

# INDEX

[illegible]

Name: Chandi Charan Mahato

Roll NO: 2300970140053

# Program 1

**Objective : WAP in Java to print Hello.**

```
package Cos;

public class Q1printHello {

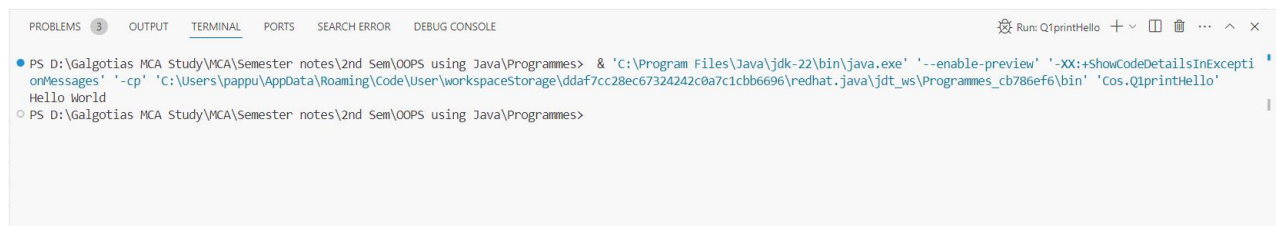
    public static void main(String[] args) {

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

    }

}
```

## Output



The screenshot shows an IDE terminal window with the following content:

```
PROBLEMS 3 OUTPUT TERMINAL PORTS SEARCH ERROR DEBUG CONSOLE
Run: Q1printHello + - [ ] ... ^ x

• PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q1printHello'
Hello World
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 2

**Objective :** WAP in java to understand the difference between print() and println().

```
package Cos;

public class Q2diffBetweenPrintPrintLn {

    public static void main(String[] args) {

        System.out.print("Hello "); //This will print hello world followed by a new line

        System.out.println("world!"); // This will print "Hello from the println" followed by a
        newline

        // Printing multiple values using print()

        System.out.print("Java");

        System.out.print(" is");

        System.out.print(" awesome!");

        // This will print "Java is awesome!" without any newline

        // Printing multiple values using println()

        System.out.println("Java");

        System.out.println(" is");

        System.out.println(" awesome!");

        // This will print each word on a new line

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> d:; cd 'd:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes'; & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\U
ser\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q2diffBetweenPrintPrintLn'
Hello world!
Java is awesome!Java
is
awesome!
```

## Program 3

**Objective : WAP in Java with two classes create a object of first class and call into another class (having main method).**

```
class first{

    public void display(){

        System.out.println("Hello from class first");

    }

}

class second extends first{

    public void display2(){

        System.out.println("Hello from class second");

    }

}

public class Q3TwoClass {

    public static void main(String[] args) {

        second s = new second();

        s.display();

        s.display2();

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> d::; cd 'd:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes'; & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\U
ser\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q3TwoClass'
Hello from class first
Hello from class second
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 4

**Objective : WAP in Java to product of two numbers.**

```
package Cos;

public class Q4Product {

    public static void main(String[] args) {

        int product = 5 * 6 ;

        System.out.println("The product is: " + product);

    }

}
```

## Output

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q4Product'
The product is: 30
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 5

**Objective : WAP in Java to product of two numbers (Input by the user).**

```
package Cos;

import java.util.Scanner;

public class Q5productByUser {

    public static void main(String[] args) {

        try (Scanner sc = new Scanner(System.in)) {

            System.out.println("Enter first number: ");

            int num1 = sc.nextInt();

            System.out.println("Enter second number: ");

            int num2 = sc.nextInt();

            int product = num1 * num2 ;

            System.out.println("The product is: " + product);

        }

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q5productByUser'
Enter first number:
2
Enter second number:
3
The product is: 6
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 6

**Objective : WAP in Java illustrate the concept of local instance and static variable.**

```
public class Q6 {
    // Static variable
    static int staticVar = 50;

    // Instance variable
    int instanceVar = 10;

    public static void main(String[] args) {
        // Local variable
        int localVar = 5;

        System.out.println("Static variable: " + staticVar);

        // Creating an object of the class to access instance variable
        Q6 obj = new Q6();
        System.out.println("Instance variable: " + obj.instanceVar);

        System.out.println("Local variable: " + localVar);

        // Calling a method to demonstrate usage of local, instance, and static variables
        obj.methodExample();
    }

    void methodExample() {
        // Accessing local, instance, and static variables within a method
        int localVar = 20;
        System.out.println("Inside method - Local variable: " + localVar);
        System.out.println("Inside method - Instance variable: " + instanceVar);
        System.out.println("Inside method - Static variable: " + staticVar);
    }
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q6Local'
Static variable: 50
Instance variable: 10
Local variable: 5
Inside method - Local variable: 20
Inside method - Instance variable: 10
Inside method - Static variable: 50
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 7

**Object : WAP in Java to implement implicit and explicit type casting.**

```
package Cos;

public class Q7ImplecitandExplicit {

    public static void main(String [] args){

        int num = 90;

        double num2 = num; //Implicit type casting

        System.out.println("Implicit type casting Integer "+num+" to double "+num2);

        double num3 = 45.00;

        int num4 = (int) num3; //Explicit type casting

        System.out.println("Explicit type casting Integer "+num3+" to double "+num4);

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q7ImplecitandExplicit'
Implicit type casting Integer 90 to double 90.0
Explicit type casting Integer 45.0 to double 45
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```



## Program 8

Object : WAP in Java to implement the various operators in java.

