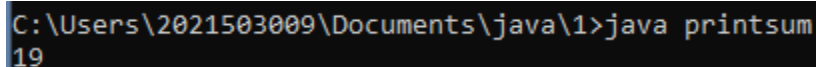


Aim:

To implement addition of two numbers using class, object and static methods in java.

1) Class:**Program:**

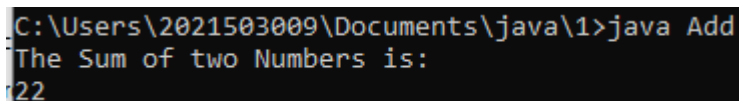
```
class printsum
{
    public static void main(String args[])
    {
        int x=12;
        int y=7;
        int sum=x+y;
        System.out.println(sum);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\1>java printsum
19
```

2) Static:**Program:**

```
class Add{
static void AddNumbers(int n,int m){
int res=0;
res=n+m;
System.out.println(res);
}
public static void main(String args[]){
System.out.println("The Sum of two Numbers is:");
AddNumbers(10,12);
}
}
```

Output:

```
C:\Users\2021503009\Documents\java\1>java Add
The Sum of two Numbers is:
22
```

3) Object:**Program:**

```
class BasAdd{
public static void main(String args[]){
Addition object= new Addition();
System.out.println("The Addition of two numbers= ");
}
```

```
object.Add();  
}  
}  
class Addition{  
void Add(){  
int a=10,b=20;  
int res=a+b;  
System.out.println(res);  
}  
}
```

Output:

```
C:\Users\2021503009\Documents\java\1>java BasAdd  
The Addition of two numbers=  
30
```

Result:

Thus, the java program has been implemented using class, objects and static methods for addition of two numbers.

Aim:

To perform addition of two numbers by getting different ways of input from user in java.

1) Run time:**Program:**

```
import java.io.*;
public class add1{
    public static void main(String inputval[]){
        int a,b,c;
        a = Integer.parseInt(inputval[0]);
        b = Integer.parseInt(inputval[1]);
        c = a+b;
        System.out.println(c);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\2>java add1 1 2
3
```

2) Using Scanner:**Program:**

```
import java.util.Scanner;
public class assignment{
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
        System.out.println("Enter a number: ");
        int num1 = S.nextInt();
        System.out.println(num1+=2);
    }
}
```

Output:

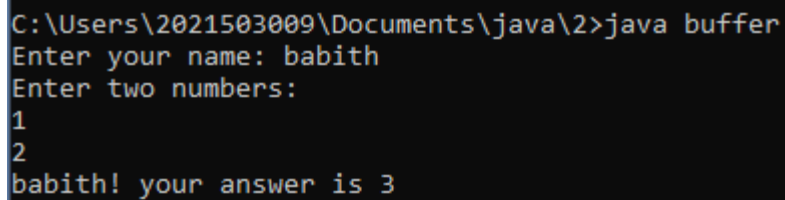
```
C:\Users\2021503009\Documents\java\2>java assignment
Enter a number:
20
22
```

3) Using BufferedReader:

Program:

```
import java.io.*;
class buffer{
    public static void main(String inputval[]) throws IOException{
        InputStreamReader ir = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(ir);
        System.out.print("Enter your name: ");
        String name = br.readLine();
        System.out.println("Enter two numbers: ");
        int num1 = Integer.parseInt(br.readLine());
        int num2 = Integer.parseInt(br.readLine());
        int sum = num1 + num2;
        System.out.println(name + "! your answer is " + sum);
    }
}
```

Output:



```
C:\Users\2021503009\Documents\java\2>java buffer
Enter your name: babith
Enter two numbers:
1
2
babith! your answer is 3
```

Result:

Thus, the java program to implement addition of two number by getting different ways of input from the user has been done.

EXP.NO: 03

DATE: 07.08.2023

IMPLEMENTATION OF DIFFERENT TYPES OF OPERATORS, FINAL VARIABLES AND STATIC & NON- STATIC VARIABLES IN JAVA

Aim:

To implement different types of operators, final variables and static and non static variables in java.

1) Basic Assignment operations:

Program:

```
import java.util.Scanner;
public class assignment{
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num1 = S.nextInt();
        System.out.print("+ = : ");
        System.out.println(num1+=2);
        System.out.print("- = : ");
        System.out.println(num1-=2);
        System.out.print("* = : ");
        System.out.println(num1*=2);
        System.out.print("/ = : ");
        System.out.println(num1/=2);
        System.out.print("% = : ");
        System.out.println(num1%=2);
        System.out.print("& = : ");
        System.out.println(num1&=2);
        System.out.print("| = : ");
        System.out.println(num1|=2);
        System.out.print("<= : ");
        System.out.println(num1<=2);
        System.out.print(">= : ");
        System.out.println(num1>=2);
        System.out.print("^ = : ");
        System.out.println(num1^=2);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\3>java assignment
Enter a number: 20
+= : 22
-= : 20
*= : 40
/= : 20
%= : 0
&= : 0
|= : 2
<<= : 8
>>= : 2
^= : 0
```

2) Ternary Operator:

Program:

```
import java.util.Scanner;
public class ternary{
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
        System.out.println("Enter two number: ");
        int num1 = S.nextInt();
        int num2 = S.nextInt();
        System.out.println("Enter your command(1-add,2-sub): ");
        int comm = S.nextInt();
        int add = num1 + num2;
        int sub = num1 - num2;
        int result = (comm == 1) ? add : sub;
        System.out.print("Result: " + result);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\3>java ternary
Enter two number:
20
10
Enter your command(1-add,2-sub):
1
Result: 30
```

3) Final Variable:

Program:

```
import java.util.Scanner;
class finalVar
{
    public static void main(String args[])
    {
        Scanner inp=new Scanner(System.in);
        final double pi=3.14;
        double rad,res;
        System.out.print("Enter the radius:");
```

```

        rad=inp.nextDouble();
        res=pi*rad*rad;
        System.out.println("Area of circle:"+ res);
    }
}

```

Output:

```

D:\2021503011\OPERATORS>javac finalVar.java

D:\2021503011\OPERATORS>java finalVar
Enter the radius:3
Area of circle:28.259999999999998

```

4) Bitwise operators:

Program:

```

import java.util.Scanner;
public class bitwise{
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
        System.out.print("Enter first number: ");
        int num1 = S.nextInt();
        System.out.print("Enter second number: ");
        int num2 = S.nextInt();
        System.out.print("Bitwise AND: ");
        System.out.println(num1 & num2);
        System.out.print("Bitwise OR: ");
        System.out.println(num1 | num2);
        System.out.print("Bitwise NOT: ");
        System.out.print(~num1);

    }
}

```

Output:

```

C:\Users\2021503009\Documents\java\3>java bitwise
Enter first number: 20
Enter second number: 12
Bitwise AND: 4
Bitwise OR: 28
Bitwise NOT: -21

```

Result:

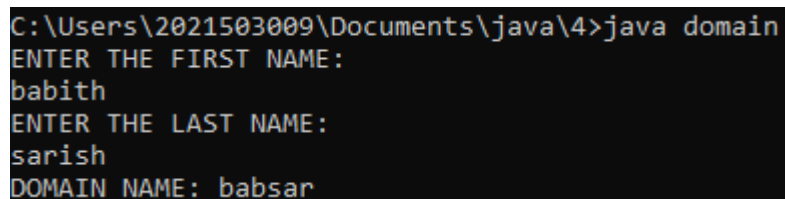
Thus, the java program to implement different types of operators, final variables and static and non-static variables has been done.

Aim:

To implement various string functions in java.

