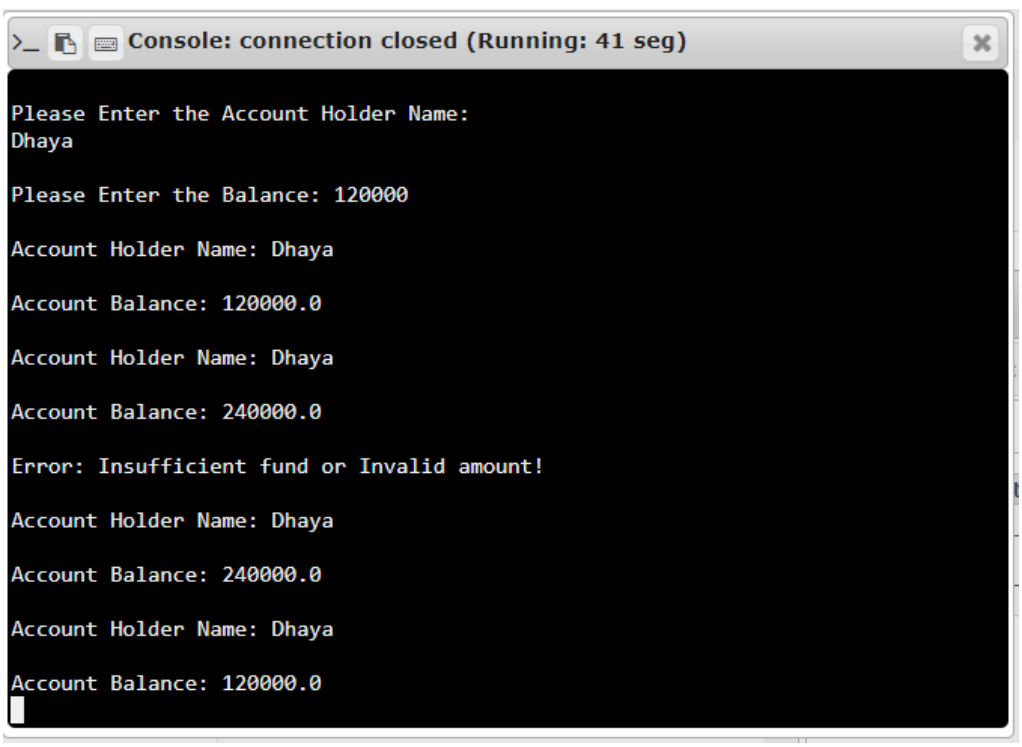
PROGRAM 1.1

```
import java.util.*;

class bank

{

   public String OwnName;

   public int accNo;

   public double balance;


   public bank(String OwnName,double balance)

   {

      this.OwnName=OwnName;

      this.accNo=accNo;

      this.balance=balance;


   }

   public void display()

   {

      System.out.println("\nAccount Holder Name: "+this.OwnName);

      System.out.println("\nAccount Balance: "+this.balance);

   }

   public void deposit(double dep)

   {

      balance+=dep;

   }

   public void withdraw(double w)
```

```
    {
        if(w>balance)
        {
            System.out.println("\nError: Insufficient fund or Invalid amount!");
        }
        else
        {
            balance-=w;
        }


    }
}
public class exercise1
{
    public static void main(String args[])
    {
        int val=0;
        Scanner getval=new Scanner(System.in);
        System.out.println("Please Enter an Account Number: ");
        int accNo=getval.nextInt();
        getval.nextLine();
        System.out.println("\nPlease Enter the Account Holder Name: ");
        String OwnName=getval.nextLine();
        System.out.print("\nPlease Enter the Balance: ");
        double balance=getval.nextDouble();
```

```
        bank acc=new bank(OwnName,balance);

        acc.display();

        acc.deposit(acc.balance);

        acc.display();


        acc.withdraw(2*acc.balance);

        acc.display();


        acc.withdraw(balance);

        acc.display();

    }

}
```

