/\*

**Program-1**: Write a Java RMI program to accept an integer and check whether it is a Prime number or not.

**Reg No**: 24251115

**Date**: 23/10/2025

\*/

**Interface:**

import java.rmi.RemoteException;

import java.rmi.Remote;

public interface PrimeCheckInterface extends *Remote* {

    public *boolean* checkPrime(*int* *number*) throws RemoteException;

}

**Implement:**

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class PrimeCheckImplement extends UnicastRemoteObject implements  PrimeCheckInterface  {

    public PrimeCheckImplement() throws RemoteException{

        super();

    }

    public *boolean* checkPrime(*int* *number*) throws RemoteException{

        if (*number* <= 1) {

            return false;

        }

        for (*int* i = 2; i <= Math.sqrt(*number*); i++) {

            if (*number* % i == 0) {

                return false;

            }

        }

        return true;

    }

}

**Client:**

import java.rmi.registry.\*;

import java.util.Scanner;

public class PrimeCheckClient {

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

        try{

            Registry reg = LocateRegistry.getRegistry(4000);

            PrimeCheckInterface imp = (PrimeCheckInterface) reg.lookup("PrimeCheck");

            Scanner scanner = new Scanner(System.in);

            System.out.println("Enter a number to check if it's prime: ");

*int* number = scanner.nextInt();

            if(imp.checkPrime(number)){

                System.out.println(number + " is a prime number.");

            } else {

                System.out.println(number + " is not a prime number.");

            }

            scanner.close();

        }catch(Exception e){

            System.out.println("Unexpected error: "+e);

         }

    }

}

**Server:**

import java.rmi.registry.Registry;

import java.rmi.registry.LocateRegistry;

public class PrimeCheckServer{

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

        try{

        PrimeCheckImplement obj = new PrimeCheckImplement();

        Registry registry = LocateRegistry.createRegistry(4000);

        registry.bind("PrimeCheck", obj);

        System.out.println("PrimeCheck Server is ready.");

    }catch(Exception e){

        System.out.println("Unexpected error: "+e);

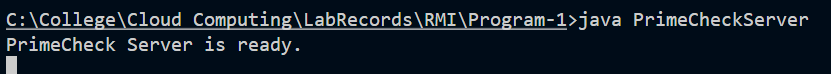
    }

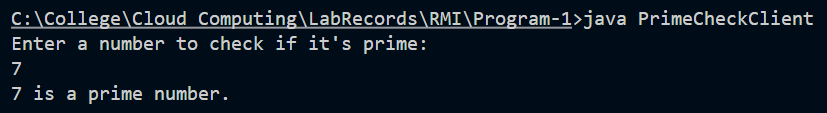
}

}

Output:







/\*

**Program-2**: Write a Java RMI program to accept an integer and calculate the sum of factorial

of each digit in it. E.g.: Num = 251 → S = 2! + 5! + 1!

**Reg No**: 24251115

**Date**: 23/10/2025

\*/

**Interface:**

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface FactSumInter extends *Remote*

{

    public *int* sumFact(*int* *num*) throws RemoteException;

}

**Implement:**

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class FactSumImpl extends UnicastRemoteObject implements FactSumInter

{

    public FactSumImpl() throws RemoteException

    {

        super();

    }

    public *int* sumFact(*int* *num*) throws RemoteException{

*int* rem=0,sum=0,fact;

        while(*num*>0)

        {

            rem=*num*%10;

            if(rem==0)

                fact=0;

            else

                fact=1;

            for(*int* i=2;i<=rem;i++)

                fact=fact\*i;

            sum+=fact;

*num*=*num*/10;

        }

        return sum;

    }

}

**Client:**

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

import java.util.Scanner;

public class FactSumClient

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

    {

        Scanner sc=new Scanner(System.in);

        try

        {

            Registry reg=LocateRegistry.getRegistry(5000);

            FactSumInter temp=(FactSumInter) reg.lookup("factsum");

            System.out.println("Client is accessing...");