1) Find the domain from the first three letters of first and last name:**Program:**

```
import java.io.*;
import java.util.Scanner;
class domain
{
    public static void main(String args[])
    {
        Scanner s = new Scanner(System.in);
        System.out.println("ENTER THE FIRST NAME: ");
        String firstname = s.nextLine();
        System.out.println("ENTER THE LAST NAME: ");
        String lastname = s.nextLine();
        String domain="";
        int i=0,flag=0;
        for(i=0;i!=firstname.length() && flag!=3;i++)
        {
            domain+=firstname.charAt(i);
            flag++;
        }
        flag=0;
        for(i=0;i!=lastname.length() && flag!=3;i++)
        {
            domain+=lastname.charAt(i);
            flag++;
        }
        System.out.println("DOMAIN NAME: "+domain);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\4>java domain
ENTER THE FIRST NAME:
babith
ENTER THE LAST NAME:
sarish
DOMAIN NAME: babsar
```

2) To check if the mail id is valid or not:**Program:**

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.Scanner;

public class mailid {
```



```

public static void main(String[] args) {
    System.out.println("Enter your mail id: ");
    Scanner S = new Scanner(System.in);
    String email = S.nextLine();

    if (mailfun(email)) {
        System.out.println("Valid email address");
    } else {
        System.out.println("Invalid email address");
    }
}

public static boolean mailfun(String email) {
    String regex = "^[A-Za-z0-9+_.-]+@(.+)[A-Za-z]$";

    Pattern pat = Pattern.compile(regex);
    Matcher mat = pat.matcher(email);

    return mat.matches();
}

```

Output:

```

C:\Users\2021503009\Documents\java\4>java mailid
Enter your mail id:
babith@gmail.com
Valid email address

```

Result:

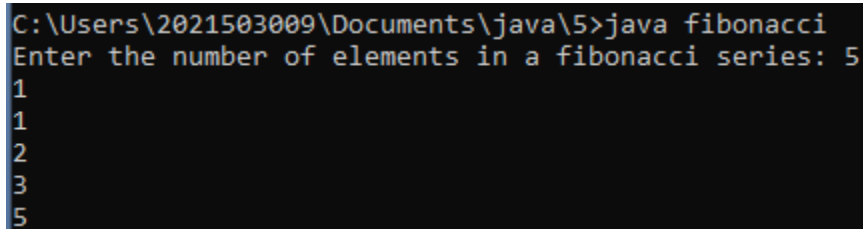
Thus, the java program to implement string programs has been done.

Aim:

To implement different types of loops in java.

1) Fibonacci Series using while loop:**Program:**

```
import java.util.*;
import java.util.Scanner;
public class fibonacci{
    public static void main(String args[]){
        int a=1, b=1,c,n,i=2;
        Scanner S = new Scanner(System.in);
        System.out.print("Enter the number of elements in a fibonacci series: ");
        n = S.nextInt();
        System.out.print(a+"\n"+b+"\n");
        do{
            c=a+b;
            a=b;
            b=c;
            i++;
            System.out.print(c+"\n");
        }while(i<n);
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\5>java fibonacci
Enter the number of elements in a fibonacci series: 5
1
1
2
3
5
```

2) Check if the given number is Armstrong or not:**Program:**

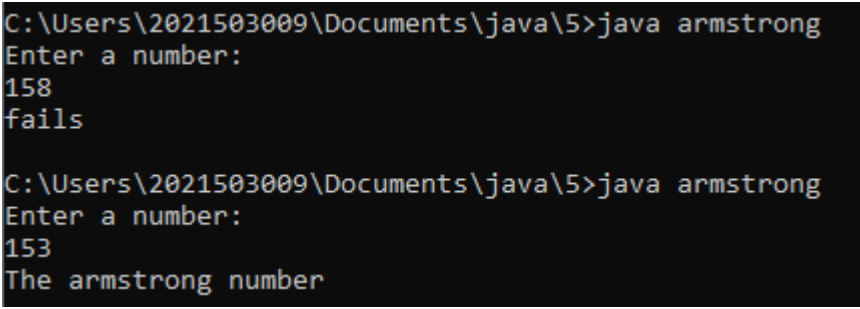
```
import java.util.*;
import java.util.Scanner;
import java.lang.Math;
class armstrong{
    public static int len(String num){
        int i=0;
        char[] l= num.toCharArray();
        for(char it:l){
            i++;
        }
        return i;
    }
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
```

```

        System.out.println("Enter a number: ");
        String num = S.nextLine();
        int length = len(num);
        int res=0,i=0;
        char[] arr= num.toCharArray();
        do{
            char ch = arr[i];
            int number = ch-48;
            res += Math.pow(number,length);
            i++;
        }while(i<length);
        String num2 = String.valueOf(res);
        if(num.equals(num2))
            System.out.println("The armstrong number");
        else
            System.out.println("fails");
    }
}

```

Output:



```

C:\Users\2021503009\Documents\java\5>java armstrong
Enter a number:
158
fails

C:\Users\2021503009\Documents\java\5>java armstrong
Enter a number:
153
The armstrong number

```

3) Check if the given string is Palindrome or not:

Program:

```

import java.util.*;
class palindrome{
    public static void main(String[] args){
        int n, n2, sum=0;
        System.out.print("Enter a number: ");
        Scanner S = new Scanner(System.in);
        n = S.nextInt();
        n2 = n;
        do{
            int temp = n2%10;
            sum *= 10;
            sum += temp;
            n2 /= 10;
        }while(n2>0);
        if(n==sum)
            System.out.print("Palindrome");
        else
            System.out.print("fails");
    }
}

```

Output:

```
C:\Users\2021503009\Documents\java\5>java palindrome
Enter a number: 101
Palindrome
C:\Users\2021503009\Documents\java\5>java palindrome
Enter a number: 112
fails
```

4) Check whether the given number is odd or even:

Program:

```
import java.util.*;
class oddeven{
    public static void main(String[] args){
        int n;
        System.out.print("Enter a number: ");
        Scanner S = new Scanner(System.in);
        n = S.nextInt();
        while(n>=0){
            if(n==1){
                System.out.print("Odd");
                break;
            }
            if(n==0){
                System.out.print("Even");
                break;
            }
            n-=2;
        }
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\5>java oddeven
Enter a number: 10
Even
C:\Users\2021503009\Documents\java\5>java oddeven
Enter a number: 11
Odd
```

5) User Choice:

Program:

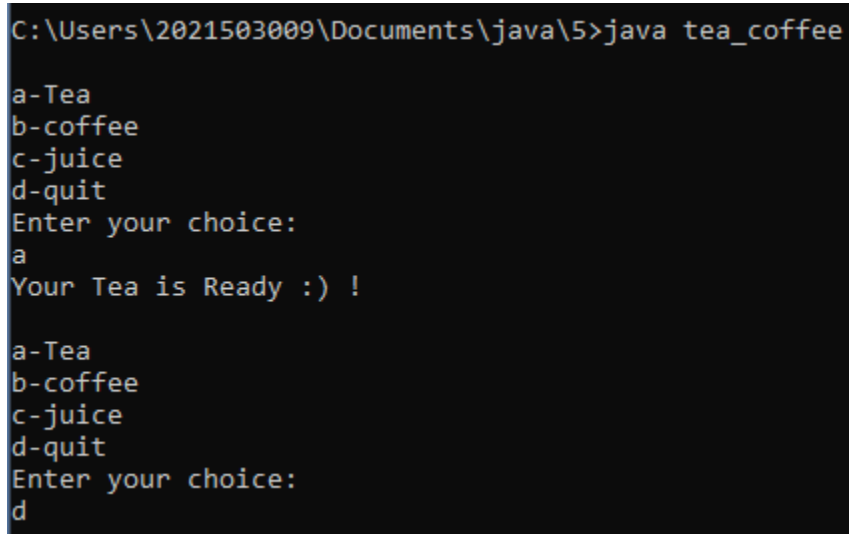
```
import java.util.*;
import java.util.Scanner;
class tea_coffee{
    public static void main(String args[]){
        Scanner S = new Scanner(System.in);
        char sol;
        int flag=1;
        do{
            flag=0;
            System.out.println("\na-Tea\nb-coffee\nc-juice\nd-quit");
```

