11
Main : 12
Main : 13
Main : 14
Main : 15
Main : 16
Main : 17
Main : 18
Main : 19
Main : 20
2
3
4
5
6
7
8
9
10
11
12
13
14
15
```

```
D:\app2022\oops>java ThreadDemo1
```

```
1
Main : 1
2
3
Main : 2
4
Main : 3
5
6
Main : 4
7
8
9
Main : 5
10
11
Main : 6
12
Main : 7
13
14
Main : 8
15
16
Main : 9
17
18
Main : 10
19
20
Main : 11
21
22
Main : 12
23
24
Main : 13
25
```

[Thread
sleep]

5.)

```
*Convertor.java - Notepad
File Edit View

// WAP to take a decimal number as input and convert it into binary ,
// octal and hexa-decimal equivalent .

import java.util.*;

class Convertor
{
    public static void main(String [] args )
    {
        int n;
        Scanner s = new Scanner(System.in);
        System.out.println("Enter a number : ");
        n=s.nextInt();

        // Int conver to string
        System.out.print("Binary Format = "+Integer.toString(n,2));
        System.out.print(" Octal Format = "+Integer.toString(n,8));
        System.out.print(" Hexa-Decimal Format = "+Integer.toString(n,16));
    }
}
```

```
D:\app2022\oops>notepad Convertor.java
```

```
D:\app2022\oops>javac Convertor.java

D:\app2022\oops>java Convertor
Enter a number :
9
Binary Format = 1001

D:\app2022\oops>javac Convertor.java

D:\app2022\oops>java Convertor
Enter a number :
2
Binary Format = 10Octal Format = 2Hexa-Decimal Format = 2
D:\app2022\oops>java Convertor
Enter a number :
8999
Binary Format = 100011001001110Octal Format = 21447Hexa-Decimal Format = 2327
D:\app2022\oops>
```

6.)

```
p15.java > ...  
1 // WAP to print table of given number.  
2  
3 import java.util.*;  
4  
5 class table  
6 {  
    Run | Debug  
7 public static void main(String [] args)  
8 {  
9     int n,t,i;  
10    Scanner sc=new Scanner(System.in);  
11    System.out.println(x:"Enter a number :");  
12    n=sc.nextInt();  
13    for(i=1; i<=10; i++)  
14    {  
15        t=n*i;  
16        //System.out.println(n+"*"+i+"="+t);  
17        System.out.printf(format:"%d*%d=%d\n",n,i,t);  
18    }  
19  
20 }  
21 }
```


7.)

p16.java > ...

```
1  // WAP to print fibonacci series up to n terms .
2  //
3  //
4
5  import java.util.*;
6
7  class p16
8  {
    Run | Debug
9  public static void main(String [] args)
10 {
11     int n1=0, n2=1, n3, n, i;
12     Scanner scanner=new Scanner(System.in);
13     System.out.println(x:"How many terms you want in series? ")
14     n=scanner.nextInt();
15     System.out.print(n1+" " +n2+ " ");
16     for(i=1; i<=n-2; i++)
17     {
18         n3=n1+n2;
19         System.out.print(n3+" ");
20         n1=n2;
21         n2=n3;
22     }
23
24 }
25 }
26
```

8.)

 p18.java > ...

```
1 // WAP to take 10 numbers as input. Store these numbers in array and find sum and average.
2 
3
4 import java.util.*;
5
6 class p18
7 {
8     Run | Debug
9     public static void main(String [] args)
10    {
11        int [] list=new int[10];
12        int i, sum=0;
13        double avg;
14        Scanner sc=new Scanner(System.in);
15        System.out.println(x:"Enter the ten numbers to the list");
16        for(i=0; i<10; i++)
17        {
18            list[i]=sc.nextInt();
19            sum=sum+list[i];
20        }
21        // typecasting
22        avg=(float)sum/10;
23        System.out.println("Sum = "+sum);
24        System.out.println("Average = "+avg);
25    }
```

9.)

p19.java > ...