OUTPUT



PROGRAM 1.2

```
import java.lang.Math;

class City{

   public String name;

   public double lon;

   public double lat;

   public City(String name,double lon,double lat){

      this.name=name;

      this.lon=lon;

      this.lat=lat;

   }

   public void report(){

      System.out.println("City: "+this.name+" is at: "+this.lon+" , "+this.lat);

   }


   public double distancefrom(double lon1,double lat1,double lon2,double lat2){

      double R=6371;

      double dlon=(lon2-lon1)*(Math.PI/180);

      double radlat1=lat1*(Math.PI/180);

      double radlat2=lat2*(Math.PI/180);

      double dlat=(lat2-lat1)*(Math.PI/180);

      double a
=Math.pow(Math.sin(dlat/2),2)+Math.cos(radlat1)*Math.cos(radlat2)*Math.pow(Math.sin(dlon/2),2);

      double c=2*Math.atan2(Math.sqrt(a),Math.sqrt(1-a));
```

```
        double distance=R*c;
        return distance;
    }
}


public class exercise2{
    public static void main(String args[]){
        City ob1=new City("SLM",11.6643,78.1460);
        City ob2=new City("CBR",11.0168,76.9558);
        System.out.println("City #1");
        System.out.println("Name: "+ob1.name);
        System.out.println("Longitude: "+(int)ob1.lon);
        System.out.println("Latitude: "+(int)ob1.lat);
        System.out.println();
        System.out.println("City #2");
        System.out.println("Name: "+ob2.name);
        System.out.println("Longitude: "+(int)ob2.lon);
        System.out.println("Latitude: "+(int)ob2.lat);
        System.out.println();
        ob1.report();
        ob2.report();
        int dist=(int)ob2.distancefrom(ob1.lon,ob1.lat,ob2.lon,ob2.lat);
        System.out.println(ob1.name+" is "+dist+" kms away from "+ob2.name);
    }
}
```

OUTPUT

```
Console: connection closed (Running)
City #1
Name: SLM
Longitude: 11
Latitude: 78

City #2
Name: CBR
Longitude: 11
Latitude: 76

City: SLM is at: 11.6643 , 78.146
City: CBR is at: 11.0168 , 76.9558
SLM is 133 kms away from CBR
```

PROGRAM 2.1

```
import java.util.Scanner;

import java.util.InputMismatchException;

public class exercise6{

  public static void main(String[] args)

   {

      Scanner scan = new Scanner(System.in);

      for(int count = 0 ; count <= 2 ; count++){

      try {

                int num = 0;

                 do{

                       System.out.println("Enter a number between 1 and 10");

                      num = scan.nextInt();


                      if(num < 1 || num > 10)

                      System.out.println("\nIllegal value, " + num + " entered.  Please try again.");


                    }while(num < 1 || num > 10);

                    System.out.println("\nValue correctly entered! Thank you.");

                    break;

       }catch(InputMismatchException ime) {

                      System.out.println("Enter whole numbers only, with no spaces or other characters");

                      scan.next();

            }

      }
```

```
  }

}
```

OUTPUT

```
Console: connection closed (Running: 13 seg)
Enter a number between 1 and 10
3.5
Enter whole numbers only, with no spaces or other characters
Enter a number between 1 and 10
15

Illegal value, 15 entered.  Please try again.
Enter a number between 1 and 10
6

Value correctly entered! Thank you.
```

PROGRAM 2.2

```
import java.io.*;

import java.util.*;

class GradeException{

   Hashtable<Integer , String> ht=new Hashtable<>();

   public static String grade[]=new String[7];

   static{

      grade[0]="A";

      grade[1]="B";

      grade[2]="C";

      grade[3]="D";

      grade[4]="E";

      grade[5]="F";

      grade[6]="I";

   }

   void validGrade(int id , String c)throws Exception{

      List <String> GradeList = new ArrayList<>(Arrays.asList(grade));

      if(GradeList.contains(c)){

         ht.put(id,c);

      }

      else throw new Exception("Grade Exception");


   }

   void display(){

      System.out.println("Key/Values in HasHtable are:\n"+ht);
```

```
    }
}
public class exercise7{
    public static void main (String arg[])throws IOException{
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        GradeException g = new GradeException();
        int ID[] = new int[5];
        String grd;
        for(int i=0;i<5;i++){
            ID[i]=i+101;
            System.out.print("The Student ID is :"+ID[i]+"\nEnter the grade: ");
            grd=in.readLine();
            try{
                g.validGrade(ID[i],grd);
            }
            catch(Exception e){
                System.out.println(e);
            }
        }
        g.display();
    }
}
```

OUTPUT

```
Console: connection closed (Running: 29 seg)
The Student ID is :101
Enter the grade: A
The Student ID is :102
Enter the grade: B
The Student ID is :103
Enter the grade: R
java.lang.Exception: Grade Exception
The Student ID is :104
Enter the grade: I
The Student ID is :105
Enter the grade: G
java.lang.Exception: Grade Exception
Key/Values in HasHtable are:
{104=I, 102=B, 101=A}
```

PROGRAM 3.1

```
import java.util.Scanner;

public class Add{

   public void add(int n1, int n2){

      System.out.println(n1+n2);

   }

   public void add(double n1, double n2){

      System.out.println(n1+n2);

   }

   public void add(int n1, int n2, int n3){

      System.out.println(n1+n2+n3);

   }

   public static void main(String[] args){

      Scanner input = new Scanner(System.in);

      Add ob = new Add();

      int e1 = input.nextInt(), e2=input.nextInt(), e3= input.nextInt();

      double e4 = input.nextDouble(), e5 = input.nextDouble();

      ob.add(e1, e2);

      ob.add(e1, e2, e3);

      ob.add(e4, e5);

   }

}
```

OUTPUT

```
4
9
3
2.3
4.5
13
16
6.8


...Program finished with exit code 0
Press ENTER to exit console.
```

PROGRAM 4.1

```
class Account
{
    private double bal;
    //private int accnum
    public Account (double bl)
    {
        bal=bl;
    }
    public void deposit(double sum)
    {
        if (sum>0)
            bal+=sum;
        else
            System.err.println("Account.deposit(...):"+"cannot deposit negative amount.");


    }
    public void withdraw(double sum)
    {
        if (sum>0)
            bal-=sum;
        else
            System.err.println("Account.withdraw(...):"+"cannot withdraw negative amount.");
    }
```

```
    public double getBalance()
    {
        return bal;
    }
    public final void intprint(double in)
    {
        System.out.println("Savings Account Balance = "+bal+" Interest : "+in);
    }
    public final void limitprint(double l)
    {
        System.out.println("Current Account Balance = "+bal+" Limit : "+l);
    }
}
class savingsAccount extends Account{
    double interest;
    savingsAccount(double b,double i){
        super(b);
        this.interest=i;
        super.intprint(interest);
    }
    public void updateinterest(double i){
        interest=i;
        System.out.println("After updating the interest rate");
    }
```

```
    public void addinterest(double i){
        double b1,j;
        b1=super.getBalance();
        j=(b1*i)/100;
        super.deposit(j);
        super.intprint(interest);
    }
}
class currentAccount extends Account{
    double limit;
    currentAccount(double b,double li){
        super(b);
        this.limit=li;
        super.limitprint(limit);
    }
    public void updatelimit(double li){
        limit=li;
        System.out.println("After updating the withdrawn limit");
        super.limitprint(li);
    }
    public void checklimit(double amt){
        if(amt<=limit){
            super.withdraw(amt);
            System.out.println("Withdraw Rs. "+(int)amt+" from Current Account");
```

```
            super.limitprint(limit);
        }
        else{
            System.out.println("Withdraw Rs. "+(int)amt+" from Current Account");
            System.out.println("Sorry, the limit is exceeded");
            super.limitprint(limit);
        }
    }
}
public class exercise3{
    public static void main(String args[]){
        savingsAccount ac=new savingsAccount(10000,0.25);
        currentAccount acc=new currentAccount(20000.0,1000.0);
        ac.updateinterest(1.25);
        ac.addinterest(1.25);
        acc.updatelimit(2000.0);
        acc.checklimit(1000.0);
        acc.checklimit(1000.0);
        acc.checklimit(3000.0);
    }
}
```

OUTPUT

```
Console: connection closed (Running)
Savings Account Balance = 10000.0 Interest : 0.25
Current Account Balance = 20000.0 Limit : 1000.0
After updating the interest rate
Savings Account Balance = 10125.0 Interest : 1.25
After updating the withdrawn limit
Current Account Balance = 20000.0 Limit : 2000.0
Withdraw Rs. 1000 from Current Account
Current Account Balance = 19000.0 Limit : 2000.0
Withdraw Rs. 1000 from Current Account
Current Account Balance = 18000.0 Limit : 2000.0
Withdraw Rs. 3000 from Current Account
Sorry, the limit is exceeded
Current Account Balance = 18000.0 Limit : 2000.0
```

PROGRAM 4.2

```
interface IntOperations {
	void integer();
	void prime();
	void evenOdd();
	void factorial();
	void Sumofdigit();
}
class MyNumber implements IntOperations {
	int i = 0 , no;
	MyNumber() {
		no = 0;
	}
	MyNumber(int num) {
		no = num;
	}
	public void integer() {


		if (no < 0) {
			System.out.println(no + " is a Negative Number");
		} else if (no > 0) {
			System.out.println(no + " is a Positive Number");
		} else {
			System.out.println(no + " is a Positive Number");
```

```
        }

    }
    public void prime() {
        int flag = 0;
        for (i = 2; i < no; i++) {
            if (no % i == 0) {
                flag = 1;
            }
        }

        if (flag == 1) {
            System.out.println(no + " is not a Prime Number");
        } else {
            System.out.println(no + " is a Prime Number");
        }

    }
    public void evenOdd() {

        if (no % 2 == 0) {
            System.out.println(no + " is a Even Number");
        } else {
            System.out.println(no + " is a Odd Number");
```

```
        }

    }
    public void factorial(){
        int fact = 1;
        for (i = 1; i <= no; i++) {
            fact = fact * i;
        }

        System.out.println("The factorial of " + no + " is " + fact);

    }
    public void Sumofdigit() {
        int sum = 0, rem = 0, n = 0;
        while (no > 0) {
            rem = no % 10;
            sum = sum + rem;
            no = no / 10;
        }

        System.out.println("Sum of it's digits is " + sum);

    }
}
```

```
class exercise4{
	public static void main(String  args[]){
		MyNumber m = new MyNumber(11);
		m.integer();
		m.evenOdd();
		m.prime();
		m.factorial();
		m.Sumofdigit();
		System.out.println("\n");
		MyNumber n = new MyNumber(14);
		n.integer();
		n.evenOdd();
		n.prime();
		n.factorial();
		n.Sumofdigit();
	}
}
```

OUTPUT

Console: connection closed (Running)

```
11 is a Positive Number
11 is a Odd Number
11 is a Prime Number
The factorial of 11 is 39916800
Sum of it's digits is 2


14 is a Positive Number
14 is a Even Number
14 is not a Prime Number
The factorial of 14 is 1278945280
Sum of it's digits is 5
```

PROGRAM 4.3

```
import java.io.BufferedReader;

import java.io.InputStreamReader;

interface StackOperations{

   int max = 5;

   void push(int data);

   void pop();

   int isempty();

   int isfull();

}

class MyStack implements StackOperations{

   private int arr[]=new int[StackOperations.max];

   private int top;

   public MyStack(){top = 0;}

   public void push(int data){arr[top++]=data;}

   public void pop(){top--;}

   public int isempty(){

      if(top==0){return 1;}

      else{return 0;}

   }

   public int isfull(){

      if (top == max){return 1;}

      else{return 0;}

   }
```

```
    public void display(){
        int i;
        if(top>0){
            System.out.println("The Elements in the Stack are: ");
            for(i=top-1;i>=0;i--)
                System.out.println(arr[i]);
        }else{isempty();}
    }
}
public class exercise5{
    public static void main(String[] args) throws Exception{
        int ch, data;
        String c;
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        MyStack s = new MyStack();
        do{
            System.out.println("1:Push");
            System.out.println("\n2:Pop");
            System.out.println("\n3:Display");
            System.out.println("\n4:Exit");
            System.out.println("\nEnter your choice:");
            ch = Integer.parseInt(br.readLine());
            switch(ch){
                case 1:
```

```
            if (s.isfull()==1){System.out.println("Stack is full");}
            else{
                System.out.println("Enter the data:");
                data = Integer.parseInt(br.readLine());
                s.push(data);
            }break;
        case 2:
            if(s.isempty()==1){System.out.println("Stack is empty");}
            else{s.pop();}break;
        case 3:
            if (s.isempty()==1){System.out.println("Stack is empty");}
            else{s.display();}break;
        case 4:
            System.exit(0);
        case 5:
            System.out.println("\nInvalid choice");
            break;
      }
      System.out.println();
    }while(ch!=4);
  }
}
```

OUTPUT

```
1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
1
Enter the data:
10

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
1
Enter the data:
20

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
1
Enter the data:
30

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
1
Enter the data:
40

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
2

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
3
```

```
3:Display

4:Exit

Enter your choice:
2

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
3
The Elements in the Stack are:
30
20
10

1:Push

2:Pop

3:Display

4:Exit

Enter your choice:
4


...Program finished with exit code 0
Press ENTER to exit console.
```

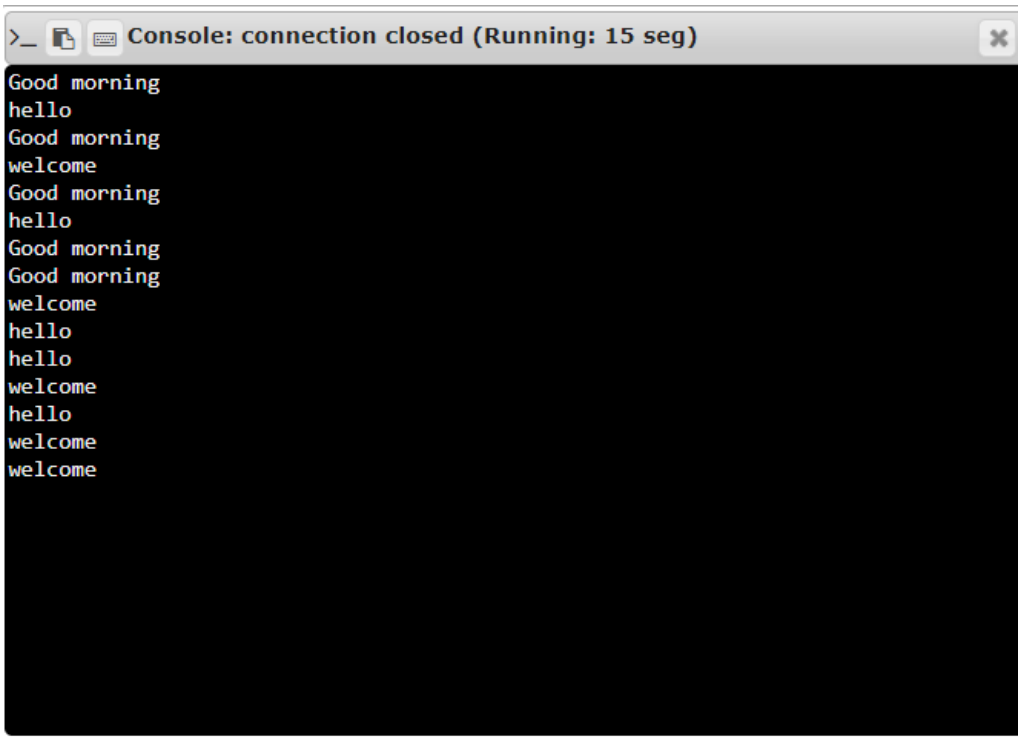
PROGRAM 5.1

```
class GoodMorning extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(1000);
                System.out.println("Good morning ");
                i++;
            }
        } catch (Exception e) {
        }
    }
}


class Hello extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(2000);
                System.out.println("hello");
                i++;
            }
```

```
        } catch (Exception e) {
        }
    }
}

class Welcome extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(3000);
                System.out.println("welcome");
                i++;
            }
        } catch (Exception e) {
        }
    }
}

class MultithreadDemo {
    public static void main(String args[]) {
        GoodMorning t1 = new GoodMorning();
        Hello t2 = new Hello();
        Welcome t3 = new Welcome();
```

```
        t1.start();

        t2.start();

        t3.start();


    }

}
```

OUTPUT



PROGRAM 5.2

```
import java.lang.*;

import java.util.*;

import java.awt.*;

class One implements Runnable

{

One()

{

new Thread(this,"one").start();

try

{

Thread.sleep(1000);

}

catch(InterruptedException e)

{

}

}

public void run()

{

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

{

try

{

Thread.sleep(1000);

}
```

```
catch(InterruptedException e)
{
}
System.out.println("Good Morning");
}
}
}
class Two implements Runnable
{
Two()
{
new Thread(this,"two").start();
try
{
Thread.sleep(2000);
}
catch(InterruptedException e)
{
}
}
public void run()
{
for(int j=0;j<5;j++)
{
```

```
try
{
Thread.sleep(2000);
}
catch(InterruptedException e)
{
}
System.out.println("Hello");
}
}
}
class Three implements Runnable
{
Three()
{
new Thread(this,"Three").start();
try
{
Thread.sleep(3000);
}
catch(InterruptedException e)
{
}
}
```

```
public void run()
{
for(int k = 0;k<5;k++)
{
try
{
Thread.sleep(3000);
}
catch(InterruptedException e)
{
}
System.out.println("Welcome");
}
}
}
class MyThread
{
public static void main(String args[])
{
One obj1=new One();
Two obj2=new Two();
Three obj3=new Three();
}
}
```

OUTPUT

Console: connection closed (Running: 9 seg)

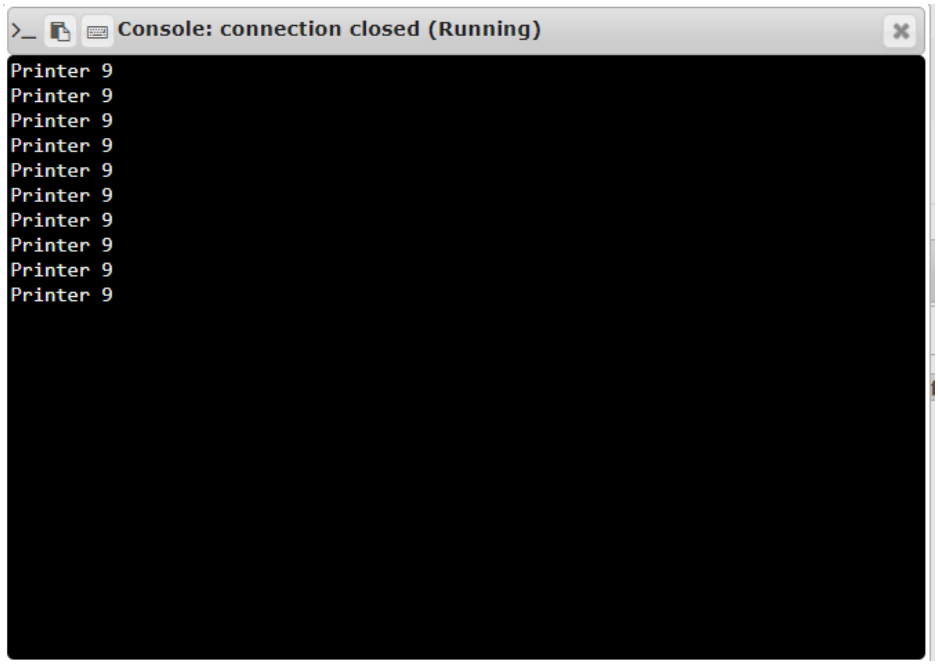
```
Good Morning
Good Morning
Good Morning
Hello
Good Morning
Hello
Good Morning
Welcome
Hello
Hello
Welcome
Hello
Welcome
Welcome
Welcome
```