```

        System.out.println("Enter your choice: ");
        String num = S.nextLine();
        switch(num){
            case "a":
                System.out.println("Your Tea is Ready :) !");
                flag=1;
                break;
            case "b":
                System.out.println("Your Coffee is Ready :) !");
                flag=1;
                break;
            case "c":
                System.out.println("Your Juice is Ready :) !");
                flag=1;
                break;
            case "d":
                System.exit(0);
        }
    }while(flag==1);
}

```

Output:



```

C:\Users\2021503009\Documents\java\5>java tea_coffee
a-Tea
b-coffee
c-juice
d-quit
Enter your choice:
a
Your Tea is Ready :) !

a-Tea
b-coffee
c-juice
d-quit
Enter your choice:
d

```

Result:

Thus, implementation of different types of loops in java has been done.

Aim:

To implement arrays in java.

1) Perform Matrix Addition using 2D array:**Program:**

```
import java.util.Scanner;

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

        // Input for the dimensions of the matrices
        System.out.print("Enter the number of rows for matrix 1: ");
        int rows1 = scanner.nextInt();
        System.out.print("Enter the number of columns for matrix 1: ");
        int cols1 = scanner.nextInt();
        System.out.print("Enter the number of rows for matrix 2: ");
        int rows2 = scanner.nextInt();
        System.out.print("Enter the number of columns for matrix 2: ");
        int cols2 = scanner.nextInt();

        if (cols1 != rows2) {
            System.out.println("Matrix dimensions are not compatible for multiplication.");
            return;
        }

        // Input for matrix 1
        int[][] matrix1 = new int[rows1][cols1];
        System.out.println("Enter elements for matrix 1:");
        for (int i = 0; i < rows1; i++) {
            for (int j = 0; j < cols1; j++) {
                matrix1[i][j] = scanner.nextInt();
            }
        }

        // Input for matrix 2
        int[][] matrix2 = new int[rows2][cols2];
        System.out.println("Enter elements for matrix 2:");
        for (int i = 0; i < rows2; i++) {
            for (int j = 0; j < cols2; j++) {
                matrix2[i][j] = scanner.nextInt();
            }
        }

        // Create the result matrix
        int[][] result = new int[rows1][cols2];

        // Perform matrix multiplication
        for (int i = 0; i < rows1; i++) {
            for (int j = 0; j < cols2; j++) {
```

```

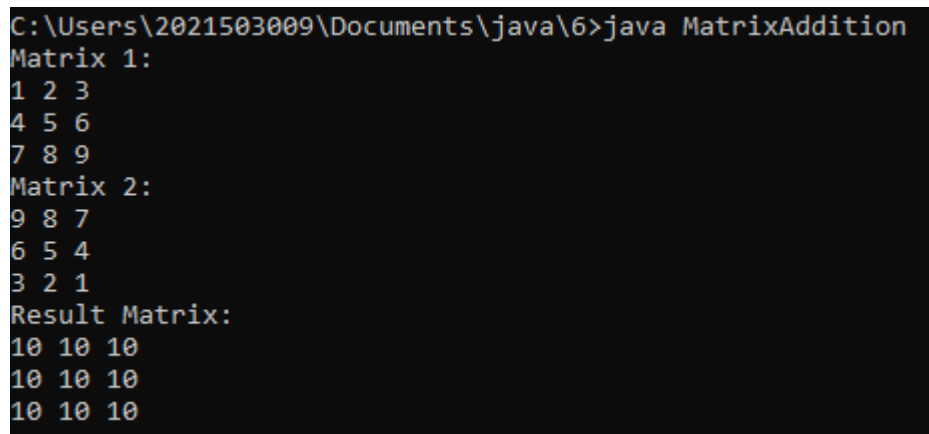
        for (int k = 0; k < cols1; k++) {
            result[i][j] += matrix1[i][k] * matrix2[k][j];
        }
    }
}

// Print the result matrix
System.out.println("Result matrix:");
for (int i = 0; i < rows1; i++) {
    for (int j = 0; j < cols2; j++) {
        System.out.print(result[i][j] + " ");
    }
    System.out.println();
}

scanner.close();
}
}

```

Output:



```

C:\Users\2021503009\Documents\java\6>java MatrixAddition
Matrix 1:
1 2 3
4 5 6
7 8 9
Matrix 2:
9 8 7
6 5 4
3 2 1
Result Matrix:
10 10 10
10 10 10
10 10 10

```

2) Perform Sorting using 1D array:

Program:

```

import java.util.Scanner;
public class BubbleSort {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter element #" + (i + 1) + ": ");
            arr[i] = scanner.nextInt();
        }
        bubbleSortarr(arr);
        System.out.println("Sorted Array:");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}

```

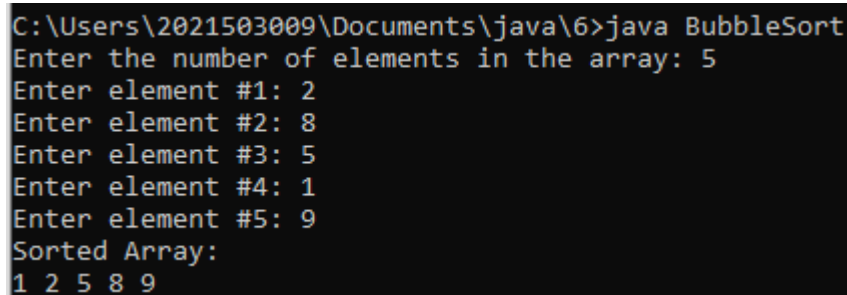
```

        scanner.close();
    }

    public static void bubbleSortarr(int[] arr) {
        int n = arr.length;
        boolean swapped;
        for (int i = 0; i < n - 1; i++) {
            swapped = false;
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                    swapped = true;
                }
            }
            if (!swapped) {
                break;
            }
        }
    }
}

```

Output:



```

C:\Users\2021503009\Documents\java\6>java BubbleSort
Enter the number of elements in the array: 5
Enter element #1: 2
Enter element #2: 8
Enter element #3: 5
Enter element #4: 1
Enter element #5: 9
Sorted Array:
1 2 5 8 9

```

3) Matrix Multiplication:

Program:

```

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

        // Input for the dimensions of the matrices
        System.out.print("Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Input for matrix 1
        int[][] matrix1 = new int[rows][cols];
        System.out.println("Enter elements for matrix 1:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix1[i][j] = scanner.nextInt();
            }
        }
    }
}

```



```

    }
}

// Input for matrix 2
int[][] matrix2 = new int[rows][cols];
System.out.println("Enter elements for matrix 2:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        matrix2[i][j] = scanner.nextInt();
    }
}

// Create the result matrix
int[][] result = new int[rows][cols];

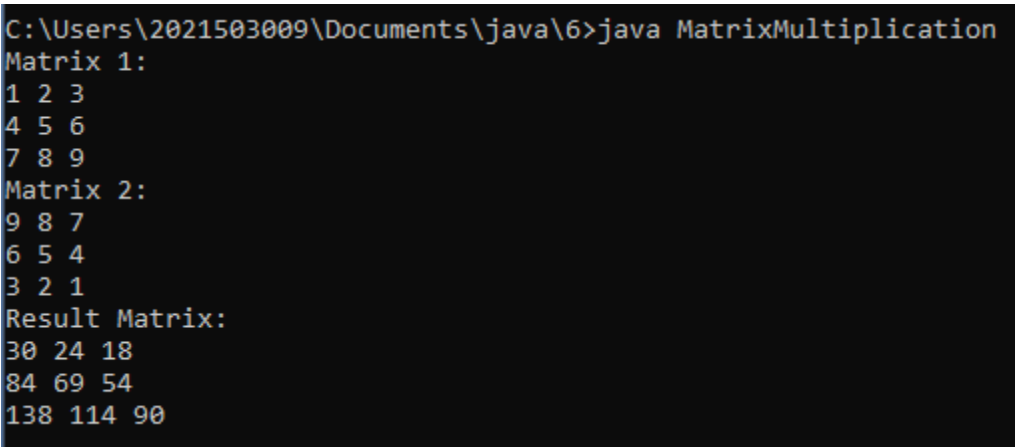
// Perform matrix addition
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        result[i][j] = matrix1[i][j] + matrix2[i][j];
    }
}