```
1 //WAP to take five names as input and display names in Alphabetical order.
2
3
4 import java.util.*;
5
6 class p19
7 {
8     Run | Debug
9     public static void main(String [] args)
10    {
11        String [] name=new String[5];
12        int i;
13        Scanner s=new Scanner(System.in);
14        System.out.println(x:"Enter Five names");
15        for(i=0; i<name.length; i++)
16        {
17            name[i]=s.nextLine();
18        }
19        Arrays.sort(name);
20        System.out.println(x:"Names in alphabetical order");
21        // foreach loop
22        for(String n:name)
23        {
24            System.out.println(n);
25        }
26    }
```


10.)

```
OverloadingDemo.java 1  ClassDemo2.java 2  p17.java 1  p20.java 1 X
p20.java > ...
1  // Input a String to convert Upper Case and Lower Case and also find length .
2
3  import java.util.*;
4  class p20
5  {
6      Run | Debug
7      public static void main(String [] args)
8      {
9          String name;
10         Scanner s=new Scanner(System.in);
11         System.out.println(x:"Enter the name :");
12         name=s.nextLine();
13         System.out.println("Name in upper case =" +name.toUpperCase());
14         System.out.println("Name in lower case =" +name.toLowerCase());
15         System.out.println("Length of your name =" +name.length());
16     }
17 }
```

11.)

```

1
2  import java.util.*;
3
4  class p21
5  {
6      Run | Debug
7      public static void main(String[]args)
8      {
9          String str1,str2;
10         Scanner s=new Scanner(System.in);
11         System.out.println(x:"Enter First String:");
12         str1=s.nextLine();
13         System.out.println(x:"Enter Second String : ");
14         str2=s.nextLine();
15         if(str1.equals(str2)==true)
16         {
17             System.out.println(x:"Both strings are equal");
18         }
19         else
20         {
21             System.out.println(x:"Both Strings are not equals");
22         }
23     }
24 }
```

12.)

```
p22.java > ...
1 // WAP to compare two Strings for equality .
2
3 import java.util.*;
4
5 class p22
6 {
7     Run | Debug
8     public static void main(String[] args)
9     {
10         String str1, str2;
11         Scanner s = new Scanner(System.in);
12         System.out.print(s: "Enter first string : ");
13         str1 = s.nextLine();
14         System.out.print(s: "Enter Second string : ");
15         str2 = s.nextLine();
16         if(str1.equalsIgnoreCase(str2) == true)
17         {
18             System.out.println(x: "Both Strings are equal");
19         }
20         else
21         {
22             System.out.println(x: "Both Strings are not equal");
23         }
24     }
25 }
```

13.)

```
p23.java > ...
1 // Word Counter
2
3 import java.util.*;
4
5 class split
6 {
7     Run | Debug
8     public static void main(String[] args)
9     {
10         String sen;
11         Scanner scanner = new Scanner(System.in);
12         System.out.print(s: "Enter a Sentence : ");
13         sen = scanner.nextLine();
14         String [] words = sen.split(regex: " ");
15         int l = words.length;
16         System.out.println("No of words : " + l);
17     }
18 }
```

14.)

```
OverridingDemo.java  ClassDemo2.java  p17.java  p24.java
p24.java > ...
1  // FindWhat and replace
2  ⚡
3  import java.util.*;
4
5  class p24
6  {
7      Run | Debug
8      public static void main(String [] args)
9      {
10         String sen,fw,rw;
11         Scanner sc=new Scanner(System.in);
12         System.out.println(x:"Enter a Sentence : ");
13         sen=sc.nextLine();
14         System.out.print(s:"Find What ? ");
15         fw=sc.nextLine();
16         System.out.print(s:"Replace with : ");
17         rw=sc.nextLine();
18         System.out.println("Modified Sentence = " +sen.replace(fw,rw));
19     }
20 }
21
```

15.)