```
package Cos;
```

```
public class Q8Operators {
```

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

```
        // Arithmetic Operators
```

```
        System.out.println("""
```

```
            In Java Arithmetic operators:
```

1. Addition '+' :  $a + b$  or  $5 + 6 = 11$
2. Subtraction '-' :  $a - b$  or  $9 - 8 = 1$
3. Multiplication '\*' :  $a * b$  or  $7 * 4 = 28$
4. Division '/' :  $a / b$  or  $4 / 2 = 2$
5. Modulus '%' :  $a \% b$  or  $5 \% 2 = 1$ """);

```
        // Assignment Operators
```

```
        System.out.println("""
```

```
            In Java Assignment Operators:
```

1. Assignment '=' : Assigns the value of the right operand to the left operand""");

```
        // Conditional Operators
```

```
        System.out.println("""
```

```
            In Java Conditional Operators:
```

1. Ternary Operator '?' : Evaluates a condition and returns one of two values""");

```
        // Increment and Decrement Operators
```

```
        System.out.println("""
```

```
            In Java Increment and Decrement Operators:
```

1. Increment '++' : Increases the value of a variable by 1

2. Decrement '--' : Decreases the value of a variable by 1

```
""");
```

### // Relational Operators

```
System.out.println("""
```

In Java Relational Operators:

1. Greater than '>'
2. Greater than or equal to '>='
3. Less than '<'
4. Less than or equal to '<='
5. Equal to '=='
6. Not equal to '!='

```
""");
```

### // Logical Operators

```
System.out.println("""
```

In Java Logical Operators:

1. Logical AND '&&'
2. Logical OR '||'
3. Logical NOT '!'""");

### // Bitwise Operators

```
System.out.println("""
```

In Java Bitwise Operators:

1. Bitwise AND '&'
2. Bitwise OR '|'
3. Bitwise XOR '^'
4. Bitwise Complement '~'
5. Left Shift '<<'
6. Right Shift '>>'
7. Unsigned Right Shift '>>>'

```

        """);

// Unary Operators

System.out.println("""

In Java Unary Operators:

1. Unary Plus '+'

2. Unary Minus '-'

3. Unary Increment '++'

4. Unary Decrement '--'

5. Logical NOT '!'

6. Bitwise Complement '~'""");
    }
}

```

## Output

```

PROBLEMS 3 OUTPUT TERMINAL PORTS SEARCH ERROR DEBUG CONSOLE

86ef6\bin' 'Cos.Q80Operators'
In Java Arithmetic operators:
1. Addition '+' : a + b or 5 + 6 = 11
2. Subtraction '-' : a - b or 9 - 8 = 1
3. Multiplication '*' : a * b or 7 * 4 = 28
4. Division '/' : a / b or 4 / 2 = 2
5. Modulus '%' : a % b or 5 % 2 = 1

In Java Assignment Operators:
1. Assignment '=' : Assigns the value of the right operand to the left operand
In Java Conditional Operators:
1. Ternary Operator '?': ! Evaluates a condition and returns one of two values

In Java Increment and Decrement Operators:
1. Increment '++' : Increases the value of a variable by 1
2. Decrement '--' : Decreases the value of a variable by 1

In Java Relational Operators:
1. Greater than '>'
2. Greater than or equal to '>='
3. Less than '<'
4. Less than or equal to '<='
5. Equal to '=='
6. Not equal to '!='

In Java Logical Operators:
1. Logical AND '&&'
2. Logical OR '||'
3. Logical NOT '!'

In Java Bitwise Operators:
1. Bitwise AND '&'
2. Bitwise OR '|'
3. Bitwise XOR '^'
4. Bitwise Complement '~'
5. Left Shift '<<'
6. Right Shift '>>'

In Java Unary Operators:
1. Unary Plus '+'
2. Unary Minus '-'
3. Unary Increment '++'
4. Unary Decrement '--'
5. Logical NOT '!'
6. Bitwise Complement '~'
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>

```

## Program 9

**Objective:** WAP in java for constructor overloading.

```
package Cos;

public class Q9ConsOverload {

    int data = 50;

    // constructor overload

    public Q9ConsOverload() {

        this.data = 50;

    }

    public Q9ConsOverload(int data2) {

        this.data = data2;

    }

    public Q9ConsOverload(int data1,int data2) {

        this.data = data1 + data2;

    }

    public static void main(String[] args) {

        Q9ConsOverload obj1 = new Q9ConsOverload(10);

        Q9ConsOverload obj2 = new Q9ConsOverload(10, 20);

        Q9ConsOverload obj3 = new Q9ConsOverload();

        System.out.println("Constructor no parameters = " + obj1.data);

        System.out.println("Constructor with single parameters = " + obj2.data);

        System.out.println("Constructor with multiple parameters = " + obj3.data);

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q9ConsOverload'
Constructor no parameters = 10
Constructor with single parameters = 30
Constructor with multiple parameters = 50
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

## Program 10

Objective: WAP in java for method overloading.

```
package Cos;

public class Q10MethodOverloading {

    int add(){

        int a = 10;

        int b = 20;

        return a + b;

    }

    int add(int a, int b) {

        return a + b;

    }

    int add(int a, int b, int c) {

        return a + b + c;

    }

    public static void main(String []args){

        Q10MethodOverloading obj1 = new Q10MethodOverloading();

        Q10MethodOverloading obj2 = new Q10MethodOverloading();

        Q10MethodOverloading obj3 = new Q10MethodOverloading();

        System.out.println("Method with no parameter : "+obj1.add());

        System.out.println("Method with two parameter : "+obj2.add(10, 20));

        System.out.println("Method with three parameter : "+obj3.add(12,45,67));

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q10MethodOverloading'
Method with no parameter : 30
Method with two parameter : 30
Method with three parameter : 124
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes>
```

# Program 11

**Objective : WAP in java for method overriding.**

```
package Cos;

class Man{

    void run(){

        System.out.println("Man Running");

    }

}

class Boy extends Man{

    @Override

    void run(){

        System.out.println("Boy Running");

    }

}

class Q11MethodOverriding{

    public static void main(String[] args) {

        Man man = new Man();

        man.run();

        Boy boy = new Boy();

        boy.run();

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> d::; cd 'd:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes'; & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\ddaf7cc28ec67324242c0a7c1cbb6696\redhat.java\jdt_ws\Programmes_cb786ef6\bin' 'Cos.Q11MethodOverriding'
Man Running
Boy Running
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes> █
```

## Program 12

**Objective:** WAP in java to show run time polymorphism (up casting).

```
package Cos;

public class Q12Polymorphism {

    static int add(int a, int b){

        return a + b;

    }

    static double add(double a,double b){

        return a + b;

    }

    public static void main(String []args){

        System.out.println("Addition of two integers : "+ add(14,15));

        System.out.println("Addition of two doubles : "+ add(18.5,17.5));

    }

}
```

## Output

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q12Polymorphism'
Addition of two integers : 29
Addition of two doubles : 36.0
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 13