// Print the result matrix
System.out.println("Result matrix:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        System.out.print(result[i][j] + " ");
    }
    System.out.println();
}

scanner.close();
}
}

```

Output:



```

C:\Users\2021503009\Documents\java\6>java MatrixMultiplication
Matrix 1:
1 2 3
4 5 6
7 8 9
Matrix 2:
9 8 7
6 5 4
3 2 1
Result Matrix:
30 24 18
84 69 54
138 114 90

```

Result:

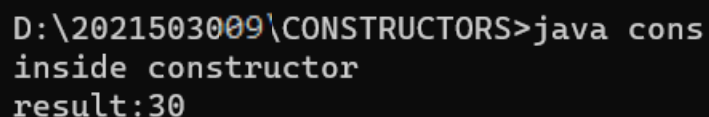
Thus, arrays in java have been implemented.

Aim:

To implement different types of constructors in java.

1) Simple constructor:**Program:**

```
public class cons {  
    int x, y,z;  
    cons()  
    {  
        x=10;  
        y=20;  
        System.out.println("inside constructor");  
    }  
    void add()  
    {  
        z=x+y;  
        System.out.println("result:"+z);  
    }  
  
    public static void main(String[] args) {  
        cons c= new cons();  
        c.add();  
    }  
}
```

Output:

```
D:\2021503009\CONSTRUCTORS>java cons  
inside constructor  
result:30
```

2) Default constructor**Program:**

```
class Default_constructor {  
    String name;  
    int age;  
    public Default_constructor() {  
        name = "John Doe"; // Default name  
        age = 30;          // Default age  
    }  
    public void displayInfo() {  
        System.out.println("Name: " + name);  
        System.out.println("Age: " + age);  
    }  
}  
  
public class DefaultConstructor {
```

```

    public static void main(String[] args) {
        Default_constructor person = new Default_constructor();
        person.displayInfo();
    }
}

```

Output:

```

C:\Users\2021503009\Documents\java\7>java Default_constructor
0 null

```

3) Parameterized constructor

Program:

```

import java.io.*;
class Parenthesis_constructor {
    class Test {
        int id;
        String name;

        Test(int num, String str) {
            id = num;
            name = str;
        }

        void display() {
            System.out.println(id + " " + name);
        }
    }
    public static void main(String[] args) {
        Parenthesis_constructor hello = new Parenthesis_constructor();
        Test testObj = hello.new Test(1001, "ABC");
        testObj.display();
    }
}

```

Output:

```

C:\Users\2021503009\Documents\java\7>java Parenthesis_constructor
1001 ABC

```

4) Constructor overloading

Program:

```

class Student {
    String name;
    int age;

    // Constructor with no parameters (default constructor)
    public Student() {
        name = "Unknown";
        age = 0;
    }
}

```

```

// Constructor with one parameter (name)
public Student(String studentName) {
    name = studentName;
    age = 0; // Default age
}

// Constructor with two parameters (name and age)
public Student(String studentName, int studentAge) {
    name = studentName;
    age = studentAge;
}

// Method to display student information
public void displayInfo() {
    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
}
}

public class ConstructorOverloadingExample {
    public static void main(String[] args) {
        Student student1 = new Student();           // Default constructor
        Student student2 = new Student("Alice");     // Constructor with name
        Student student3 = new Student("Bob", 21);   // Constructor with name and age

        student1.displayInfo();
        student2.displayInfo();
        student3.displayInfo();
    }
}

```

Output:

```

C:\Users\2021503009\Documents\java\7>java Person
Person 1: John Doe, Age: 30
Person 2: Alice, Age: 30
Person 3: Bob, Age: 25

```

Result:

Thus, implementation of different types of constructors in java has been done.

Aim:

To implement different types of inheritance in java.

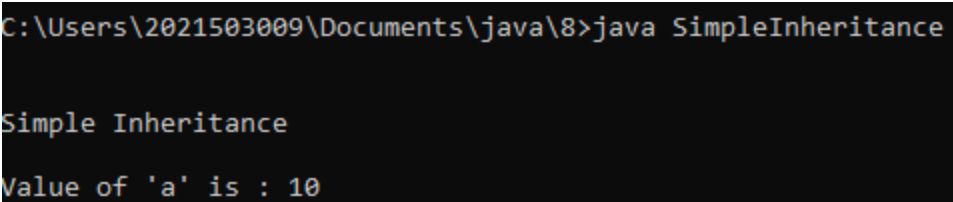
1) Single level Inheritance:**Program:**

```
// Superclass (base class)
class Animal {
    void eat() {
        System.out.println("The animal eats food.");
    }
}

// Subclass (derived class)
class Dog extends Animal {
    void bark() {
        System.out.println("The dog barks.");
    }
}

public class SimpleInheritanceExample {
    public static void main(String[] args) {
        // Create an object of the Dog class
        Dog dog = new Dog();

        // Call methods from the superclass and subclass
        dog.eat(); // Inherited from Animal
        dog.bark(); // Specific to Dog
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\8>java SimpleInheritance
Simple Inheritance
Value of 'a' is : 10
```

2) Hierarchical Inheritance:**Program:**

```
// Parent class (base class)
class Animal {
    void eat() {
        System.out.println("The animal eats food.");
    }
}

// Child class 1
class Dog extends Animal {
```

```

        void bark() {
            System.out.println("The dog barks.");
        }
    }

    // Child class 2
    class Cat extends Animal {
        void meow() {
            System.out.println("The cat meows.");
        }
    }

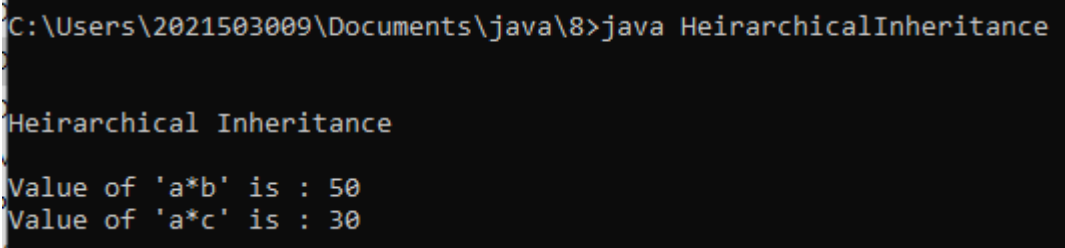
    public class HierarchicalInheritanceExample {
        public static void main(String[] args) {
            // Create objects of child classes
            Dog dog = new Dog();
            Cat cat = new Cat();

            // Call methods from the parent class
            dog.eat();
            cat.eat();

            // Call methods specific to each child class
            dog.bark();
            cat.meow();
        }
    }
}

```

Output:



```

C:\Users\2021503009\Documents\java\8>java HeirarchicalInheritance
Heirarchical Inheritance
Value of 'a*b' is : 50
Value of 'a*c' is : 30

```

3) Multilevel Inheritance:

Program:

```

// Grandparent class
class Animal {
    void eat() {
        System.out.println("The animal eats food.");
    }
}