PROGRAM 6

```
public class Counter implements  Runnable{


    public static void main(String[] args) {

        Storage store = new Storage();

        Counter c1 = new Counter(store);

        Printer p1 = new Printer(store);

    }

    Storage st;

    public Counter(Storage store){

        st = store;

        new Thread(this, "Counter").start();


    }

    @Override

    public void run() {

        for(int i = 0 ; i < 10; i++){

            st.setValue(i);

        }


    }


}

class Printer implements Runnable{

    Storage st;
```

```
    public Printer(Storage st){
        this.st = st;
        new Thread(this, "Printer").start();
    }
    @Override
    public void run() {
        for(int i = 0; i < 10; i++)
        System.out.println(Thread.currentThread().getName() + " " + st.getValue());
    }

}
class Storage{
    int  i;
    public synchronized void setValue(int i){
        this.i = i;
    }
    public synchronized int getValue(){
        return this.i;
    }
}
```

OUTPUT

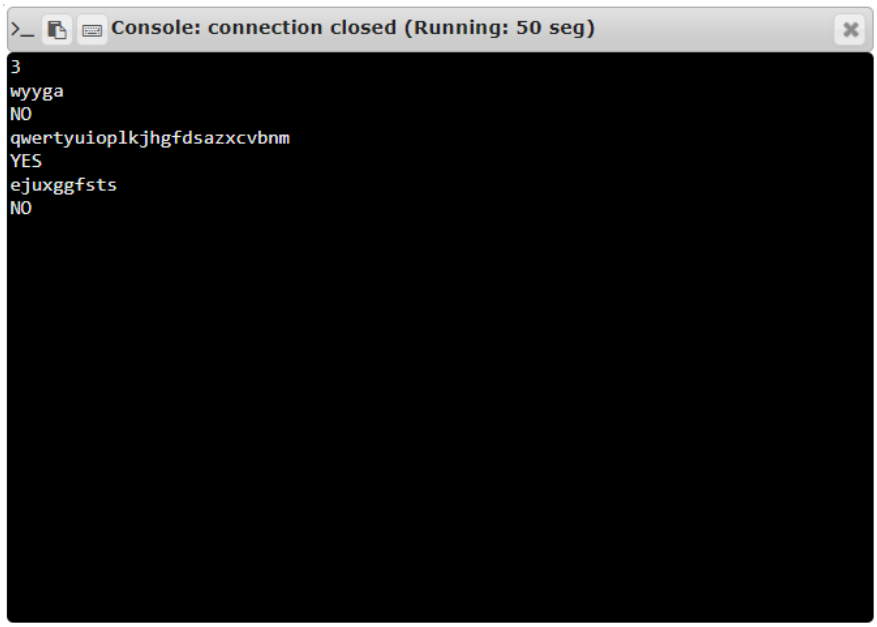

Console: connection closed (Running)
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9
Printer 9

PROGRAM 7

```
import java.util.Scanner;

public class Strings_ex1

{

   public static void main(String[] args)

   {

      int N;

      String str;

      Scanner in = new Scanner(System.in);

      N = in.nextInt();

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

         str = in.next();

         System.out.println(checkAllChars(str));

      }

   }


   private static String checkAllChars ( String input )

   {

      //If input length is less than 26 then it can never be complete

      if(input.length() < 26)

      {

         return "NO";

      }


      for (char ch = 'A'; ch <= 'Z'; ch++)
```

```
        {
            if (input.indexOf(ch) < 0 && input.indexOf((char) (ch + 32)) < 0)
            {
                return "NO";
            }
        }
        return "YES";
    }
}
```

OUTPUT

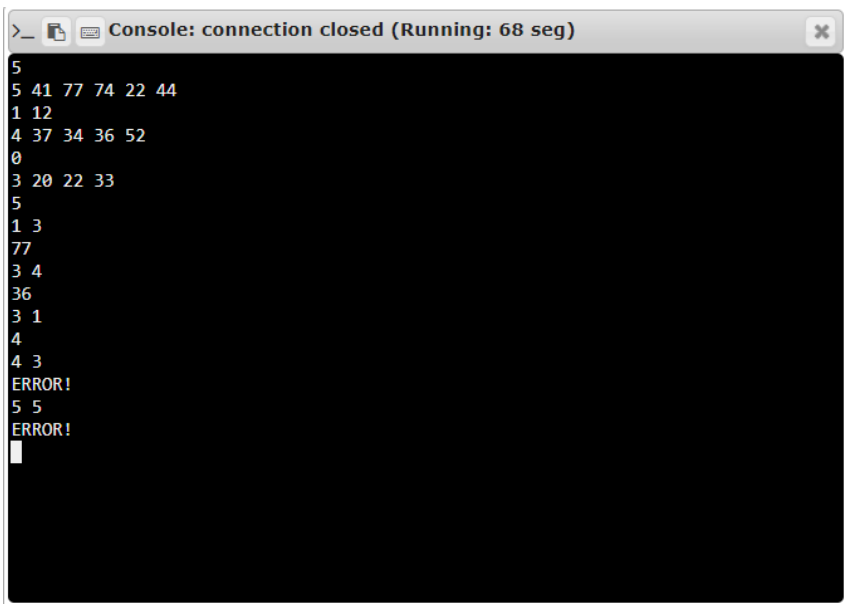


PROGRAM 8

```
import java.util.*;

 public class Main {

   public static void main(String[] args) {

      //define new ArrayList and initialize it

      ArrayList<ArrayList> numList = new ArrayList<ArrayList>();

      int n=0;

      Scanner in = new Scanner(System.in);

      n=in.nextInt();


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

         ArrayList<Integer> na = new ArrayList<Integer>();

         int nr = in.nextInt();

         na.add(nr);

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

            int elm = in.nextInt();

            na.add(elm);

         }

         numList.add(na);

      }

      n=in.nextInt();

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

         try {

            int x=in.nextInt();

            int y=in.nextInt();
```

```
            System.out.println(numList.get(x-1).get(y-1));
        } catch (Exception e) {
            //TODO: handle exception
            System.out.println("ERROR!");
        }
    }


  }
}
```

OUTPUT



PROGRAM 9

```
import java.io.*;

import java.util.*;


class SortedList{

   ArrayList<Integer> array=null;

   SortedList(){

      array = new ArrayList<Integer>();

   }

   public void add(int u){

      array.add(u);

      Collections.sort(array);

   }

   public boolean isEmpty(){

      return array.isEmpty();

   }

   public int getFirst(){

      return array.get(0);

   }

   public int getLast(){

      return array.get(array.size()-1);

   }

}


public class Exercise10{
```

```
public static void main (String arg[]){
        String data= null;
        Scanner sc= new Scanner(System.in);

        data=sc.nextLine();
        try{

        File fi=new File(data.trim());
        FileReader fr= new FileReader(fi);
        BufferedReader dip= new BufferedReader(fr);
        String i;
        SortedList sl= new SortedList();
        while((i=dip.readLine())!=null){

            sl.add(Integer.parseInt(i));

        }
        if(sl.isEmpty()){
            System.out.println("Empty array");
            System.out.println("min undefined");
            System.out.println("max undefined");
        }

        else{
```

```
            System.out.println("min = "+sl.getFirst());
            System.out.println("max = "+sl.getLast());
        }
        fr.close();
        }
        catch(Exception e){
            System.out.println(e);
        }
    }
  }
```

OUTPUT