```
task3.java > Quadratic
1  // 3. WAP to find roots of Quadratic equation .
2  // ax2+bx+c=0
3  // ax*x+b*x+c=0
4  // c= - (ax*x+b*x)
5
6  import java.util.*;
7  class Quadratic
8  {
9      Run | Debug
10     public static void main(String [] args)
11     {
12         int x;
13     }
```

16.)

p25.java > ...

```
1  // WAP to check given string is palindrome or not .
2  // palindrome --> reverse string is same
3
4  import java.util.*;
5
6  class reverse
7  {
8      Run | Debug
9      public static void main(String[] args)
10     {
11         String str, revstr="";
12         Scanner sc=new Scanner(System.in);
13         System.out.print(s:"Enter a String : ");
14         str=sc.nextLine();
15
16         for(int i=str.length()-1; i>=0; i--)
17         {
18             revstr=revstr+str.charAt(i);
19         }
20         if(str.equalsIgnoreCase(revstr)==true)
21         {
22             System.out.println(x:"String is palindrome");
23         }
24         else
25         {
26             System.out.println(x:"String is not Palindrome");
27         }
28     }
29 }
```

17.)

```
OverloadingDemo.java 1 ClassDemo2.java 2 p17.java 1 p26.java 1 X Cl.
p26.java > ...
1 //WAP to convert a decimal number to its binary, octal, hexa-decimal.
2
3
4 import java.util.*;
5
6 class p26
7 {
8     Run | Debug
9     public static void main(String[] args)
10    {
11        int n;
12        Scanner s=new Scanner(System.in);
13        System.out.print(s:"Enter a number : ");
14        n=s.nextInt();
15        System.out.println("Binary format="+Integer.toString(n,radix:2));
16        System.out.println("Octal format="+Integer.toString(n,radix:8));
17        System.out.println("Hexa-decimal format="+Integer.toString(n,radix:16));
18    }
19 }
```

18.)

```
OverloadingDemo.java 1 ClassDemo2.java 2 p17.java 1 p27
p27.java > ...
1 // WAP to find sum of two numbers using user defined method .
2
3 import java.util.*;
4
5 class p27
6 {
7     static int add(int x, int y)
8     {
9         return(x+y);
10    }
11
12    Run | Debug
13    public static void main(String[] args)
14    {
15        int a,b,c;
16        Scanner s=new Scanner(System.in);
17        System.out.println(x:"Enter two number");
18        a=s.nextInt();
19        b=s.nextInt();
20        c=add(a,b);
21        System.out.println("Sum= "+c);
22    }
23 }
```

19.)

```
p28.java > p28 > main(String[])
1 // WAP to find the volume of cuboid.
2
3 import java.util.*;
4
5 class p28
6 {
7     int volume(int l,int b,int h)
8     {
9         return(l*b*h);
10    }
11    Run | Debug
12    public static void main(String[] args)
13    {
14        int x,y,z,v;
15        Scanner s=new Scanner(System.in);
16        System.out.print(s:"Enter length of cuboid : ");
17        x=s.nextInt();
18        System.out.print(s:"Enter breadth of cuboid : ");
19        y=s.nextInt();
20        System.out.print(s:"Enter heighth of cuboid : ");
21        z=s.nextInt();
22
23        // class ka object create
24        // first p28 --> variable bnane ke liye
25        // p --> store
26        // new p28() --> object create
27        // new p28().volume ---> annonymous
28        p28 p=new p28(); // --> object with reference variable
29        v=p.volume(x,y,z);
30        System.out.println("Volume of cuboid = "+v);
31    }
```

20.)

```
task2.java > ...
1 // 2. WAP to check given year is leap year or not
2
3 import java.util.*;
4
5 class Leapyear
6 {
7     Run | Debug
8     public static void main(String[] args)
9     {
10         int n;
11         Scanner sc=new Scanner(System.in);
12         System.out.println(x:"Enter a year : ");
13         n=sc.nextInt();
14         if(n%4==0)
15         {
16             System.out.println(x:"Leap year");
17         }
18         else
19         {
20             System.out.println(x:"Number is not a leap year");
21         }
22     }
23 }
```

21.)