// Parent class (inherits from Animal)
class Mammal extends Animal {
    void sleep() {
        System.out.println("The mammal sleeps.");
    }
}

```

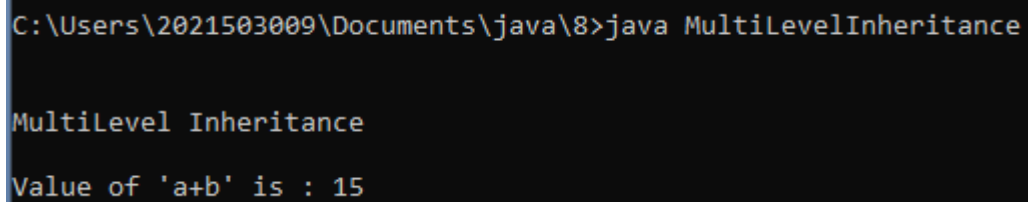
```
// Child class (inherits from Mammal)
class Dog extends Mammal {
    void bark() {
        System.out.println("The dog barks.");
    }
}

public class MultilevelInheritanceExample {
    public static void main(String[] args) {
        // Create an object of the child class Dog
        Dog dog = new Dog();

        // Call methods from the parent and grandparent classes
        dog.eat(); // Inherited from Animal
        dog.sleep(); // Inherited from Mammal

        // Call method specific to the Dog class
        dog.bark();
    }
}
```

Output:



```
C:\Users\2021503009\Documents\java\8>java MultiLevelInheritance

MultiLevel Inheritance
Value of 'a+b' is : 15
```

Result:

Thus, different types of Inheritance have been implemented in java.

Aim:

To implement calculator using package and to implement multiple inheritance using interface in java.

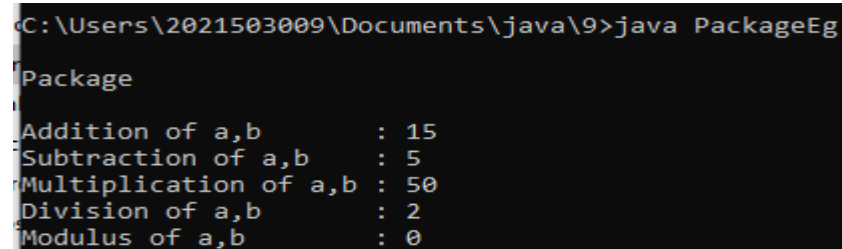
1) Basic calculator using package:**Program:****Package:**

```
package pack;
```

```
public class Calculator{
    public int add(int a,int b){
        return a+b;
    }
    public int sub(int a,int b){
        return a-b;
    }
    public int mul(int a,int b){
        return a*b;
    }
    public int div(int a,int b){
        return a/b;
    }
    public int mod(int a,int b){
        return a%b;
    }
}
```

Main function:

```
import pack.Calculator;
class PackageEg{
    public static void main(String args[]){
        System.out.println("\nPackage\n");
        int a=10,b=5;
        Calculator C=new Calculator();
        System.out.println("Addition of a,b    : "+C.add(a,b));
        System.out.println("Subtraction of a,b  : "+C.sub(a,b));
        System.out.println("Multiplication of a,b : "+C.mul(a,b));
        System.out.println("Division of a,b    : "+C.div(a,b));
        System.out.println("Modulus of a,b     : "+C.mod(a,b));
    }
}
```

Output:

```
C:\Users\2021503009\Documents\java\9>java PackageEg
Package
Addition of a,b    : 15
Subtraction of a,b  : 5
Multiplication of a,b : 50
Division of a,b    : 2
Modulus of a,b     : 0
```


2) Multiple Inheritance using Interface:

Program:

```
interface CNG_Car{
    void drive();
    void CNG_Kit();
}
interface Petrol_Car{
    void drive();
    void Petrol_Kit();
}
class Hybrid_Car implements CNG_Car, Petrol_Car{
    public void drive(){
        System.out.println("Driving a Hybrid Car");
    }
    public void Petrol_Kit(){
        System.out.println("Using the Petrol_Kit for Hybrid Car");
    }
    public void CNG_Kit(){
        System.out.println("Using the CNG_Kit for Hybrid Car");
    }
}
class MultipleInheritance{
    public static void main(String[] args){
        Hybrid_Car obj = new Hybrid_Car();
        obj.drive();
        obj.CNG_Kit();
        obj.Petrol_Kit();
    }
}
```

Output:

```
D:\2021503009\INHERITANCE>java MultipleInheritance
Driving a Hybrid Car
Using the CNG_Kit for Hybrid Car
Using the Petrol_Kit for Hybrid Car
```

Result:

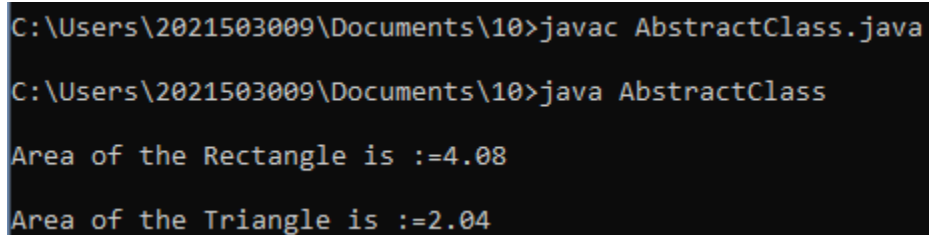
Thus, package and interface has been implemented successfully.

Aim:

To implement Abstract class & method, Super keyword, Access specifiers and Instance() in java.

1) Abstract class:**Program:**

```
abstract class Shape {  
    abstract void area(double b, double h);  
}  
class Triangle extends Shape {  
    void area(double base, double height){  
        double area=0.5* base*height;  
        System.out.println("\nArea of the Triangle is :="+area);  
    }  
}  
class Rectangle extends Shape {  
    void area(double length, double breadth){  
        double area=length*breadth;  
        System.out.println("\nArea of the Rectangle is :="+area);  
    }  
}  
class AbstractClass{  
    public static void main(String[] args){  
        Rectangle r =new Rectangle();  
        r.area(1.2,3.4);  
        Triangle t =new Triangle();  
        t.area(1.2,3.4);  
    }  
}
```

Output:

```
C:\Users\2021503009\Documents\10>javac AbstractClass.java  
C:\Users\2021503009\Documents\10>java AbstractClass  
Area of the Rectangle is :=4.08  
Area of the Triangle is :=2.04
```

2) Abstract method:**Program:**

```
abstract class Shape {  
    double dim1;  
    double dim2;  
    Shape(double a, double b) {  
        dim1 = a;
```

```

        dim2 = b;
    }
    abstract double area();
}
class Rectangle extends Shape {
    Rectangle(double a, double b) {
        super(a, b);
    }

    double area() {
        System.out.println("Inside Area for Rectangle.");
        return dim1 * dim2;
    }
}
class Triangle extends Shape {
    Triangle(double a, double b) {
        super(a, b);
    }

    double area() {
        System.out.println("Inside Area for Triangle.");
        return dim1 * dim2 / 2;
    }
}
class Abstractmethod {
    public static void main(String args[]) {
        Rectangle r = new Rectangle(9, 5);
        Triangle t = new Triangle(10, 8);
        Shape shaperef;
        shaperef = r;
        System.out.println("Area is " + shaperef.area());
        shaperef = t;
        System.out.println("Area is " + shaperef.area());
    }
}

```

Output:

```

C:\Users\2021503009\Documents\java\10>java AbstractClassEg
Inside Area for Rectangle.
Area is 45.0
Inside Area for Triangle.
Area is 40.0

```

3) Super keyword:

Program:

```

class SuperKeyword{
    public static void main(String args[]){
        child c=new child();
    }
}

```

```

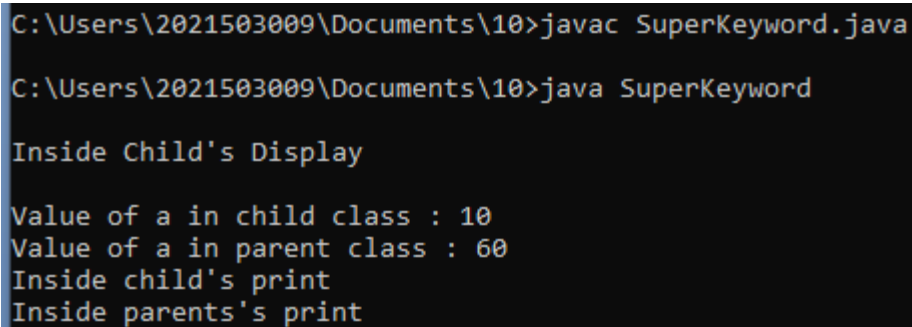
        c.display();
    }
}
class parent {
    int a=60;
    void print(){
        System.out.println("Inside parents's print ");
    }
}
class child extends parent{
    int a=10;
    void print(){
        System.out.println("Inside child's print ");
    }

    void display(){
        System.out.println("\nInside Child's Display ");
        System.out.println("\nValue of a in child class : "+a);

        System.out.println("Value of a in parent class : "+super.a);
        print();
        super.print();
    }
}

```

Output:



```

C:\Users\2021503009\Documents\10>javac SuperKeyword.java

C:\Users\2021503009\Documents\10>java SuperKeyword

Inside Child's Display

Value of a in child class : 10
Value of a in parent class : 60
Inside child's print
Inside parents's print

```

4) Access specifiers:

Program:

```

class parent{
    public int a=10;
    private int b=20;
    protected int c=30;
    int d=40; //default access modifier
    int get(){
        return b;
    }
}
class AccessModifiers{
    public static void main(String args[]){
        parent p=new parent();
        System.out.println("Accessing the default variable : "+p.d);
        System.out.println("Accessing the public variable : "+p.a);
    }
}

```

```

        System.out.println("Accessing the protected variable : "+p.c);

        System.out.print("Accessing the private variable using getter method in parent
class: "+p.get());

    }
}

```

Output:

```

C:\Users\2021503009\Documents\10>javac AccessModifiers.java

C:\Users\2021503009\Documents\10>java AccessModifiers
Accessing the default variable : 40
Accessing the public variable : 10
Accessing the protected variable : 30
Accessing the private variable using getter method in parent class: 20
C:\Users\2021503009\Documents\10>javac InstanceOf.java

```

5) Instanceof:

Program:

```

class Parent{
}
class InstanceOf extends Parent{
    public static void main(String args[]){
        InstanceOf i=new InstanceOf();
        System.out.println("\ni instanceof InstanceOf : " +(i instanceof InstanceOf) );
        InstanceOf j=null ;
        System.out.println("\nj instanceof InstanceOf : " +(j instanceof InstanceOf) );
        try{
            InstanceOf k=(InstanceOf) new Parent();
            System.out.println(k instanceof InstanceOf );
        }catch(Exception e){}
        Parent p1=new InstanceOf ();
        System.out.println("\np1 instanceof InstanceOf : " +(p1 instanceof InstanceOf)
);
    }
}

```

Output:

```

C:\Users\2021503009\Documents\10>java InstanceOf

i instanceof InstanceOf : true

j instanceof InstanceOf : false

p1 instanceof InstanceOf : true

```

Result:

Thus, Abstract class & method, Super keyword, Access specifiers and Instance () in java has been implemented successfully.

Aim:

To design an employee registration form using GUI in java.

Program:

```
import java.awt.*;
import java.awt.event.*;
class EmployeeRegistrationForm extends Frame {
    Label lblTitle, lblName, lblSupervisor, lblAge, lblGender, lblDepartment, lblExperience,
    lblAddress;
    TextField txtName, txtSupervisor, txtAge, txtExperience;
    TextArea txtAddress;
    Checkbox checkMale, checkFemale;
    CheckboxGroup cbg;
    Choice department;
    Button btnSave, btnClear;
    public EmployeeRegistrationForm() {
        super("Employee Registration Form");
        setSize(800, 600);
        setLayout(null);
        setVisible(true);
        Color formColor = new Color(240, 240, 240);
        setBackground(formColor);

        Font titleFont = new Font("Arial", Font.BOLD, 25);
        Font labelFont = new Font("Arial", Font.PLAIN, 18);
        Font textFont = new Font("Arial", Font.PLAIN, 15);

        lblTitle = new Label("Employee Registration Form");
        lblTitle.setBounds(200, 40, 400, 50);
        lblTitle.setFont(titleFont);
        lblTitle.setForeground(Color.BLUE);
        add(lblTitle);

        lblName = new Label("Full Name");
        lblName.setBounds(100, 120, 200, 30);
        lblName.setFont(labelFont);
        add(lblName);

        txtName = new TextField();
        txtName.setBounds(300, 120, 400, 30);
        txtName.setFont(textFont);
        add(txtName);

        lblSupervisor = new Label("Supervisor");
        lblSupervisor.setBounds(100, 170, 200, 30);
        lblSupervisor.setFont(labelFont);
        add(lblSupervisor);

        txtSupervisor = new TextField();
        txtSupervisor.setBounds(300, 170, 400, 30);
        txtSupervisor.setFont(textFont);
```

```

add(txtSupervisor);

lblAge = new Label("Age");
lblAge.setBounds(100, 220, 200, 30);
lblAge.setFont(labelFont);
add(lblAge);

txtAge = new TextField();
txtAge.setBounds(300, 220, 400, 30);
txtAge.setFont(textFont);
add(txtAge);

lblGender = new Label("Gender");
lblGender.setBounds(100, 270, 200, 30);
lblGender.setFont(labelFont);
add(lblGender);

cbg = new CheckboxGroup();

checkMale = new Checkbox("Male", cbg, true);
checkMale.setBounds(300, 270, 100, 30);
checkMale.setFont(labelFont);
add(checkMale);

checkFemale = new Checkbox("Female", cbg, false);
checkFemale.setBounds(400, 270, 100, 30);
checkFemale.setFont(labelFont);
add(checkFemale);

lblDepartment = new Label("Department");
lblDepartment.setBounds(100, 320, 200, 30);
lblDepartment.setFont(labelFont);
add(lblDepartment);

department = new Choice();
department.setFont(labelFont);
department.setBounds(300, 320, 400, 50);
department.add("HR");
department.add("Finance");
department.add("IT");
department.add("Marketing");
department.add("Operations");
add(department);

lblExperience = new Label("Years of Experience");
lblExperience.setBounds(100, 370, 200, 30);
lblExperience.setFont(labelFont);
add(lblExperience);

txtExperience = new TextField();
txtExperience.setBounds(300, 370, 400, 30);
txtExperience.setFont(textFont);
add(txtExperience);

lblAddress = new Label("Address");
lblAddress.setBounds(100, 420, 200, 30);
lblAddress.setFont(labelFont);

```

```

add(lblAddress);
txtAddress = new TextArea();
txtAddress.setBounds(300, 410, 350, 100);
txtAddress.setFont(textFont);
add(txtAddress);
btnSave = new Button("Register Employee");
btnSave.setBounds(250, 550, 200, 30);
btnSave.setFont(labelFont);
btnSave.setBackground(new Color(30, 144, 255));
btnSave.setForeground(Color.WHITE);
add(btnSave);
btnClear = new Button("Clear All");
btnClear.setBounds(500, 550, 200, 30);
btnClear.setFont(labelFont);
btnClear.setBackground(new Color(255, 69, 0));
btnClear.setForeground(Color.WHITE);
add(btnClear);

// Close Button Code
this.addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        System.exit(0);
    }
});
}
}

public class EmployeeApp {
    public static void main(String[] args) {
        EmployeeRegistrationForm empForm = new EmployeeRegistrationForm();
    }
}

```

Output:



The screenshot shows a Java Swing window titled "Employee Registration Form". The window has a light gray background and a blue title bar. The form contains the following fields and controls:

- Full Name:** A text input field.
- Supervisor:** A text input field.
- Age:** A text input field.
- Gender:** Two radio buttons labeled "Male" and "Female".
- Department:** A dropdown menu with "HR" selected.
- Years of Experience:** A text input field.
- Address:** A text area with a vertical scrollbar.
- Buttons:** Two buttons at the bottom: "Register Employee" (blue) and "Clear All" (orange).

Result:

Thus, an employee registration form has been designed successfully.

Aim:

To implement a database connection using java.

Program:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Student Registration</title>
</head>
<body>

    <!-- JSP Scriptlet to Load MySQL JDBC Driver -->
    <%
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            out.println("<p>Error loading MySQL JDBC driver: " + e.getMessage() +
"</p>");
        }
    %>

    <h2>Student Registration</h2>

    <!-- Form for Data Insertion -->
    <form action="form.jsp" method="post">
        Enter Registration Number: <input type="text" name="regno"><br>
        Enter Name: <input type="text" name="name"><br>
        <input type="submit" value="Register Student">
    </form>

    <!-- JSP Scriptlet to Process Form Data and Insert into Database -->
    <%
        String regNo = request.getParameter("regno");
        String studentName = request.getParameter("name");

        if (regNo != null && studentName != null && !regNo.isEmpty() &&
!studentName.isEmpty()) {
            try {
                // Establish the database connection
                String jdbcUrl = "jdbc:mysql://localhost:3306/db";
                String dbUser = "root";
                String dbPassword = "root";
```

```

        Connection connection = DriverManager.getConnection(jdbcUrl, dbUser,
dbPassword);

        // Insert data into the database
        String insertQuery = "INSERT INTO data (regno, name) VALUES (?, ?)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(insertQuery)) {
            preparedStatement.setString(1, regNo);
            preparedStatement.setString(2, studentName);
            preparedStatement.executeUpdate();
        }

        // Close the database connection
        connection.close();

        // Display success message
        out.println("<p>Student registered successfully.</p>");

    } catch (SQLException e) {
        out.println("<p>Error: " + e.getMessage() + "</p>");
    }
}
%>

<!-- View Data from Database -->
<h3>View Students</h3>
<%
    try {
        // Establish the database connection
        String jdbcUrl = "jdbc:mysql://localhost:3306/db";
        String dbUser = "root";
        String dbPassword = "root";
        Connection connection = DriverManager.getConnection(jdbcUrl, dbUser,
dbPassword);

        // Retrieve and display data from the database
        String selectQuery = "SELECT * FROM data";
        try (Statement statement = connection.createStatement());
            ResultSet resultSet = statement.executeQuery(selectQuery)) {

            out.println("<table border='1'>");
            out.println("<tr><th>Registration Number</th><th>Name</th></tr>");

            while (resultSet.next()) {
                String regNoFromDB = resultSet.getString("regno");
                String nameFromDB = resultSet.getString("name");

```

```

        out.println("<tr><td>" + regNoFromDB + "</td><td>" + nameFromDB
+ "</td></tr>");
    }

    out.println("</table>");
}

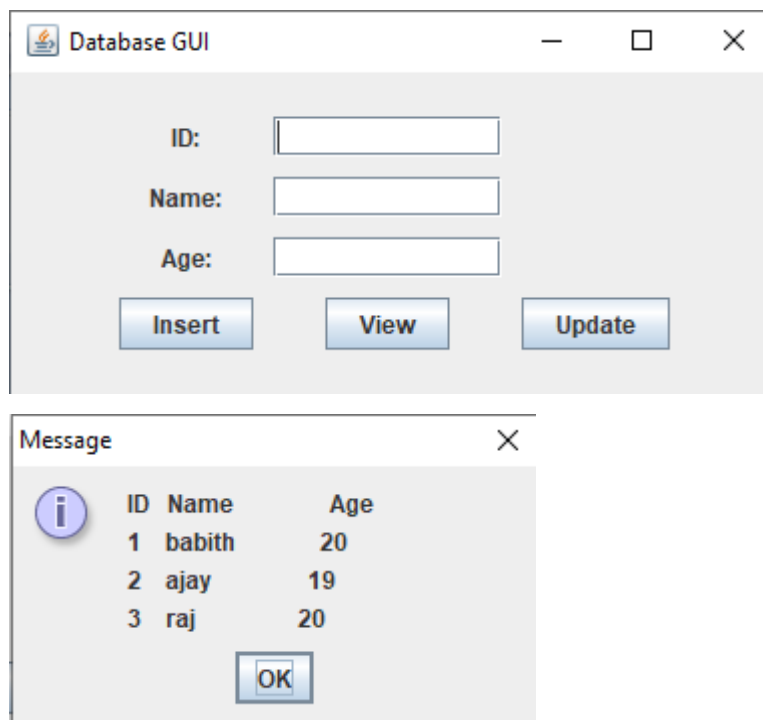
// Close the database connection
connection.close();

} catch (SQLException e) {
    out.println("<p>Error: " + e.getMessage() + "</p>");
}
%>

</body>
</html>

```

Output:



Result:

Thus, database connectivity had been established using java.

Aim:

To implement a basic JSP program to add two numbers in java.

Program:

```
<!-- sum.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Sum Calculator</title>
</head>
<body>
    <h2>Sum Calculator</h2>
    <form action="sum.jsp" method="post">
        Enter Number 1: <input type="text" name="num1"><br>
        Enter Number 2: <input type="text" name="num2"><br>
        <input type="submit" value="Calculate Sum">
    </form>

    <%-- JSP Scriptlet to process form data and display the sum --%>
    <%
        // Retrieve values from the form
        String num1Str = request.getParameter("num1");
        String num2Str = request.getParameter("num2");

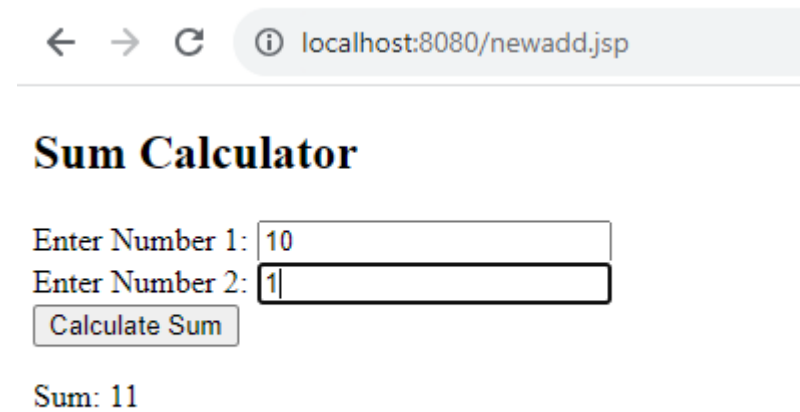
        // Check if both inputs are provided
        if (num1Str != null && num2Str != null && !num1Str.isEmpty() && !num2Str.isEmpty()) {
            try {
                // Parse input strings to integers
                int num1 = Integer.parseInt(num1Str);
                int num2 = Integer.parseInt(num2Str);

                // Calculate the sum
                int sum = num1 + num2;

                // Display the result
                out.println("<p>Sum: " + sum + "</p>");
            } catch (NumberFormatException e) {
                // Handle the case where the input is not a valid number
                out.println("<p>Please enter valid numbers.</p>");
            }
        }
    %>

</body>
</html>
```

Output:



← → ↻ ⓘ localhost:8080/newadd.jsp

Sum Calculator

Enter Number 1: 10

Enter Number 2: 1

Calculate Sum

Sum: 11

Result:

Thus, the JSP program to add two numbers has been implemented successfully.

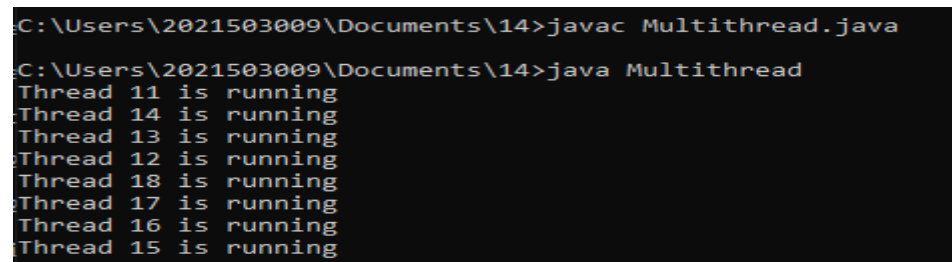
Aim:

To implement multithreading program with synchronization and without synchronization in java.