            System.out.print("\nEnter an Integer: ");

*int* num=sc.nextInt();

            System.out.println("The sum of the factorials of each digit in an integer   "+num+" is: "+temp.sumFact(num));

        }

        catch(Exception e){

            System.out.println(e.getMessage());

            sc.close();

        }

    }

}

**Server:**

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

public class FactSumServer

{

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

    {

        try{

            FactSumImpl obj=new FactSumImpl();

            Registry reg=LocateRegistry.createRegistry(5000);

            reg.rebind("factsum",obj);

            System.out.println("Server is started...");

        }

        catch(Exception e){

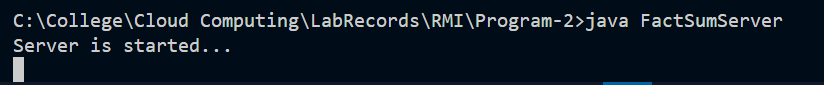
            System.out.println(e.getMessage());}

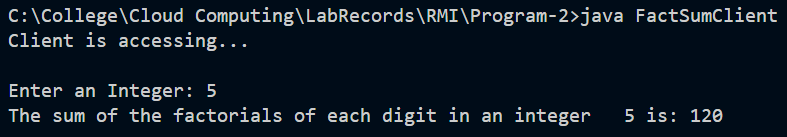
    }

}

Output:







/\*

**Program-3**: Write a Java RMI program to accept some integers and store the same in an array

then find the maximum element among those.

**Reg No**: 24251115

**Date**: 23/10/2025

\*/

**Interface**:

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface MaxEleInterface extends *Remote* {

   public *int* findMax(*int* *num*[]) throws RemoteException;

}

**Implement**:

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class MaxEleClass extends UnicastRemoteObject implements MaxEleInterface {

    public MaxEleClass() throws RemoteException {

        super();

    }

    public *int* findMax(*int* *num*[]) throws RemoteException{

*int* max = *num*[0];

        for (*int* i = 0; i<*num*.length;i++)

            if(*num*[i]>max)

                max = *num*[i];

        return max;

    }

}

**Client**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

import java.util.Scanner;

public class MaxEleClient {

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

        Scanner sc = new Scanner(System.in);

        try {

            Registry reg = LocateRegistry.getRegistry(8000);

            MaxEleInterface temp = (MaxEleInterface) reg.lookup("maxele");

            System.out.println("Client is accessing....");

            System.out.print("Enter the size of an array: ");

*int* n = sc.nextInt();

*int* num[] = new *int*[n];

            System.out.print("Enter  " + n + " elements: ");

            for(*int* i=0; i<num.length;i++){

                num[i]=sc.nextInt();

            }

*int* max = temp.findMax(num);

            System.out.println("The maximum element among the given array is: " +max);

        }

        catch (Exception e) {

            System.out.println(e.getMessage());

            sc.close();

        }

    }

}

**Server**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

public class MaxEleServer {

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

        try {

            MaxEleClass obj = new MaxEleClass();

            Registry reg = LocateRegistry.createRegistry(8000);

            reg.rebind("maxele", obj);

            System.out.println("Server is started....");

        }

        catch (Exception e) {

            System.out.println(e.getMessage());

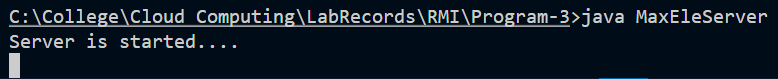
        }

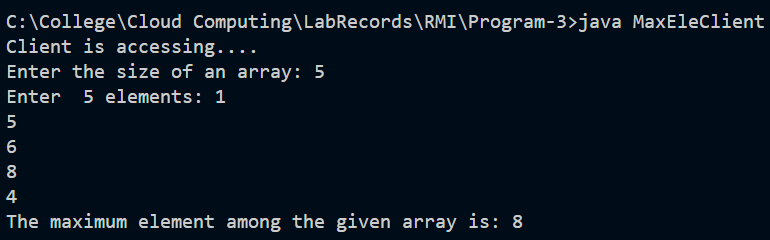
    }

}

Output:







/\*

**Program-4**: Write a Java RMI program to accept a String as command-line argument and

check whether it has repeated characters in it or not.