```
s > ClassDemo1.java > MyClass
1 // This program demonstrate concept of class.
2
3 class MyClass
4 {
5     void sayHello()
6     {
7         System.out.println(x:"Hello World !!");
8     }
9 }
0 class ClassDemo
1 {
2     Run | Debug
3     public static void main(String [] args)
4     {
5         MyClass m=new MyClass();
6         m.sayHello();
7     }
8 }
```

22.)

```
1
2 // This program demonstrate concept of class.
3
4 class MyClass
5 {
6     void sayHello( String name)
7     {
8         System.out.println("Hello "+name);
9     }
10 }
11 class ClassDemo
12 {
13     Run | Debug
14     public static void main(String [] args)
15     {
16         MyClass m=new MyClass();
17         m.sayHello(name:"Soni Nishad");
18     }
19 }
```

23.)

24.)

ClassDemo3.java > Employee > empname

```
//
```

```
class Employee
```

```
{
```

```
    int empid;
```

```
// Instance Variable --> Instance Variable ve variable hote hai, jo class ke andar declare kiye jate hai
```

```
    String empname;    // Instance Variable
```

```
    double salary;    // Instance Variable
```

```
    void setValue(int eid, String ename, double sal)
```

```
    {
```

```
        empid=eid;
```

```
        empname=ename;
```

```
        salary=sal;
```

```
    }
```

```
    void display()
```

```
    {
```

```
        System.out.println("Employee Id="+empid);
```

```
        System.out.println("Employee Name="+empname);
```

```
        System.out.println("Employee Salary="+salary);
```

```
    }
```

```
}
```

```
class Demo3
```

```
{
```

```
    Run | Debug
```

```
    public static void main(String [] args)
```

```
    {
```

```
        Employee e1=new Employee();
```

```
        e1.setValue(eid:1001,ename:"soni nishad",sal:40000.0);
```

```
        e1.display();
```

```
        ..
```

```
        ..
```

```
        Employee e2=new Employee();
```

```
        e2.setValue(eid:1002,ename:"soumya nishad",sal:30000.0);
```

```
        e2.display();
```

```
    }
```

```
}
```

24.)

>  ClassDemo4.java >  Student

```
// Constructor
class Student
{
    int rollno;
    String name;
    double fee;

    Student(int rno, String nm, double f) // Constructor
    {
        rollno=rno;
        name=nm;
        fee=f;
    }