1) With Threads:**Program:**

```
class MultithreadingRunnable implements Runnable {
    public void run() {
        try {
            // Displaying the thread that is running
            System.out.println("Thread " + Thread.currentThread().getId() + " is running");
        } catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}

class Multithread {
    public static void main(String[] args) {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            Thread t = new Thread(new MultithreadingRunnable()); // Fix the class name here
            t.start();
        }
    }
}
```

Output:

```
C:\Users\2021503009\Documents\14>javac Multithread.java
C:\Users\2021503009\Documents\14>java Multithread
Thread 11 is running
Thread 14 is running
Thread 13 is running
Thread 12 is running
Thread 18 is running
Thread 17 is running
Thread 16 is running
Thread 15 is running
```

2) With Synchronization:**Program:**

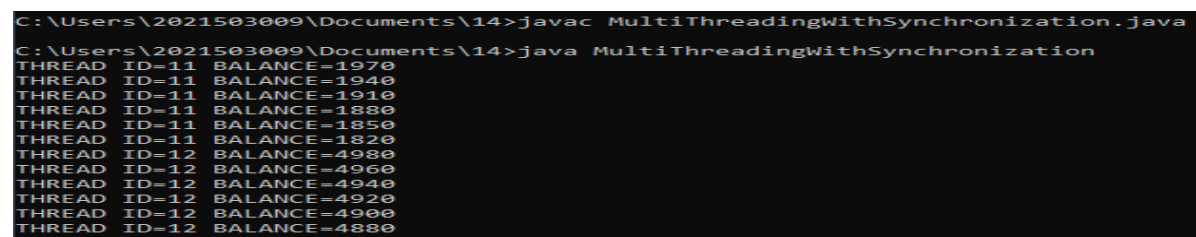
```
class Account {
    int balance;
    synchronized void balInquiry(int bal, int wd) {
        balance = bal - wd;
        try {
            Thread.sleep(400);
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

```

        System.out.println("THREAD ID=" + Thread.currentThread().getId() + " BALANCE=" +
balance);
    }
}
class MyThread1 extends Thread {
    Account a;
    int balance;
    MyThread1(Account a, int bal) {
        this.a = a;
        this.balance = bal;
    }
    public void run() {
        for (int i = 0; i <= 5; i++) {
            a.balInquiry(balance, 30);
            balance = a.balance;
        }
    }
}
class MyThread2 extends Thread {
    Account a;
    int balance;
    MyThread2(Account a, int bal) {
        this.a = a;
        this.balance = bal;
    }
    public void run() {
        for (int i = 0; i <= 5; i++) {
            a.balInquiry(balance, 20);
            balance = a.balance;
        }
    }
}
public class MultiThreadingWithSynchronization {
    public static void main(String args[]) {
        Account a = new Account();
        MyThread1 t1 = new MyThread1(a, 2000);
        MyThread2 t2 = new MyThread2(a, 5000);
        t1.start();
        t2.start();
    }
}

```

Output:



```

C:\Users\2021503009\Documents\14>javac MultiThreadingWithSynchronization.java
C:\Users\2021503009\Documents\14>java MultiThreadingWithSynchronization
THREAD ID=11 BALANCE=1970
THREAD ID=11 BALANCE=1940
THREAD ID=11 BALANCE=1910
THREAD ID=11 BALANCE=1880
THREAD ID=11 BALANCE=1850
THREAD ID=11 BALANCE=1820
THREAD ID=12 BALANCE=4980
THREAD ID=12 BALANCE=4960
THREAD ID=12 BALANCE=4940
THREAD ID=12 BALANCE=4920
THREAD ID=12 BALANCE=4900
THREAD ID=12 BALANCE=4880

```

Result:

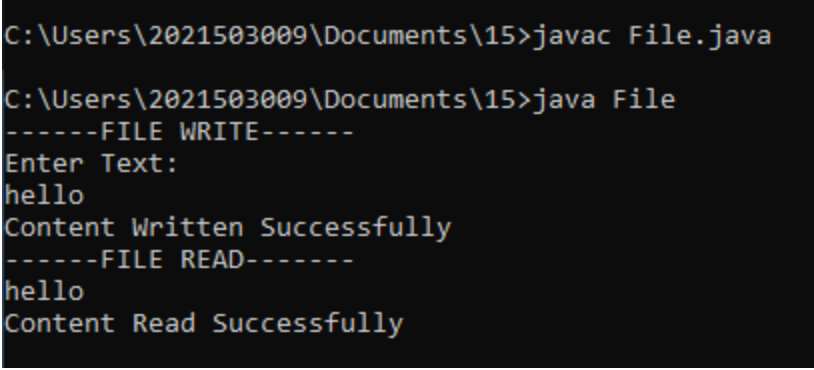
Thus, implementation of multithreading program with synchronization and without synchronization in java has been done.

Aim:

To implement a file reading and writing program in java.

Program:

```
import java.io.*;
class File {
    public static void main(String args[]) {
        try {
            BufferedReader inp = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("-----FILE WRITE-----");
            System.out.println("Enter Text:");
            String text = inp.readLine();
            BufferedWriter writer = new BufferedWriter(new FileWriter("out.txt"));
            writer.write(text);
            writer.close();
            System.out.println("Content Written Successfully");
            System.out.println("-----FILE READ-----");
            BufferedReader reader = new BufferedReader(new FileReader("out.txt"));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
            System.out.println("Content Read Successfully");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output:

```
C:\Users\2021503009\Documents\15>javac File.java

C:\Users\2021503009\Documents\15>java File
-----FILE WRITE-----
Enter Text:
hello
Content Written Successfully
-----FILE READ-----
hello
Content Read Successfully
```

Result:

Thus, the file reading and writing has been implemented in java.

Aim:

To implement a chat application program in java.

Program:**Server:**

```
import java.io.*;
import java.net.*;
import java.util.Scanner;
public class Server{
    public static void main(String[] args){
        try{
            ServerSocket ss = new ServerSocket(8000);
            Socket s=ss.accept();
            System.out.println("Client Connected");
            DataInputStream FromClient = new DataInputStream(s.getInputStream());
            DataOutputStream ToClient = new DataOutputStream(s.getOutputStream());
            Scanner in = new Scanner(System.in);
            while(true){
                String msg="";
                msg=FromClient.readUTF();
                System.out.println("From Client: " + msg);
                System.out.print("To Client: ");
                msg=in.nextLine();
                ToClient.writeUTF(msg);
            }
        }catch(IOException e){
            e.printStackTrace();
        }
    }
}
```

Client:

```
import java.io.*;
import java.net.*;
import java.util.Scanner;
public class Client{
    public static void main(String[] args){
        try{
            Socket s = new Socket("localhost",8000);
            DataInputStream FromServer = new DataInputStream(s.getInputStream());
            DataOutputStream ToServer = new DataOutputStream(s.getOutputStream());
            Scanner in = new Scanner(System.in);
            while(true){
                String msg="";
                System.out.print("To Server: ");
                msg=in.nextLine();
                ToServer.writeUTF(msg);
                msg=FromServer.readUTF();
            }
        }
    }
}
```

```
System.out.println("From Server: " + msg);  
}  
}  
catch(IOException e){  
    e.printStackTrace();  
}  
}  
}
```

Output:

```
C:\Users\2021503009\Documents\16>javac Server.java  
  
C:\Users\2021503009\Documents\16>java Server  
Client Connected  
From Client: hi  
To Client: hello]
```

```
C:\Users\2021503009\Documents\16>javac Client.java  
  
C:\Users\2021503009\Documents\16>java Client  
To Server: hi  
From Server: hello]
```

Result:

Thus, implementation of a chat application program in java has been done.