**Reg No**: 24251115

**Date**: 16/09/2025

\*/

**Interface:**

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface FindRepCharInterface extends *Remote*{

    public *boolean* findRepChar(String *str*) throws RemoteException;

}

**Implement:**

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class FindRepCharImpl extends UnicastRemoteObject implements FindRepCharInterface{

    public FindRepCharImpl() throws RemoteException{

        super();

    }

    public *boolean* findRepChar(String *str*) throws RemoteException{

        for (*int* i = 0; i < *str*.length(); i++) {

            for (*int* j = i + 1; j < *str*.length(); j++) {

                if (*str*.charAt(i) == *str*.charAt(j)) {

                    return true;

                }

            }

        }

        return false;

    }

}

**Client**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

import java.util.Scanner;

public class FindRepCharClient{

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

        Scanner sc = new Scanner(System.in);

        try{

            Registry reg = LocateRegistry.getRegistry(5000);

            FindRepCharInterface temp = (FindRepCharInterface) reg.lookup("findrepchar");

            System.out.println("Client is accessing...");

            String st = *args*[0];

            if(temp.findRepChar(st))

                System.out.println("\nThe given string have repeatedcharacters.....");

            else

                System.out.println("\nThe given string does not have repeated characters.....");

        }

        catch (Exception e){

            System.out.println(e.getMessage());

            sc.close();

        }

    }

}

**Server**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

public class FindRepCharServer{

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

        try{

            FindRepCharImpl obj = new FindRepCharImpl();

            Registry reg = LocateRegistry.createRegistry(5000);

            reg.rebind("findrepchar",obj);

            System.out.println("Server is started...");

        }

        catch (Exception e){

            System.out.println(e.getMessage());

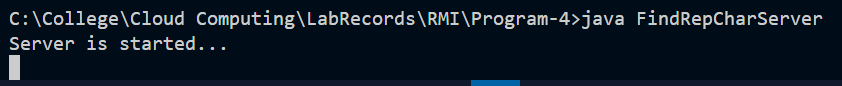
        }

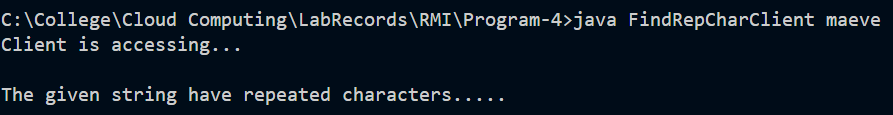
    }

 }

**Output**:







/\*

**Program-5**: Write a Java RMI program to accept a line of text and display only the strings

which contains vowels in it.

**Reg No**: 24251115

**Date**: 23/10/2025

\*/

**Interface**:

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface StrVowInterface extends *Remote*{

    public *boolean* findVow(String *str*) throws RemoteException;

}

**Client**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

import java.util.Scanner;

public class StrVowClient{

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

        Scanner sc = new Scanner(System.in);

        try{

            Registry reg = LocateRegistry.getRegistry(4000);

            StrVowInterface temp = (StrVowInterface) reg.lookup("checkvow");

            System.out.println("Client is accessing...");

            System.out.print("\nEnter a line of text: ");

            String st = sc.nextLine();

            String strArray[] = st.split(" ");

            System.out.println("\nWords With Vowels: ");

            for (*int* i = 0 ; i < strArray.length; i++){

                String str = strArray[i];

                if(temp.findVow(str))

                    System.out.println(str);}

        }

        catch (Exception e){

            System.out.println(e.getMessage());

            sc.close();

        }

    }

}

**Server**:

import java.rmi.registry.Registry;

import java.rmi.registry.LocateRegistry;

public class StrVowServer {

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

        try{

            StrVowImpl obj = new StrVowImpl();