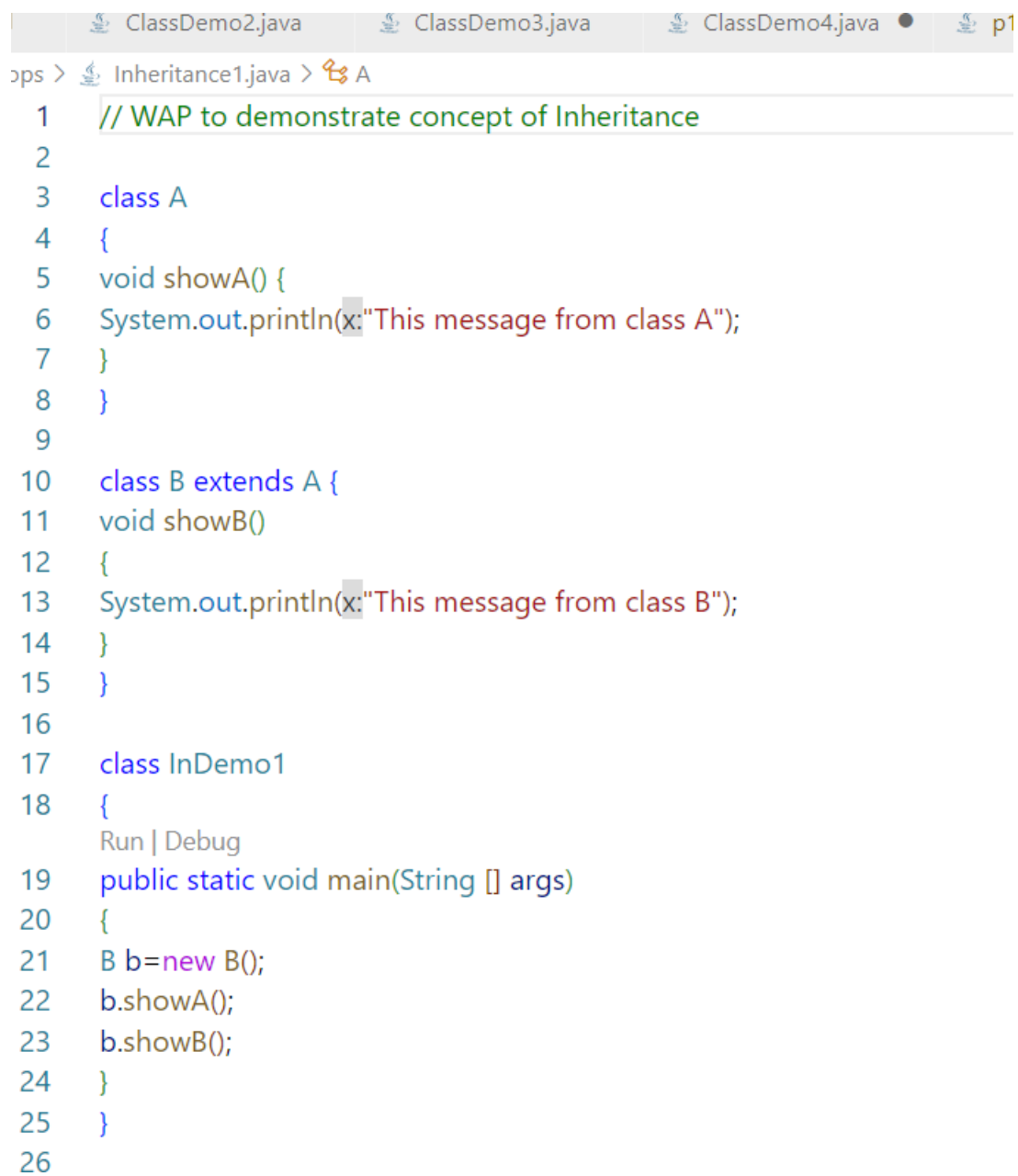
    void display()
    {
        System.out.println("Roll No="+rollno);
        System.out.println("Name="+name);
        System.out.println("Fee="+fee);
    }
}

class Demo4
{
    Run | Debug
    public static void main(String [] args)
    {
        Student s=new Student(rno:1001,nm:"soni",f:2000.0);
        s.display();
    }
}
```

25.)

```
ps > ClassDemo5.java > Student > Student(int, String, double)
1 // Constructor
2
3
4 class Student
5 {
6     int rollNo;
7     String name;
8     double fee;
9
10 Student(int rollNo, String name, double fee )// Constructor
11 {
12 }
13
14 void display()
15 {
16     System.out.println("Roll No="+rollNo);
17     System.out.println("Name="+name);
18     System.out.println("Fee="+fee);
19 }
20 }
21
22 class Demo4
23 {
24     Run | Debug
25     public static void main(String [] args)
26     {
27         Student s=new Student(rollNo:1001,name:"soni",fee:2000.0);
28         s.display();
29     }
30 }
```

26.)



```
ops > Inheritance1.java > A
1 // WAP to demonstrate concept of Inheritance
2
3 class A
4 {
5     void showA() {
6         System.out.println(x:"This message from class A");
7     }
8 }
9
10 class B extends A {
11     void showB()
12     {
13         System.out.println(x:"This message from class B");
14     }
15 }
16
17 class InDemo1
18 {
19     Run | Debug
20     public static void main(String [] args)
21     {
22         B b=new B();
23         b.showA();
24         b.showB();
25     }
26 }
```

27.)

ps > Inheritance2.java > ...

```
1 // Example of Heirarchical Inheritance
2
3 import java.util.*;
4 class Shape {
5     int s;
6     void setValue( int side)
7     {
8         s=side;
9     }
10 }
11
12 class Square extends Shape
13 {
14     int area()
15     {
16         return s*s;
17     }
18 }
19
20 class Cube extends Shape
21 {
22     int volume()
23     {
24         return s*s*s;
25     }
26 }
27
28 class demo2
29 {
30     Run | Debug
31     public static void main(String [] args)
32     {
33         int x,a,v;
34         Scanner sc=new Scanner(System.in);
35
36         Square sq=new Square();
37         System.out.print(s:"Enter side of square : ");
38         x=sc.nextInt();
39         sq.setValue(x);
40         a=sq.area();
41         System.out.println("Area of Square : "+a);
42
43         Cube cu=new Cube();
44         System.out.print(s:"Enter the side of Cube : ");
45         x=sc.nextInt();
46         cu.setValue(x);
47         v=cu.volume();
48         System.out.println("Volume of Cube : "+v);
49     }
50 }
```

28.)

//WAP to demonstrate Multi-level Inheritance.

```
class X
{
    void showX()
    {
        System.out.print(s:"Message from class X ");
    }
}
class Y extends X
{
    void showY()
    {
        System.out.println(x:"Message from class Y ");
    }
}
class Z extends y
{
    void showZ()
    {
        System.out.println(x:"Message from class Z ");
    }
}
```

```
class demo3 {
```

Run | Debug

```
public static void main(String[] args)
{
    Z z=new Z();
    z.showX();
    z.showY();
    z.showZ();
}
```

29.)

```
ops > interfaceDemo1.java > TestInterface > m2()
1 // WAP to understand concept of interface in java.
2 interface MyInterface
3 {
4 void m1(); // public abstract void m1();
5 void m2(); // public abstract void m2();
6 void m3(); // public abstract void m3();
7 }
8
9 class TestInterface implements MyInterface
10 {
11 public void m1()
12 {
13 System.out.println(x:"This message from m1() method ");
14 }
15 public void m2()
16 {
17 System.out.println(x:"This message from m2() method");
18 }
19 public void m3()
20 {
21 System.out.println(x:"This is from m3() method ");
22 }
23
24 Run | Debug
25 public static void main(String [] args)
26 {
27 TestInterface t=new TestInterface();
28 t.m1();
29 t.m2();
30 t.m3();
31 }
```

30.)

ps > interfaceDemo2.java > School



```
1 //First Meeting
2
3 interface School
4 {
5     void registration();
6     void feeSubmission();
7     void batchAllotment();
8 }
9
10 //Second Meeting
11 abstract class Test1 implements School
12 {
13     public void registration()
14     {
15         System.out.println(x:"Business Login of registration");
16     }
17 }
18
19 // Third Meeting
20 abstract class Test2 extends Test1
21 {
22     public void feeSubmission()
23     {
24         System.out.println(x:"Business Logic of FeeSubmission");
25     }
26 }
27
28 // Fourth Meeting
29 class Test3 extends Test2
30 {
31     public void batchAllotment()
32     {
33         System.out.println(x:"Bussiness Logic of batch allotment ");
34     }
35 }
36
37 // Testing
38 class InterfaceDemo2
39 {
40     Run | Debug
41     public static void main(String [] args)
42     {
43         Test3 t = new Test3();
44         t.registration();
45         t.feeSubmission();
46         t.batchAllotment();
47     }
48 }
```


31.)

```
ops > OverloadingDemo.java > Overloading > main(String[])
1 // WAP to demonstrate concept of method overloading
2
3 import java.util.*;
4 class figure
5 {
6     int area(int s)
7     {
8         return s*s;
9     }
10    int area(int l,int b)
11    {
12        return l*b;
13    }
14    double area(double r)
15    {
16        return 3.14*r*r;
17    }
18 }

20 class Overloading
21 {
    Run | Debug
22    public static void main(String [] args)
23    {
24        int s, l, b, a1, a2;
25        double a3, r;
26        Scanner sc=new Scanner(System.in);
27
28        figure fig=new figure();
29        System.out.print(s:"Enter side of Square : ");
30        s=sc.nextInt();
31        System.out.print(s:"Enter length of rectangle : ");
32        l=sc.nextInt();
33        System.out.print(s:"Enter breath of rectangle : ");
34        b=sc.nextInt();
35        System.out.print(s:"Enter radius of circle : ");
36        r=sc.nextDouble();
37        a1=fig.area(s);
38        a2=fig.area(l,b);
39        a3=fig.area(r);
40        System.out.println("Area of square : "+a1);
41        System.out.println("Area of rectangle : "+a2);
42        System.out.println("Area of radius : "+a3);
43    }
44 }
```

32.)

```
ops >  Overriding.java >  Base
1  // WAP to demonstrate concept of Method Overriding.
2
3  class Base
4  {
5      void m1()
6      {
7          System.out.println(x:"m1 of base");
8      }
9      void m2()
10     {
11         System.out.print(s:"m2 of base");
12     }
13 }
14
15 class Derived extends Base
16 {
17     void m1()
18     {
19         System.out.println(x:"m1 of derived");
20     }
21
22     void m3()
23     {
24         System.out.println(x:"m3 of derived");
25     }
26 }
27
28 class OverridingDemo
29 {
    Run | Debug
30     public static void main(String [] args)
31     {
32         Base b=new Base();
33         b.m1();
34         b.m2();
35
36         Derived d=new Derived();
37         d.m1();
38         d.m2();
39         d.m3();
40     }
41 }
```

Java Assignment – 1

1. Write a java program to find area and perimeter of circle.

Sol –

```
import java.util.*;

class circle
{
    public static void main(String[]args)
    {
        double r, a, p;
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the value of r : ");
        r=sc.nextInt();
        a=(3.14)*r*r;
        p=2*(3.14)*r;
        System.out.println("Area : " +a);
        System.out.println("Perimeter : "+p);
    }
}
```

2. Write a java program to calculate compound interest.

// $ci = p(1+r/100)^n - p$; $si = p*n*r/100$;

Sol –

```
import java.util.*;

class compound {
    public static void main(String []args) {
        int p,n,r;
        double ci;
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the value of p,n,r : ");
        p=sc.nextInt();
        r=sc.nextInt();  n=sc.nextInt();
        ci=p*(1+r/100)*n-p;
        System.out.println("Compound Interest :"+ci);
    }
}
```

3. Write a java program to find square root of given number.

Formula – $sr = x * x$;

Sol –

```
import java.util.*;
class Square
{
    public static void main(String [] args)
    {
        double x,sr;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter any number :");
        x=sc.nextDouble();
        sr =x*x;
        System.out.println("Square Root: "+sr);
    }
}
```

4. Write a java program to convert a given number of days to a measure of time given in years, Weeks, and days. For example, 375 days is equal to 1 year 1 week and 3 days (ignore leap year)

Sol—

```
import java.util.*;
class Convert {
    public static void main(String []args)
    {
        int day,year,week,dayno;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter a day :32 ");
        dayno = sc.nextInt();
        year=dayno/365;
        week = (dayno%365)/7;
        day = (dayno%365)%7;
        System.out.println("years : " +year );
        System.out.println("Weeks : " +week );
        System.out.println("Days : " +day );
    } }
```

Java Assignment – 3

1. Write a java program to make a temperature convertor based on user choice. If user input 1 convert temperature from centigrade to Fahrenheit and if user input 2 convert temperature from Fahrenheit to centigrade.

Hint:-

For c to f:-

$$f=(9*c)/5+32;$$

For f to c:-

$$c=(f-32)*5/9;$$

Sol –

```
import java.util.*;
```

```
class convertor
```

```
{
```

```
public static void main(String [] args)
```

```
{
```

```
float c,f;
```

```
Scanner s=new Scanner(System.in);
```

```
System.out.print("Enter a value : ");
```

```
c=s.nextFloat();
```

```
//f=c*9/5+32;
```

```
f=(9*c)/5+32;
```

```
System.out.println(" Centigrate to Fahrenheit : "+f);
```

```
System.out.println("Enter the value of f :");
```

```
f=s.nextFloat();
```

```
c=(f-32)*5/9;
```

```
System.out.println("F to c : "+c);
```

```
}
```

```
}
```

2. Write a java program to find sum and average of five numbers by taking input from user.

Sol –

3. Write a java program to take day number as input and display day of week.

Sol –

```
import java.util.*;
```

```
class day
```

```
{
```

```
public static void main(String [] args)
```

```
{
```

```
int week,day,dayno;
```

```
Scanner sc=new Scanner(System.in);
```

```
System.out.println("Enter a day : ");
```

```
dayno = sc.nextInt();
```

```
week = (dayno%365)/7;
```

```
day = (dayno%365)%7;
```

```
System.out.println( "Weeks:"+week + " Days:" +day);
```

```
}
```

```
}
```

Java Assignment – 2

1.) Accept the salary of an employee from the user. Calculate the gross salary on the

following basis:

BASIC HRA DA

1 – 4000 10% 50%

4001 – 8000 20% 60%

8001 - 12000 25% 70%

12000 and above 30% 80%

Sol –

```
import java.util.*;
class employee {
public static void main(String [] args)
{
double hra, da, gs, bs;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a basic Salary : ");
bs=sc.nextDouble();
```

```
if( bs<=4000)
{
hra=bs*10/100;
da=bs*50/100;
}
else if(bs>4000 && bs<=8000)
{
hra=bs*20/100;
da=bs*60/100;
}
else if(bs>8000 && bs<=12000)
{
hra=bs*25/100;
da=bs*70/100;
}
```

```

else
{
hra=bs*30/100;
da=bs*80/100;
}

gs=hra+bs+da;
System.out.println("Gross Salary : "+gs);
}
}

```

2. Write a java program to make a electricity bill calculator.

Unit Bill

1-150 2.40/unit

For next 151-300 3.00/unit

For next more than 300 3.20/unit

Sol –

```

import java.util.*;
class Electricity
{
public static void main(String [] args)
{
Double unit, bill , EB ;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a Bill unit : ");
unit=sc.nextDouble();
if( unit<=150 )
{
bill=unit*2.40;
}
else if(unit>150 && unit<=300)
{
bill=(150*2.40)+(unit-150)*3.00;
}
}
}

```



```

}
else
{
bill=(150*2.40)+(150*3.00)+(unit-300)*3.20;
}

System.out.println("Total Electricity Bill : "+bill);
}
}

```

3. Write a java program to check given year is leap year or not.

4. Write a java program to find roots of quadratic equation $ax^2 + bx + c = 0$.

Java Assignment – 4

1. Write a java program to reverse the digits of given number.

2. Write a java program to convert binary number to its decimal equivalent.

3. Write a java program to print Fibonacci series up to n terms. Where value of n is entered

by user.

E.g.

0 1 1 2 3 5 8.....n terms

4. Write a java program to check given number is prime or not.

Java Assignment – 5

1. Write a java program to create an array AR with size 10. Take 10 numbers as input and

store in array AR. Now copy even numbers in array EAR and odd numbers in array OAR.

2. Write a java program to create an array with 10 numbers by using input from user. Now

find maximum and minimum number in array.

3. Write a java program to check given string is palindrome or not.

4. Write a java program check occurrence of 'The' word in a sentence.