**Objective:** WAP in java for access specifiers (all four).

```
package Cos;

public class Q13AccessSpecifier {

    public static int publicField = 10;

    private final static int privateField = 20;

    static int defaultField = 30;

    protected final static int protectedField = 40;


    public static void publicMethod() {

        System.out.println("This is a public method");

        System.out.println("Value of public int: " + publicField);

    }


    private static void privateMethod() {

        System.out.println("This is a private method");

        System.out.println("Value of private int: " + privateField);

    }


    static void defaultMethod() {

        System.out.println("This is a default method");

        System.out.println("Value of default int: " + defaultField);

    }


    protected static void protectedMethod() {

        System.out.println("This is a protected method");

        System.out.println("Value of protected int: " + protectedField);

    }

}
```



```
}

public static void main(String[] args) {

    publicMethod();

    privateMethod();

    defaultMethod();

    protectedMethod();

}

}
```

## Output

```
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q13AccessSpecifier'
This is a public method
Value of public int: 10
This is a private method
Value of private int: 20
This is a default method
Value of default int: 30
This is a protected method
Value of protected int: 40
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 14

**Objective:** WAP in java to implement the single dimension array.

```
package Cos;

import java.util.Scanner;

public class Q14SingleDimArray {

    public static void main(String[] args) {

        try (Scanner sc = new Scanner(System.in)) {

            int[] arr = new int[5];

            for(int i = 0; i < 5; i++){

                arr[i] = sc.nextInt();

            }

            for(int i = 0; i < 5; i++){

                System.out.println("Value in array index "+ i + " is "+ arr[i]);

            }

        }

    }

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q14SingleDimArray'
1 2 3 4 5
Value in array index 0 is 1
Value in array index 1 is 2
Value in array index 2 is 3
Value in array index 3 is 4
Value in array index 4 is 5
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 15

**Objective:** WAP in java to copy the elements from one array to another array.

```
package Cos;

import java.util.Scanner;

public class Q15ArrayCopy {

    public static void main(String[] args) {

        try (Scanner sc = new Scanner(System.in)) {

            int[] arr1 = new int[5];

            int[] arr2 = new int[5];

            for(int i = 0; i < 5; i++){

                arr1[i] = sc.nextInt();

            }

            System.arraycopy(arr1, 0, arr2, 0, 5);

            for(int i = 0; i < 5; i++){

                System.out.println("Value in array index "+ i + " is "+ arr2[i]);

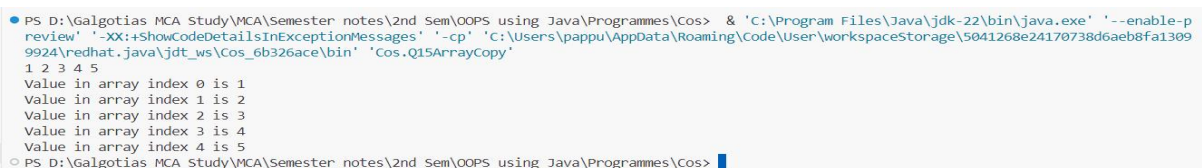
            }

        }

    }

}
```

## Output



```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q15ArrayCopy'
1 2 3 4 5
Value in array index 0 is 1
Value in array index 1 is 2
Value in array index 2 is 3
Value in array index 3 is 4
Value in array index 4 is 5
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 16

**Objective:** WAP in java to perform the addition and multiplication in 2-D array.

```
package Cos;

import java.util.Scanner;

public class Q16AddMulin2D {

    public static void addMatrix(int[][] arr1, int[][] arr2, int[][] arr3) {

        int m = arr1.length;

        int n = arr1[0].length;

        for (int i = 0; i < m; i++) {

            for (int j = 0; j < n; j++) {

                arr3[i][j] = arr1[i][j] + arr2[i][j];

            }

        }

        System.out.println("Resultant matrix after addition:");

        printMatrix(arr3);

    }

    public static void multiplyMatrix(int[][] arr1, int[][] arr2, int[][] arr3) {

        int m = arr1.length;

        int n = arr1[0].length;

        int p = arr2[0].length;

        for (int i = 0; i < m; i++) {

            for (int j = 0; j < p; j++) {

                arr3[i][j] = 0; // Reset value before calculating
```

```

        for (int k = 0; k < n; k++) {
            arr3[i][j] += arr1[i][k] * arr2[k][j];
        }
    }
}
}

```

```

public static void inputMatrix(int[][] matrix, Scanner sc) {
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[i].length; j++) {
            matrix[i][j] = sc.nextInt();
        }
    }
}

```

```

public static void printMatrix(int[][] matrix) {
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[i].length; j++) {
            System.out.print(matrix[i][j] + " ");
        }
        System.out.println();
    }
}

```

```

public static void main(String[] args) {
    try (Scanner sc = new Scanner(System.in)) {
        System.out.println("Enter size of first matrix (m x n): ");
        int m = sc.nextInt();
        int n = sc.nextInt();
    }
}

```

```

System.out.println("Enter size of second matrix (o x p): ");

int o = sc.nextInt();

int p = sc.nextInt();

if (n == o) {

    int[][] arr1 = new int[m][n];

    int[][] arr2 = new int[o][p];

    int[][] arr3 = new int[m][p];

    System.out.println("Enter elements of first matrix:");

    inputMatrix(arr1, sc);

    System.out.println("Enter elements of second matrix:");

    inputMatrix(arr2, sc);

    // Perform addition

    addMatrix(arr1, arr2, arr3);

    // Perform multiplication

    System.out.println("Resultant matrix after multiplication:");

    multiplyMatrix(arr1, arr2, arr3);

} else {

    System.out.println("Matrices are not compatible for multiplication. Exiting...");

}

}

}

}

```

## Output

```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q16AddMulin2D'
Enter size of first matrix (m x n):
3 3
Enter size of second matrix (o x p):
3 3
Enter elements of first matrix:
1 2 3
4 5 6
7 8 9
Enter elements of second matrix:
9 8 7
6 5 4
3 2 1
Resultant matrix after addition:
10 10 10
10 10 10
10 10 10
Resultant matrix after multiplication:
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 17

**Objective:** WAP in java for simple inheritance.

```
package Cos;

// Parent class
class A {
    void display() {
        System.out.println("This is the parent class");
    }
}

// Single Inheritance
class B extends A {
    void show() {
        System.out.println("This is the child class");
    }
}

// Multilevel Inheritance
class C extends B {
    void greet() {
        System.out.println("Hello from the grandchild class");
    }
}

// Hierarchical Inheritance
class D extends C {
    void message() {
        System.out.println("This is another child class");
    }
}
```

```

    }
}

// Main class
public class Q17Inheritance {
    public static void main(String[] args) {
        // Single Inheritance
        B child = new B();
        child.display();
        child.show();

        System.out.println();

        // Multilevel Inheritance
        C grandChild = new C();
        grandChild.display();
        grandChild.show();
        grandChild.greet();

        System.out.println();

        // Hierarchical Inheritance
        D anotherChild = new D();
        anotherChild.display();
        anotherChild.message();
    }
}

```



## Output

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa13099924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q17Inheritance'
○ This is the parent class
  This is the child class

  This is the parent class
  This is the child class
  Hello from the grandchild class

  This is the parent class
  This is another child class
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 18

**Objective:** WAP in java for Final Keyword.

//WAP in java for Final keyword

package Cos;

```
class Q18final {  
    public static void main(String[] args) {  
        final int x = 10;  
        // x = 20; // Error: Cannot assign a value to final variable  
        x  
        System.out.println("Value of x: " + x);  
    }  
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p  
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309  
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q18final'  
Value of x: 10  
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 19

**Objective:** WAP in java for Super Keyword.

```
package Cos;

//WAP in java for super keyword

class Parent {
    Parent() {
        System.out.println("This is Parent class constructor.");
    }
}

class Child extends Parent {
    Child() {
        super(); // Calling Parent class constructor
        System.out.println("This is Child class constructor.");
    }
}

public class Q19Super {
    public static void main(String[] args) {
        Child childObj = new Child();
    }
}
```

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q19Super'
This is Parent class constructor.
This is Child class constructor.
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 20

**Objective:** WAP in java for chaining constructor.

```
package Cos;

//WAP in java for chaining constructor

public class Q20ConstructorChain {
    private String name;
    private int age;

    // Default constructor
    public Q20ConstructorChain() {
        this("Unknown", 0); // Calling parameterized constructor
    }

    // Parameterized constructor
    public Q20ConstructorChain(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void display() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }

    public static void main(String[] args) {
        Q20ConstructorChain person1 = new Q20ConstructorChain(); //
        Calls default constructor

        Q20ConstructorChain person2 = new Q20ConstructorChain("John",
        30); // Calls parameterized constructor
    }
}
```

```
    person1.display();  
    person2.display();  
}  
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p  
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309  
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q20ConstructorChain'  
Name: Unknown  
Age: 0  
Name: John  
Age: 30  
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 21

**Objective:** WAP in java for abstract method and abstract class.

//WAP in java for abstract method and abstract class

```
package Cos;
```

```
// Abstract class
```

```
abstract class Shape {
```

```
    // Abstract method (no implementation)
```

```
    abstract void draw();
```

```
    // Concrete method
```

```
    void display() {
```

```
        System.out.println("Displaying shape...");
```

```
    }
```

```
}
```

```
// Concrete subclass of Shape
```

```
class Circle extends Shape {
```

```
    // Implementing abstract method
```

```
    void draw() {
```

```
        System.out.println("Drawing circle...");
```

```
    }
```

```
}
```

```
// Concrete subclass of Shape
```

```
class Rectangle extends Shape {
```

```
    // Implementing abstract method
```

```
    void draw() {
```

```
        System.out.println("Drawing rectangle...");
```

```

    }
}

public class Q21Abstract {
    public static void main(String[] args) {
        Shape shape1 = new Circle(); // Upcasting
        Shape shape2 = new Rectangle(); // Upcasting

        shape1.draw();
        shape1.display();

        shape2.draw();
        shape2.display();
    }
}

```

## Output

```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q21Abstract'
Drawing circle...
Displaying shape...
Drawing rectangle...
Displaying shape...
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 22

**Objective: WAP in java for interface.**

//WAP in java for interface

package Cos;

// Define an interface

interface Animal {

// Abstract method (does not have a body)

void sound();

// Default method

default void eat() {

System.out.println("This animal eats food.");

}

// Static method

static void sleep() {

System.out.println("This animal sleeps.");

}

}

// Implementing the Animal interface

class Dog implements Animal {

// Implementing sound method

public void sound() {

System.out.println("Dog barks");

}



```
// Overriding the default eat method
public void eat() {
    System.out.println("Dog eats bones.");
}
}

// Implementing the Animal interface
class Cat implements Animal {
    // Implementing sound method
    public void sound() {
        System.out.println("Cat meows");
    }
}

public class Q22Interface {
    public static void main(String[] args) {
        Animal myDog = new Dog(); // Upcasting
        Animal myCat = new Cat(); // Upcasting

        myDog.sound();
        myDog.eat(); // Overridden method
        Animal.sleep(); // Static method

        myCat.sound();
        myCat.eat(); // Default method
        Animal.sleep(); // Static method
    }
}
```

## Output

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q22Interface'
Dog barks
Dog eats bones.
This animal sleeps.
Cat meows
This animal eats food.
This animal sleeps.
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 23

**Objective:** WAP in java for multiple inheritance.

```
package Cos;

// Interface for Animal
interface Animal {
    void eat();
}

// Interface for Bird
interface Bird {
    void fly();
}

// Class implementing multiple interfaces
class Sparrow implements Animal, Bird {
    @Override
    public void eat() {
        System.out.println("Sparrow eats insects.");
    }

    @Override
    public void fly() {
        System.out.println("Sparrow flies high.");
    }
}

public class Q23InterfaceAnimal {
    public static void main(String[] args) {
```

```
Sparrow sparrow = new Sparrow();

// Calling methods

sparrow.eat();

sparrow.fly();

}

}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q23InterfaceAnimal'
Sparrow eats insects.
Sparrow flies high.
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 24

**Objective:** WAP in java for Object Cloning(shallow and deep copy).

```
package Cos;

// Class to be cloned
class Student implements Cloneable {

    String name;

    Address address;

    public Student(String name, Address address) {

        this.name = name;

        this.address = address;

    }

    // Shallow copy
    protected Object clone() throws CloneNotSupportedException {

        return super.clone();

    }

    // Deep copy
    public Student deepClone() {

        Address clonedAddress = new Address(this.address.street, this.address.city);

        return new Student(this.name, clonedAddress);

    }

    @Override
    public String toString() {

        return "Student [name=" + name + ", address=" + address + "];"

    }

}
```

```
}  
}
```

```
// Address class for deep copy
```

```
class Address {  
    String street;  
    String city;  
  
    public Address(String street, String city) {  
        this.street = street;  
        this.city = city;  
    }  
  
    @Override  
    public String toString() {  
        return "Address [street=" + street + ", city=" + city + "];"  
    }  
}
```

```
public class Q24ClassClone {  
    public static void main(String[] args) throws CloneNotSupportedException {  
        Address address = new Address("123 Main St", "City");  
        Student originalStudent = new Student("John", address);  
  
        // Shallow copy  
        Student clonedStudent = (Student) originalStudent.clone();  
        System.out.println("Shallow Copy: ");  
        System.out.println("Original: " + originalStudent);  
    }  
}
```

```

        System.out.println("Cloned: " + clonedStudent);

        // Deep copy
        Student deepClonedStudent = originalStudent.deepClone();

        System.out.println("\nDeep Copy: ");

        System.out.println("Original: " + originalStudent);

        System.out.println("Deep Cloned: " + deepClonedStudent);
    }
}

```

## Output

```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q24ClassClone'
Shallow Copy:
Original: Student [name=John, address=Address [street=123 Main St, city=City]]
Cloned: Student [name=John, address=Address [street=123 Main St, city=City]]

Deep Copy:
Original: Student [name=John, address=Address [street=123 Main St, city=City]]
Deep Cloned: Student [name=John, address=Address [street=123 Main St, city=City]]
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 25

**Objective:** WAP in java for Inner Classes (all types).

```
package Cos;

public class Q25NestedClass {

    // Nested Inner Class (Static Inner Class)
    static class StaticInnerClass {
        void display() {
            System.out.println("Inside Static Inner Class");
        }
    }

    // Non-static Nested Inner Class (Inner Class)
    class InnerClass {
        void display() {
            System.out.println("Inside Non-static Inner Class");
        }
    }

    // Method with Local Inner Class
    void localInnerClassExample() {
        class LocalInner {
            void display() {
                System.out.println("Inside Local Inner Class");
            }
        }

        LocalInner localInner = new LocalInner();

        localInner.display();
    }
}
```



```
}
```

```
// Method with Anonymous Inner Class
```

```
void anonymousInnerClassExample() {  
    Thread t = new Thread(new Runnable() {  
        @Override  
        public void run() {  
            System.out.println("Inside Anonymous Inner Class");  
        }  
    });  
    t.start();  
}
```

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

```
    // Accessing Static Inner Class
```

```
    Q25NestedClass.StaticInnerClass staticInnerObj = new  
Q25NestedClass.StaticInnerClass();  
    staticInnerObj.display();
```

```
    // Accessing Non-static Inner Class
```

```
    Q25NestedClass outerObj = new Q25NestedClass();  
    Q25NestedClass.InnerClass innerObj = outerObj.new InnerClass();  
    innerObj.display();
```

```
    // Accessing Local Inner Class
```

```
    outerObj.localInnerClassExample();
```

```
    // Accessing Anonymous Inner Class
```

```
        outerObj.anonymousInnerClassExample();  
    }  
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p  
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309  
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q25NestedClass'  
Inside Static Inner Class  
Inside Non-static Inner Class  
Inside Local Inner Class  
Inside Anonymous Inner Class  
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 26

**Objective:** WAP in java to create the package (user defined package).

File 1:

```
// Here Package same name as directory is created with class HelloWorld same name as file
// name
// This file is located in directory 'MyPackage'
// This directory is created where the jdk is located, inside the 'bin' directory
package Cos;
```

```
public class Q26PackageF1 {
    public void sayHello() {
        System.out.println("Hello, World!");
    }
}
```

### Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q1printHello'
Hello World
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

File 2:

```
// This file is located in bin directory
// Here package is imported
import Cos.HelloWorld;

public class Q26Package {
    public static void main(String[] args) {
        HelloWorld hello = new HelloWorld();
        hello.sayHello();
    }
}
```

### Output

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  ...  Code  [Running] cd "c:\Program Files\Eclipse Adoptium\jdk-21.0.1.12-hotspot\bin\" && javac
Main.java && java Main
Hello, World!
[Done] exited with code=0 in 1.592 seconds
```

## Program 27

**Objective:** WAP in java for exception handling by using try, catch and finally.

```
package Cos;

public class Q27TryCatch {
    public static void main(String[] args) {
        try {
            // Code that may throw an exception
            int result = divide(10, 0);
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            // Catching specific exception
            System.out.println("Exception caught: Division by zero");
        } finally {
            // Code that will always execute, regardless of whether
            // an exception occurred or not
            System.out.println("Finally block executed");
        }
    }

    public static int divide(int dividend, int divisor) {
        return dividend / divisor;
    }
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q27TryCatch'
Exception caught: Division by zero
Finally block executed
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 28

**Objective:** WAP in java for throw and throws Exception.

```
package Cos;

public class Q27TryCatch {
    public static void main(String[] args) {
        try {
            // Code that may throw an exception
            int result = divide(10, 0);
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            // Catching specific exception
            System.out.println("Exception caught: Division by zero");
        } finally {
            // Code that will always execute, regardless of whether
            // an exception occurred or not
            System.out.println("Finally block executed");
        }
    }

    public static int divide(int dividend, int divisor) {
        return dividend / divisor;
    }
}
```

### Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q28Exception'
Caught exception: Age must be 18 or above.
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 29

**Objective:** WAP in java to throw your own Exceptions.

```
package Cos;

public class Q29CustomException {
    public static void main(String[] args) {
        try {
            // Simulating a condition where a custom exception needs
            // to be thrown

            int balance = 100;
            int amountToWithdraw = 200;
            withdrawMoney(balance, amountToWithdraw);
        } catch (InsufficientBalanceException e) {
            System.out.println("Caught exception: " +
e.getMessage());
        }
    }

    // Method that throws a custom exception if the balance is
    // insufficient

    public static void withdrawMoney(int balance, int amount) throws
InsufficientBalanceException {
        if (balance < amount) {
            throw new InsufficientBalanceException("Insufficient
balance in the account.");
        } else {
            System.out.println("Withdrawal successful. Remaining
balance: " + (balance - amount));
        }
    }
}
```

```
// Custom exception class for insufficient balance
class InsufficientBalanceException extends Exception {
    public InsufficientBalanceException(String message) {
        super(message);
    }
}
```

## Output

```
● PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q29CustomException'
Caught exception: Insufficient balance in the account.
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```



## Program 30

**Objective:** WAP in java to reading and writing in file using byte stream.

```
package Cos;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class Q30FileIO {
    public static void main(String[] args) {
        String fileName = "example.txt";
        String content = "Hello, this is a sample text.";

        // Writing to a file using FileOutputStream
        try (FileOutputStream outputStream = new
FileOutputStream(fileName)) {
            byte[] bytes = content.getBytes();
            outputStream.write(bytes);
            System.out.println("Content written to file
successfully.");
        } catch (IOException e) {
            System.out.println("Error writing to file: " +
e.getMessage());
        }

        // Reading from a file using FileInputStream
        try (FileInputStream inputStream = new
FileInputStream(fileName)) {
            int ch;
            System.out.println("Content read from file:");

```

```

        while ((ch = inputStream.read()) != -1) {
            System.out.print((char) ch);
        }
    } catch (IOException e) {
        System.out.println("Error reading from file: " +
e.getMessage());
    }
}
}
}

```

## Output

```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
● review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q30FileIO'
Content written to file successfully.
Content read from file:
Hello, this is a sample text.
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 31

**Objective:** WAP in java to reading and writing in file using character stream.

```
package Cos;

import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class Q31FileWrite {
    public static void main(String[] args) {
        String fileName = "example.txt";
        String content = "Hello, this is a sample text.";

        // Writing to a file using FileWriter
        try (FileWriter writer = new FileWriter(fileName)) {
            writer.write(content);
            System.out.println("Content written to file successfully.");
        } catch (IOException e) {
            System.out.println("Error writing to file: " + e.getMessage());
        }

        // Reading from a file using FileReader
        try (FileReader reader = new FileReader(fileName)) {
            int ch;
            System.out.println("Content read from file:");
            while ((ch = reader.read()) != -1) {
                System.out.print((char) ch);
            }
        }
    }
}
```

```

    }
} catch (IOException e) {
    System.out.println("Error reading from file: " +
e.getMessage());
}
}
}
}

```

## Output

```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q31Filewrite'
Content written to file successfully.
Content read from file:
Hello, this is a sample text.
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 32

**Objective:** WAP in java to reading and writing through console class.

```
package Cos;

import java.util.Scanner;

public class Q32ConsoleIO {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Reading input from the console
        System.out.print("Enter your name: ");
        String name = scanner.nextLine();

        // Writing output to the console
        System.out.println("Hello, " + name + "! Welcome to the
console I/O example.");

        // Closing the scanner
        scanner.close();
    }
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q32ConsoleIO'
Enter your name: Chandi Charan Mahato
Hello, Chandi Charan Mahato! Welcome to the console I/O example.
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 33

**Objective:** WAP in java how to create thread using Thread Class.

```
package Cos;

public class Q33Thread extends Thread {
    public void run() {
        // Code to be executed by the thread
        for (int i = 0; i < 5; i++) {
            System.out.println("Thread running: " + i);
            try {
                // Sleep for 1 second
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        // Create an instance of Q33
        Q33Thread thread = new Q33Thread();
        // Start the thread
        thread.start();
    }
}
```

## Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
● review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q33Thread'
Thread running: 0
Thread running: 1
Thread running: 2
Thread running: 3
Thread running: 4
○ PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```

## Program 34

**Objective:** WAP in java how to create thread using runnable interface.

```
package Cos;

public class Q34ThreadInstance implements Runnable {
    public void run() {
        // Code to be executed by the thread
        for (int i = 0; i < 5; i++) {
            System.out.println("Thread running: " + i);
            try {
                // Sleep for 1 second
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        // Create an instance of Q34
        Q34ThreadInstance myRunnable = new Q34ThreadInstance();
        // Create a Thread object with myRunnable as the target
        Thread thread = new Thread(myRunnable);
        // Start the thread
        thread.start();
    }
}
```



# Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+Sh
owCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa13099924\redhat.java\jdt_ws\Cos_6b32
6ace\bin' 'Cos.Q34ThreadInstance'
Thread running: 0
Thread running: 1
Thread running: 2
Thread running: 3
Thread running: 4
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> █
```

## Program 35

**Objective:** WAP in java to implement multithreading.

```
package Cos;

// Runnable implementation
class MyRunnable implements Runnable {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println("Thread " +
                Thread.currentThread().getId() + ": " + i);
            try {
                Thread.sleep(1000); // Sleep for 1 second
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class Q35ThreadsWInterface {
    public static void main(String[] args) {
        // Creating and starting threads using Runnable interface
        Thread thread1 = new Thread(new MyRunnable());
        Thread thread2 = new Thread(new MyRunnable());
        thread1.start();
        thread2.start();

        // Creating and starting threads using Thread class
        MyThread myThread1 = new MyThread();
        MyThread myThread2 = new MyThread();
    }
}
```

```

        myThread1.start();

        myThread2.start();

    }

}

// Thread class extension

class MyThread extends Thread {

    public void run() {

        for (int i = 1; i <= 5; i++) {

            System.out.println("Thread " +
Thread.currentThread().getId() + ": " + i);

            try {

                Thread.sleep(1000); // Sleep for 1 second

            } catch (InterruptedException e) {

                e.printStackTrace();

            }

        }

    }

}

}

```

## Output



```

PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-p
review' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa1309
9924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q35ThreadswInterface'
Thread 21: 1
Thread 22: 1
Thread 24: 1
Thread 23: 1
Thread 24: 2
Thread 22: 2
Thread 23: 2
Thread 21: 2
Thread 21: 3
Thread 22: 3
Thread 24: 3
Thread 23: 3
Thread 24: 4
Thread 21: 4
Thread 23: 4
Thread 22: 4
Thread 24: 5
Thread 22: 5
Thread 23: 5
Thread 21: 5
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>

```

## Program 36

**Objective:** WAP in java to achieve synchronization in threads.

```
package Cos;

class Counter {
    private int count = 0;

    // Synchronized method to increment the count
    public synchronized void increment() {
        count++;
    }

    // Method to get the current count
    public int getCount() {
        return count;
    }
}

class MyThread extends Thread {
    private Counter counter;

    public MyThread(Counter counter) {
        this.counter = counter;
    }

    public void run() {
        for (int i = 0; i < 1000; i++) {
            counter.increment();
        }
    }
}
```

```

    }
}

public class Q36synchronizedFunction{
    public static void main(String[] args) {
        Counter counter = new Counter();
        MyThread thread1 = new MyThread(counter);
        MyThread thread2 = new MyThread(counter);

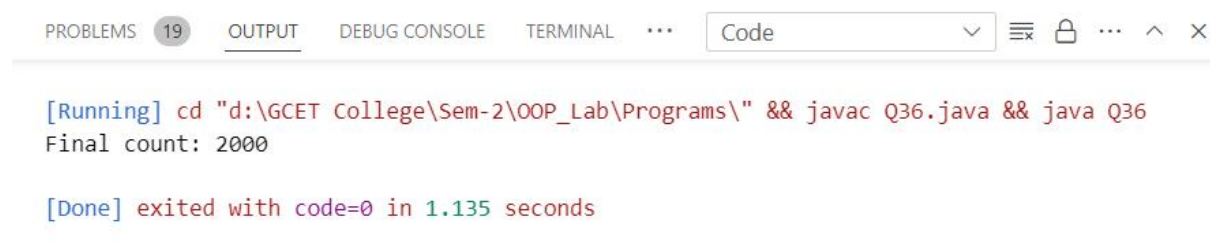
        thread1.start();
        thread2.start();

        try {
            thread1.join();
            thread2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        // The expected value of count should be 2000
        System.out.println("Final count: " + counter.getCount());
    }
}

```

## Output



```

PROBLEMS 19 OUTPUT DEBUG CONSOLE TERMINAL ... Code
[Running] cd "d:\GCET College\Sem-2\OOP_Lab\Programs\" && javac Q36.java && java Q36
Final count: 2000

[Done] exited with code=0 in 1.135 seconds

```

## Program 37

**Objective:** WAP in java to implement the concept of Priorities of threads.

```
package Cos;

class MyThread extends Thread {
    public MyThread(String name) {
        super(name);
    }

    public void run() {
        System.out.println("Thread: " + getName() + " Priority: " +
        getPriority());
    }
}

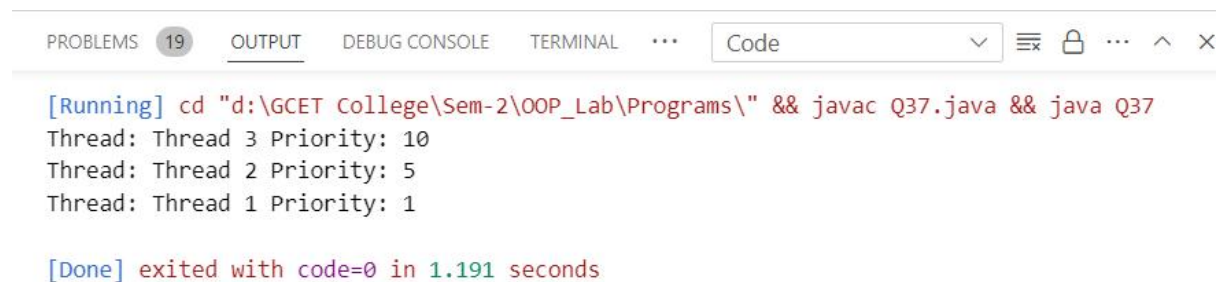
public class Q37DifferThread {
    public static void main(String[] args) {
        // Create three threads with different priorities
        MyThread thread1 = new MyThread("Thread 1");
        MyThread thread2 = new MyThread("Thread 2");
        MyThread thread3 = new MyThread("Thread 3");

        // Set priorities for threads
        thread1.setPriority(Thread.MIN_PRIORITY);
        thread2.setPriority(Thread.NORM_PRIORITY);
        thread3.setPriority(Thread.MAX_PRIORITY);

        // Start the threads
    }
}
```

```
        thread1.start();  
        thread2.start();  
        thread3.start();  
    }  
}
```

## Output



The screenshot shows an IDE's output window with the following content:

```
PROBLEMS 19 OUTPUT DEBUG CONSOLE TERMINAL ... Code  
[Running] cd "d:\GCET College\Sem-2\OOP_Lab\Programs\" && javac Q37.java && java Q37  
Thread: Thread 3 Priority: 10  
Thread: Thread 2 Priority: 5  
Thread: Thread 1 Priority: 1  
  
[Done] exited with code=0 in 1.191 seconds
```

## Program 38

**Objective:** WAP in java to illustrate the concept of Generic Programming.

```
// A generic class representing a generic box that can hold any type of object
class Box<T> {
    private T item;

    public void setItem(T item) {
        this.item = item;
    }

    public T getItem() {
        return item;
    }

    public void displayItemType() {
        System.out.println("Type of item in the box: " + item.getClass().getName());
    }
}

public class Q38 {
    public static void main(String[] args) {
        // Create a Box to hold an integer
        Box<Integer> intBox = new Box<>();
        intBox.setItem(123);
        System.out.println("Item in the integer box: " + intBox.getItem());
        intBox.displayItemType();

        // Create a Box to hold a string
        Box<String> stringBox = new Box<>();
        stringBox.setItem("Hello, Generics!");
        System.out.println("Item in the string box: " + stringBox.getItem());
        stringBox.displayItemType();
    }
}
```

### Output

```
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\pappu\AppData\Roaming\Code\User\workspaceStorage\5041268e24170738d6aeb8fa13099924\redhat.java\jdt_ws\Cos_6b326ace\bin' 'Cos.Q38GenericBox'
Item in the integer box: 123
Type of item in the box: java.lang.Integer
Item in the string box: Hello, Generics!
Type of item in the box: java.lang.String
PS D:\Galgotias MCA Study\MCA\Semester notes\2nd Sem\OOPS using Java\Programmes\Cos>
```



## Program 39

**Objective:** WAP in java to illustrate the concept of event handling (using various event handlers).

```
import java.awt.*;
import java.awt.event.*;

public class Q39 extends Frame implements ActionListener, MouseListener, KeyListener {
    private TextField textField;
    private Label label;

    public Q39() {
        // Create components
        label = new Label("Click inside the window and press any key:");
        textField = new TextField(20);

        // Add components to the frame
        add(label, BorderLayout.NORTH);
        add(textField, BorderLayout.SOUTH);

        // Add event listeners
        addMouseListener(this);
        addKeyListener(this);

        // Set frame properties
        setTitle("Event Handling Example");
        setSize(300, 200);
        setVisible(true);
    }

    // ActionListener event handler
    public void actionPerformed(ActionEvent e) {
        textField.setText("Button clicked!");
    }

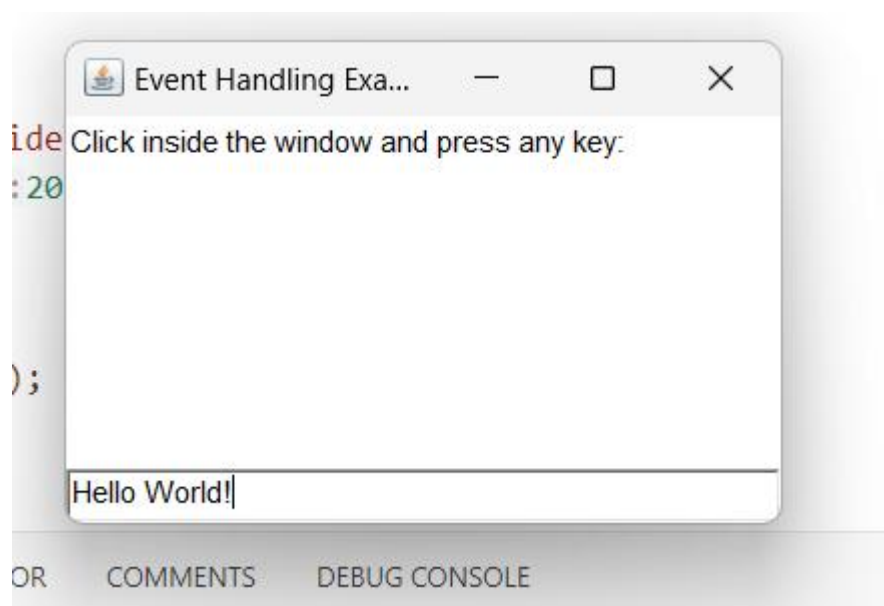
    // MouseListener event handlers
    public void mouseClicked(MouseEvent e) {
        textField.setText("Mouse clicked at (" + e.getX() + ", " + e.getY() + ")");
    }
    public void mouseEntered(MouseEvent e) {}
    public void mouseExited(MouseEvent e) {}
    public void mousePressed(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}

    // KeyListener event handlers
    public void keyPressed(KeyEvent e) {
```

```
        textField.setText("Key pressed: " + e.getKeyChar());
    }
    public void keyReleased(KeyEvent e) {}
    public void keyTyped(KeyEvent e) {}

    public static void main(String[] args) {
        new Q39();
    }
}
```

## Output



## Program 40

**Objective:** Create a simple registration application using various swing components (like: JFrame, JButton, JLabel, text fields, text areas, check box and radio buttons).

```
package Cos;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class RegistrationApp extends JFrame implements ActionListener {

    private JTextField firstNameField, lastNameField;

    private JRadioButton maleRadioButton, femaleRadioButton;

    private JTextArea addressArea;

    private JCheckBox termsCheckBox;

    private JButton registerButton;

    public RegistrationApp() {

        // Set frame properties

        setTitle("Registration Application");

        setSize(400, 300);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new BorderLayout());

        // Create components

        JPanel formPanel = new JPanel(new GridLayout(6, 2));

        JLabel firstNameLabel = new JLabel("First Name:");

        firstNameField = new JTextField();

        JLabel lastNameLabel = new JLabel("Last Name:");
```

```
lastNameField = new JTextField();

JLabel genderLabel = new JLabel("Gender:");

maleRadioButton = new JRadioButton("Male");

femaleRadioButton = new JRadioButton("Female");

ButtonGroup genderGroup = new ButtonGroup();

genderGroup.add(maleRadioButton);

genderGroup.add(femaleRadioButton);

JLabel addressLabel = new JLabel("Address:");

addressArea = new JTextArea(4, 20);

JScrollPane scrollPane = new JScrollPane(addressArea);

JLabel termsLabel = new JLabel("Accept Terms and Conditions:");

termsCheckBox = new JCheckBox();

registerButton = new JButton("Register");


// Add components to the form panel

formPanel.add(firstNameLabel);

formPanel.add(firstNameField);

formPanel.add(lastNameLabel);

formPanel.add(lastNameField);

formPanel.add(genderLabel);

formPanel.add(maleRadioButton);

formPanel.add(new JLabel()); // Empty space

formPanel.add(femaleRadioButton);

formPanel.add(addressLabel);

formPanel.add(scrollPane);

formPanel.add(termsLabel);

formPanel.add(termsCheckBox);
```

```

// Add register button action listener
registerButton.addActionListener(this);

// Add components to the frame
add(formPanel, BorderLayout.CENTER);
add(registerButton, BorderLayout.SOUTH);

// Set frame visible
setVisible(true);
}

public void actionPerformed(ActionEvent e) {
    if (e.getSource() == registerButton) {
        // Perform registration process

        String firstName = firstNameField.getText();
        String lastName = lastNameField.getText();
        String gender = maleRadioButton.isSelected() ? "Male" : "Female";
        String address = addressArea.getText();
        boolean acceptedTerms = termsCheckBox.isSelected();

        // Display registration information
        String message = "Registration Successful!\n\n";
        message += "First Name: " + firstName + "\n";
        message += "Last Name: " + lastName + "\n";
        message += "Gender: " + gender + "\n";
        message += "Address: " + address + "\n";
        message += "Terms and Conditions Accepted: " + (acceptedTerms ? "Yes" : "No");
    }
}

```

```

        JOptionPane.showMessageDialog(this, message, "Registration Info",
JOptionPane.INFORMATION_MESSAGE);

    }

}

public static void main(String[] args) {

    new RegistrationApp();

}

}

```

## Output

The screenshot shows a Java Swing window titled "Registration Application". The window contains a registration form with the following elements:

- First Name:** A text input field.
- Last Name:** A text input field.
- Gender:** Two radio buttons labeled "Male" and "Female".
- Address:** A text area with a scroll bar.
- Accept Terms and Conditions:** A checkbox.
- Register:** A button at the bottom of the form.

The background of the image shows snippets of Java code, including the `JOptionPane.showMessageDialog` call and the `main` method.