            Registry registry = LocateRegistry.createRegistry(4000);

            registry.rebind("checkvow", obj);

            System.out.println("Server is started...");

        }catch(Exception e){

            System.out.println(e.getMessage());

        }

    }

}

**Implement**:

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class StrVowImpl extends UnicastRemoteObject implements StrVowInterface{

    public StrVowImpl() throws RemoteException{

        super();

    }

    public *boolean* findVow(String *str*) throws RemoteException{

*boolean* isVow = false ;

        for (*int* i = 0 ; i < *str*.length() ; i++)

            if(*str*.charAt(i) =='a' || *str*.charAt(i) =='e' || *str*.charAt(i) =='i' || *str*.charAt(i) =='o' || *str*.charAt(i) =='u'){

                isVow = true;

                break;

            }

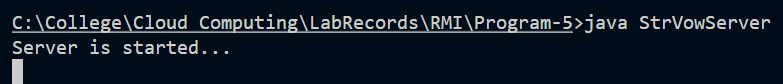
        return isVow;

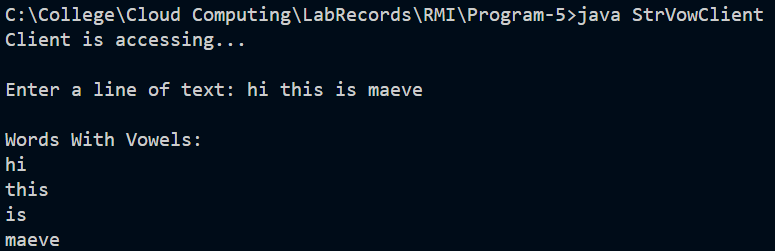
    }

}

**Output**:







/\*

**Program-6**: Write a Java RMI program to accept a line of text and display each string after

deleting all the repeated characters in it if it contains.

**Reg No**: 24251115

Date: 23/10/2025

\*/

**Interface**:

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface DelRepCharInterface extends *Remote*{

    public String delRepChar(String *str*) throws RemoteException;

}

**Client**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

import java.util.Scanner;

public class DelRepCharClient{

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

        Scanner sc = new Scanner(System.in);

        try{

            Registry reg = LocateRegistry.getRegistry(5000);

            DelRepCharInterface temp = (DelRepCharInterface) reg.lookup("delrepchar");

            System.out.println("Client is accessing...");

            System.out.print("\nEnter a line of text: ");

            String st = sc.nextLine();

            String strArray[] = st.split(" ");

            System.out.println("\nWords after deleting repeated Characters.....");

            for (*int* i = 0 ; i < strArray.length; i++){

                String str = strArray[i];

                System.out.print(temp.delRepChar(str)+" ");

            }

        }

        catch (Exception e){

            System.out.println(e.getMessage());

            sc.close();

        }

    }

}

**Server**:

import java.rmi.registry.LocateRegistry;

import java.rmi.registry.Registry;

public class DelRepCharServer{

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

        try{

            DelRepCharImpl obj = new DelRepCharImpl();

            Registry reg = LocateRegistry.createRegistry(5000);

            reg.rebind("delrepchar",obj);

            System.out.println("Server is started...");

        }

        catch (Exception e){

            System.out.println(e.getMessage());}

    }

 }

**Implement**:

import java.rmi.server.UnicastRemoteObject;

import java.rmi.RemoteException;

public class DelRepCharImpl extends UnicastRemoteObject implements DelRepCharInterface{

    public DelRepCharImpl() throws RemoteException{

        super();

    }

    public String delRepChar(String *str*) throws RemoteException{

        StringBuffer st = new StringBuffer(*str*);

        for (*int* i = 0; i < st.length(); i++) {

            for (*int* j = i + 1; j < st.length(); j++) {

                if (st.charAt(i) == st.charAt(j)) {

                    st.deleteCharAt(j);

                }

            }

        }

        String newst = st.toString();

        return newst;

    }

}

**Output**:



