RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE

CS23333
OBJECT ORIENTED PROGRAMMING USING JAVA

Laboratory Observation Note Book

Name : Girivasanth V

Year / Branch / Section : 2nd Year/ AIML / A

Register No. : 231501048

Semester : 3rd Semester

Academic Year : 2024-2025

Reg. No: 231501048 **Name: Girivasanth V**

Year: 2nd Year **Branch: AIML** **Sec:A**

S. No.	Date	Title	Page No.	Teacher's Signature/ Remarks
Java Architecture, Language Basics				
1.1		Problem 1		
1.2		Problem 2		
1.3		Problem 3		
Flow Control Statements				
2.1		Problem 1		
2.2		Problem 2		
2.3		Problem 3		
Arrays				
3.1		Problem 1		
3.2		Problem 2		
3.3		Problem 3		
Classes and Objects				
4.1		Problem 1		
4.2		Problem 2		
4.3		Problem 3		
Inheritance				
5.1		Problem 1		
5.2		Problem 2		
5.3		Problem 3		
String, StringBuffer				
6.1		Problem 1		
6.2		Problem 2		
6.3		Problem 3		
Interfaces				
7.1		Problem 1		
7.2		Problem 2		
7.3		Problem 3		

Polymorphism, Abstract Classes, final Keyword				
8.1		Problem 1		
8.2		Problem 2		
8.3		Problem 3		
Exception Handling				
9.1		Problem 1		
9.2		Problem 2		
9.3		Problem 3		
Collection- List				
10.1		Problem 1		
10.2		Problem 2		
10.3		Problem 3		
Set, Map				
11.1		Problem 1		
11.2		Problem 2		
11.3		Problem 3		
Introduction to **1/0**, 1/0 Operations, Object Serialization				
12.1		Problem 1		
12.2		Problem 2		
12.3		Problem 3		

01 - JAVA ARCHITECTURE, LANGUAGE BASICS

Ex.No. **1.1** **Date:**

Register No.: **231501048** **Name: Girivasanth V**

Odd or Even

Write a program to find whether the given input number is Odd.
If the given number is odd, the program should return 2 else It should return 1.
Note: The number passed to the program can either be negative. positive or zero.
Zero should NOT be treated as Odd.

For example:

Input	Result
123	2
456	1

PROGRAM

```
import java.util.Scanner;
public class odd{
    public static void main(String[] args){
        int n;
        Scanner sc = new Scanner(System.in);
        n=sc.nextint();
        if (n%2==0){
            System.out.println("1");

        }
        else{
            System.out.println("2");

        }
    }}
```

	input	Expected	Got	
v	123	2	2	v
v	456	1	1	v

Passed all tests! v

Ex.No. 1.2 **Date:**

Register No.: 231501048 **Name:** Girivasanth V

Return Last number of Digit

Write a program that returns the last digit of the given number. Last digit is being referred to the least significant digit i.e. the digit in the ones (units) place in the given number.
The last digit should be returned as a positive number.
For example,
if the given number is 197, the last digit is 7
if the given number is -197, the last digit is 7

For example:

Input	Result
197	7
-197	7

PROGRAM

```
import java.util.Scanner;
import java.lang.Math;
public class odd{
    public static void main(String[] args){
        int s;
        Scanner sc = new Scanner(System.in);
        s=Math.abs(sc.nextint());
        System.out.println(s%10);
    }
}
```

	Input	Expected	Got	
✓	197	7	7	✓
✓	-197	7	7	✓

Passed all tests! ✓

Ex.No. 1.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Add last 2 Digits

Rohit wants to add the last digits of two given numbers.

For example,

If the given numbers are 267 and 154, the output should be 11.

Below is the explanation:

Last digit of the 267 is 7

Last digit of the 154 is 4

Sum of 7 and 4 = 11

Write a program to help Rohit achieve this for any given two numbers.

Note: Tile sign of the input numbers should be ignored.

i.e.if the input numbers are 267 and 154, the sum of last two digits should be 11

if the input numbers are 267 and -154, the slim of last two digits should be 11

if the input numbers are -267 and 154, the sum of last twodigits should be 11

if the input numbers are -267 and -154, the sum oflast two digits should be 11

For example:

Input	Result
267 154	11
267 -154	11
-267 154	11
-267 -154	11

PROGRAM

```
import java.util.Scanner;
import java.lang.Math;
public class sign{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n1=sc.nextint();
        int n2=sc.nextint();
        int s=Math.abs(n1%10}+Math.abs(n2%10};
        System.out.println(s);
    }
}
```

	Input	Expected	Got	
✓	267 154	11	11	✓
✓	267 -154	11	11	✓
✓	-267 154	11	11	✓
✓	-267 -154	11	11	✓

Passed all tests!

02- FLOW CONTROL STATEMENTS

Ex.No. 2.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Consider a sequence of the form 0, 1, 1, 2, 4, 7, 13, 24, 44, 81, 149...

Write a method program which takes as parameter an integer n and prints the nth term of the above sequence. The nth term will fit in an integer value.

Example Input:

5

Output:

4

Example Input:

8

Output:

24

Example Input:

11

Output:

149

For example:

Input	Result
5	4
8	24
11	149

PROGRAM

```
import java.util.Scanner;
public class seq{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextint();
        int[] seq= new int[n];
        seq[0]=0;
        seq[1]=1;
        seq[2]=1;
        for (int i=3;i<n;i++){
            seq[i]=seq[i-1]+seq[i-2]+seq[i-3];

        }
        System.out.println(seq[n-1]);
    }
}
```

input	Expected	Got
5	4	4
8	24	24
11	149	149

Passed all tests!

Ex.No. 2.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

You have recently seen a motivational sports movie and want to start exercising regularly. Your coach tells you that it is important to get up early in the morning to exercise. She sets up a schedule for you:
On weekdays (Monday - Friday), you have to get up at 5:00. On weekends (Saturday & Sunday), you can wake up at 6:00. However, if you are on vacation, then you can get up at 7:00 on weekdays and 9:00 on weekends.
Write a program to print the time you should get up.
Input Format
Input containing an integer and a boolean value.
The integer tells you the day it is (1-Sunday, 2-Monday, 3-Tuesday, 4-Wednesday, 5-Thursday, 6-Friday, 7-Saturday). The boolean is true if you are on vacation and false if you're not on vacation.
You have to print the time you should get up.
Example Input:
1 false
Output:
6:00
Example Input:
5 false
Output:
5:00
Example Input:
1 true
Output:
9:00

For example:

Input	Result
1 false	6:00
5 false	5:00
1 true	9:00

PROGRAM

```
import java.util.Scanner;
public class odd{
    public static void main(String[] args){
        Scanner  sc= new Scanner(System.in);
        int day=sc.nextint();
        boolean vac=sc.nextBoolean();
        String wakeup;
        if(vac){
            if(day==1||day==7){
                wakeup="9:00";
            }
            else{
                wakeup="7:00";
            }
        }
        else{
            if(day==1||day==7){
                wakeup="6:00";
            }
            else{
                wakeup="S:00";
            }
        }
        System.out.println(wakeup);
        }

    }
```

Input	Expected	Got
1 false	6:00	6:00
5 false	5:00	5:00
1 true	9:00	9:00

Passed all tests!

Ex.No. 2.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

You and your friend are movie fans and want to predict if the movie is going to be a hit!
The movie's success formula depends on 2 parameters:
the acting power of the actor (range Oto 10)
the critic's rating of the movie (range Oto 10)
The movie is a hit if the acting power is excellent (more than 8) or the rating is excellent (more than 8). This holds true except if either the acting power is poor (less than 2) or rating is poor (less than 2), then the movie is a flop. Otherwise the movie 1s average.
Write a program that takes 2 integers:
the first integer is the acting power
second integer is the critic's rating.
You have to print Yes if the movie is a hit, Maybe if the movie is average and No if the movie is flop.
Example input:
95
Output:
Yes
Example input:
1 9
Output:
No
Example input:
64
Output:
Maybe

For example:

Input	Result
95	Yes
1 9	No
64	Maybe

PROGRAM

```
import java.util.Scanner;
public class a{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int ap=sc.nextint();
        int cr=sc.nextint();
        if((cr>=8lap>=8) && (cr>2&ap>2)){
            System.out.println("Yes");
        }
        else if(ap<=2lcr<=2){
            System.out.println("No");
        }
        else{
            System.out.println("Maybe");
        }
        }
    }
```

	Input	Expected	Got	
✓	9 5	Yes	Yes	✓
✓	1 9	No	No	✓
✓	6 4	Maybe	Maybe	✓

Passed all tests! ✓

03-ARRAYS

Ex.No. **3.1** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Given an integer array as input, perform the following operations on the array, in the below specified sequence.

1. Find the maximum number in the array.
2. Subtract the maximum number from each element of the array.
3. Multiply the maximum number (found in step 1) to each element of the resultant array.

After the operations are done, return the resultant array.

Example 1:

inputl = 4 (represents the number of elements in the inputl array)

input2 = {1, 5, 6, 9}

Expected Output= {-72, -36, 27, 0}

Explanation:

Step 1: The maximum number in the given array is 9.

Step 2: Subtracting the maximum number 9 from each element of the array:

{(1 - 9), (5 - 9), (6 - 9), (9 - 9)} = {-8, -4, -3, 0}

Step 3: Multiplying the maximum number 9 to each of the resultant array:

{(-8 x 9), (-4 x 9), (3 x 9), (0 x 9)} = {-72, -36, -27, 0}

So, the expected output is the resultant array {-72, -36, -27, O}.

Example 2:

inputl = 5 (represents the number of elements in the inputl array)

input2 = {10, 87, 63, 42, 2}

Expected Output= {-6699, 0, -2088, -3915, -7395}

Explanation:

Step 1: The maximum number in the given array is 87.

Step 2: Subtracting the maximum number 87 from each element of the array:

{(10 - 87), (87 - 87), (63 - 87), (42 - 87), (2 - 87)} = {-77, 0, -24, -45, -85}

Step 3: Multiplying the maximum number 87 to each of the resultant array:

{(-77 x 87), (0 x 87), (-24 x 87), (-45 x 87), (-85 x 87)} = {-6699, 0, -2088, -3915, -7395}

So, the expected output is the resultant array {-6699, 0, -2088, -3915, -7395}.

Example 3:

inputl = 2 (represents the number of elements in the inputl array)

input2 = {-9, 9}

Expected Output= {-162, 0}

Explanation:

Step 1: The maximum number in the given array is 9.

Step 2: Subtracting the maximum number 9 from each element of the array:

{(-9 - 9), (9 - 9)} = {-18, O}

Step 3: Multiplying the maximum number 9 to each of the resultant array:
{(-18 x 9), (Ox 9)} = {-162, O}
So, the expected output is the resultant array {-162, O}.
Note: The input array will contain not more than 100 elements

For example:

Input	Result
4 1569	-72 -36 -27 0
5 10 87 63 42 2	-6699 0 -2088 -3915 -7395
2 -9 9	-162 0

PROGRAM

```
import java.util.Scanner;
public class name{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n= sc.nextint();
        int[] a= new int[n];
        for (int i=0;i<n;i++){
            a(i]=sc.nextint();
        }
    int max=a[0];
    for (int j=l;j<n;j++){
        if(max<a[j]){
            max=a[j];
        }
    }
    for(int k=0;k<n;k++){
        a[k]=a[k]-max;
    }
    for(int 1=0;l<n;l++){
        a[l]=max*a[l];
    }
    for(int k=0;k<n;k++){
        System.out.print(a[k]+"" );
    }
}
}
```

	Input	Expected	Got	
✓	4 1 5 6 9	-72 -36 -27 0	-72 -36 -27 0	✓
✓	5 10 87 63 42 2	-6699 0 -2088 -3915 -7395	-6699 0 -2088 -3915 -7395	✓
✓	2 -9 9	-162 0	-162 0	

Passed all tests! ✓

Ex.No. 3.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Given an array of numbers, you are expected to return the sum of the longest sequence of POSITIVE numbers in the array.

If there are NO positive numbers in the array, you are expected to return -1.

In this question's scope, the number O should be considered as positive.

Note: If there are more than one group of elements in the array having the longest sequence of POSITIVE numbers, you are expected to return the total sum of all those POSITIVE numbers (see example 3 below).

inputl represents the number of elements in the array.

input2 represents the array of integers.

Example 1:

inputl = 16

input2 ={-12, -16, 12, 18, 18, 14, -4, -12, -13, 32, 34, -5, 66, 78, 78, -79}

Expected output= 62

Explanation:

The input array contains four sequences of POSITIVE numbers, i.e. "12, 18, 18, 14", "12", "32, 34", and "66, 78, 78". The first sequence "12, 18, 18, 14" is the longest of the four as it contains 4 elements. Therefore, the expected output = sum of the longest sequence of POSITIVE numbers= 12 + 18 + 18 + 14 = 63.

Example 2:

inputl = 11

input2 ={-22, -24, 16, -1, -17, -19, -37, -25, -19, -93, -61}

Expected output = -1

Explanation:

There are NO positive numbers in the input array. Therefore, the expected output for such cases = -1.

Example 3:

inputl = 16

input2 ={-58, 32, 26, 92, -10, -4, 12, 0, 12, -2, 4, 32, -9, -7, 78, -79}

Expected output= 174

Explanation:

The input array contains four sequences of POSITIVE numbers, i.e. "32, 26, 92", "12, 0, 12", "4, 32", and "78". The first and second sequences "32, 26, 92" and "12, 0, 12"

are the longest of the four as they contain 4 elements each. Therefore, the expected output = sum of the longest sequence of POSITIVE numbers = (32 + 26 + 92) + (12 + 0 + 12) = 174.

For example:

Input	Result
16 -12 -16 12 18 18 14 -4 -12 -13 32 34 -5 66 78 78 -791	62
11 -22 -24 -16 -1 -17 -19 -37 -25 -19 -93 -61	-1
16 1-58 32 26 92 -10 -4 12 0 12 -2 4 32 -9 -7 78 -79	114

PROGRAM

```
import java.util.Scanner;

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

        int n  sc.nextint();
        int[] arr= new int[n];

        for (int i  0; i < n; i++) {
            arr[i] = sc.nextint();
        }

        int maxSum = -1;
        int currentSum = 0;
        int maxlength = 0;
        int currentlength = 0;

        for (inti= 0; i < n; i++) {
```

```
            if (arr[i] >= 0){
                currentSum += arr[i];
                currentLength++;
            } else {
                if (currentlength > maxlength) {
                    maxlength = currentlength;
                    maxSum = currentSum;
                } else if (currentlength == maxlength) {
                    maxSum += currentSum;
                }
                currentSum = 0;
                currentlength = 0;
            }
        }

        if (currentlength > maxlength) {
            maxSum = currentSum;
        } else if (currentlength == maxlength) {
            maxSum += currentSum;
        }

        if (maxSum == -1){
            System.out.println(-1);
        } else {
            System.out.println(maxSum);
        }

    }
}
```

	Input	Expected	Got	
✓	16 -12 -16 12 18 18 14 -4 -12 -13 32 34 -5 66 78 78 -79	62	62	✓
✓	11 -22 -24 -16 -1 -17 -19 -37 -25 -19 -93 -61	-1	-1	✓
✓	16 -58 32 26 92 -10 -4 12 0 12 -2 4 32 -9 -7 78 -79	174	174	✓

Passed all tests! ✓

Ex.No. 3.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Problem Statement:

You are provided with a set of numbers (array of numbers).

You have to generate the sum of specific numbers based on its position in the array set provided to you.

This is explained below:

Example 1:

Let us assume the encoded set of numbers given to you is:

inputl:5 and input2: {l, 51, 436, 7860, 41236}

Step 1:

Starting from the 0^{th} index of the array pick up digits as per below:

0^{th} index - pick up the units value of the number (in this case is 1).

1^{st} index - pick up the tens value of the number (in this case it is 5).

2^{nd} index - pick up the hundreds value of the number (in this case it is 4).

3^{rd} index - pick up the thousands value of the number (in this case it is 7).

4^{th} index - pick up the ten thousands value of the number (in this case it is 4).

(Continue this for all the elements of the input array).

The array generated from Step 1 will then be- {1, 5, 4, 7, 4}.

Step 2:

Square each number present in the array generated in Step 1.

{l, 25, 16, 49, 16}

Step 3:

Calculate the sum of all elements of the array generated in Step 2 to get the final result. The result will be= 107.

Note:

1) While picking up a number in Step1, if you observe that the number is smaller than the required position then use 0.

2) In the given function, inputlD is the array of numbers and input2 represents the number of elements in inputl.

Example 2:

inputl: 5 and inputl: {1, 5, 423, 310, 61540}

Step 1:

Generating the new array based on position, we get the below array:

{l, 0, 4, 0, 6}

In this case, the value in inputl at index 1 and 3 is less than the value required to be picked up based on position, so we use a 0.

Step 2:

{1, 0, 16, 0, 36}

Step 3:

The final result= 53.

For example:

Input	Result:
5 1 514367860 41236	107
5 1 514367860 41236	53

PROGRAM

```
import java.util.Scanner;

public class SumSpecificDigitsSimplified {
    public static void main(String[] args) {
        Scanner scanner= new Scanner(System.in);
        int n = scanner.nextint();
        int[] input= new int[n];

        for (inti= 0; i < n; i++) {
            input[i]  scanner.nextint();
        }
        int[] newArray = new int[n];
        for (inti= 0; i < n; i++) {
            String number= String.valueOf(input[i]);
            if (i < number.length()){
                newArray[i] =
Character.getNumericValue(number.charAt(number.length() - 1  - i));
            } else {
                newArray[i] = 0;
            }
        }
        for (inti= 0; i < n; i++) {
            newArray[i]   newArray[i] * newArray[i];
        }
        int sum= 0;
        for (int num : newArray) {
            sum+= num;
        }
        System.out.println(sum);
        scanner.close();
    }
```

Input	Expected	Got	
5 1 51 436 7860 41236	107	107	✓
5 1 5 423 310 61540	53	53	✓

Passed all tests!

}

04 - CLASSES AND OBJECTS

Ex.No. 4.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Create a class called "Circle" with a radius attribute. You can access and modify this attribute using getter and setter methods. Calculate the area and circumference of the circle.

Area of Circle = nr^2
Circumference = 2nr
Input:
2
Output:
Area= 12.57
Circumference= 12.57
For example:

Test	Input	Result
1	4	Area= 50.27 Circumference = 25.13

PROGRAM

```
import java.io.*;
import java.util.*;
class Circle
{
    private double radius;
    public Circle(double radius){
        this.radius=radius;


    }
    public void setRadius(double radius){
        this.radius=radius;


    }
    public double getRadius()     {
        // return the radius
        return this.radius;

    }
    public double calculateArea() {//complete the below statement
       return Math.PI*radius*radius;

    }
```

```
    public double calculateCircumference()      {
        // complete the statement
       return 2*Math.PI*radius;
    }
}
class prog{
    public static void main(String[] args)  {
        int r;
        Scanner sc= new Scanner(System.in);
        r=sc.nextint();
        Circle c= new Circle(r);
        System.out.println("Area = "+String.format("%.2f",
c.calculateArea()));
        // invoke the calculatecircumference method
        System.out.println("Circumference =
"+String.format("%.2f",c.calculateCircumference()));

    }
}
```

	Test	Input	Expected	Got	
✓	1	4	Area= 50.27 Circumference= 25.13	Area = 50.27 Circumference 25.13	✓
✓	2	6	Area= 113.10 Circumference 37.70	Area = 113.10 Circumference 37.70	✓
✓	3	2	Area= 12.57 Circumference 12.57	Area= 12.57 Circumference 12.57	✓

Passed all tests! ✓

Ex.No. 4.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Create a Class Mobile with the attributes listed below,
private String manufacturer;
private String operating_system;
public String color;
private int cost;
Define a Parameterized constructor to initialize the above instance variables.
Define getter and setter methods for the attributes above.
for example: setter method for manufacturer is
void setManufacturer(String manufacturer){
this.manufacturer= manufacturer;
}
String getManufacturerO{
return manufacturer;}
Display the object details by overriding the toStringO method.

For example:

Test	Result
1	manufacturer= Redmi operating_system = Andriod color= Blue cost = 34000

PROGRAM

```
class Mobile {
    private String manufacturer;
    private String operating_system;
    public String color;
    private int cost;

    // Parameterized constructor to initialize the attributes
    public Mobile(String manufacturer, String operating_system, String color,
int cost) {
        this.manufacturer= manufacturer;
        this.operating_system = operating_system;
        this.color= color;
        this.cost= cost;
    }

    // Getter and Setter methods for manufacturer
```

```
    public void setManufacturer(String manufacturer) {
        this.manufacturer= manufacturer;
    }

    public String getManufacturer() {
        return manufacturer;
    }

    // Getter and Setter methods for operating_system
    public void setOperatingSystem(String operating_system) {
        this.operating_system = operating_system;
    }

    public String getOperatingSystem() {
        return operating_system;
    }

    // Getter and Setter methods for color
    public void setColor(String color) {
        this.color= color;
    }

    public String getColor() {
        return color;
    }

    // Getter and Setter methods for cost
    public void setCost(int cost) {
        this.cost= cost;
    }

    public int getCost() {
        return cost;
    }

    // Overriding the toString() method to display object details
    @Override
    public String toString() {
        return "manufacturer="+ manufacturer+ "\n" +
               "operating_system = "+ operating_system + "\n" +
               "color="+ color+ "\n" +
               "cost="+ cost;
    }
}
public class prog{
    public static void main(String[] args) {
        // Creating a Mobile object with the given attributes
        Mobile mobile= new Mobile("Redmi", "Andriod", "Blue", 34000);
```

```
        // Display the object details
        System.out.println(mobile);
    }
}
```

Test	Expected	Got
1	manufacturer= Redmi operating_system = Andriod color= Blue cost= 34000	manufacturer = Redmi operating_system = Andriod color'"' Blue cost = 34000

Passed all tests!

Ex.No. 4.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Create a class Student with two private attributes, name and roll number. Create three objects by invoking different constructors available in the class Student.
Student◊
Student(String name)
Student(String name, int rollno)

Input:
No input

Output:
No-arg constructor is invoked
1 arg constructor is invoked
2 arg constructor is invoked
Name =null, Roll no= 0
Name =Rajalakshmi, Roll no= 0
Name =Lakshmi, Roll no= 101

For example:

Test	Result
1	No-arg constructor is invoked 1 arg constructor is invoked 2 arg constructor is invoked Name =null , Roll no = 0 Name =Rajalakshmi, Roll no= 0 Name =Lakshmi , Roll no= 101

PROGRAM

```
class Student {
    private String name;
    private int rollno;
    public Student(){
        System.out.println("No-arg constructor is invoked");
        this.name= "null";
        this.rollno = 0;
    }
    public Student(String name){
        System.out.println("1 arg constructor is invoked");
        this.name= name;
        this.rollno = 0;
    }
    public Student(String name, int rollno) {
```

```
        System.out.println("2 arg constructor is invoked");
        this.name= name;
        this.rollno = rollno;
    }
    public void display(){
        System.out.println("Name  "+name+" , Roll no   " + rollno);

    }
}
class prog{
    public static void main(String[] args)
        Student sl = new Student();
        IIs1.display();
        Student s2 = new Student("Rajalakshmi");
        //s2.display()
        Student s3 = new Student("Lakshmi", 101);
        //s3.display();
        Sl.display();
        s2.display();
        s3.display();

    }

}
```

Test	Expected	Got
1	**No-arg** constructor **is invoked** **1 arg** constructor **is invoked** 2 arg constructor **is invoked** **Name** •null, Roll no• **0** **Name** •**Rajalakshmi**, Roll no• **0** Name •Lakshmi, Roll no• 101	**No-arg** constructor **is invoked** **1 arg** constructor **is invoked** 2 arg constructor **is invoked** **Name** •null, Roll no• 0 **Name** •**Rajalakshmi**, Roll no• 0 Name •Lakshmi, Roll no• 101

Passed all tests!

05 - INHERITANCE

Ex.No. **5.1** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

create a class called College with attribute String name, constructor to initialize the name attribute , a method called Admitted◊. Create a subclass called CSE that extends Student class, with department attribute , Course◊ method to sub class. Print the details of the Student.

College:
String collegeName;
public College◊ {}
public admitted◊ {}
Student:
String studentName;
String department;
public Student(String collegeName, String studentName,String depart) {}
public toString()
Expected Output:
A student admitted in REC
CollegeName : REC
StudentName: Venkatesh
Department : CSE
For example:

Result
A student admitted in REC CollegeName : REC StudentName: Venkatesh Department : CSE

PROGRAM

```
class College
{
public  String collegeName;

public College(String collegeName) {

    this.collegeName=collegeName;
    }

public void admitted(){
    System.out.println("A student admitted in "+collegeName);
}
}
class Student extends College{
```

```
String studentName;
String department;

public Student(String collegeName, String studentName,String department) {

   super(collegeName);
   this.studentName=studentName;
   this.department=department;

}

public String toString(){

    return "CollegeName : "+collegeName+"\n"+"StudentName
"+studentName+"\n"+"Department : "+department;
}
}
public class Main {
public static void main (String[] args) {
        Student  s1 = new Student("REC","Venkatesh","CSE");
         s1.admitted();
        System.out.println(s1.toString());
}
}
```

Expected	Got
✓ A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE	A student admitted in REC ✓ CollegeName : REC StudentName : Venkatesh Department : CSE

Passed all tests!

Ex.No. 5.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Create a class known as "BankAccount" with methods called deposit◊ and withdrawO.
Create a subclass called SavingsAccount that overrides the withdraw◊ method to prevent withdrawals if the account balance falls below one hundred.
For example:

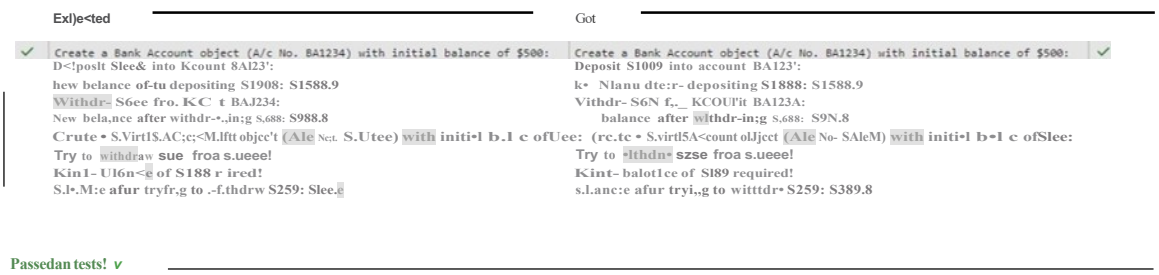
Result
Create a Bank Account object (Ale No. BA1234) with initial balance of $500: Deposit $1000 into account BA1234: New balance after depositing $1000: $1500.0 Withdraw $600 from account BA1234: New balance after withdrawing $600: $900.0 Create a SavingsAccount object *(Ale* No. SAl000) with initial balance of $300: Try to withdraw $250 from SAlO00! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0

PROGRAM

```
class BankAccount {
    // Private field to store the account number
    private String accountNumber;

    // Private field to store the balance
    private double balance;

    // Constructor to initialize account number and balance
    public BankAccount(String accountNumber,double balance){
        this.accountNumber=accountNumber;
        this.balance=balance;
    }




    // Method to deposit an amount into the account
    public void deposit(double amount) {
        // Increase the balance by the deposit amount
     balance+=amount;
    }
```

```
    // Method to withdraw an amount from the account
    public void withdraw(double amount) {
        // Check if the balance is sufficient for the withdrawal
        if (balance>= amount) {
            // Decrease the balance by the withdrawal amount
            balance-= amount;
        } else {
            // Print a message if the balance is insufficient
            System.out.println("Insufficient balance");
        }
    }

    // Method to get the current balance
    public double getBalance(){
        // Return the current balance
        return balance;
    }
    public String getAccountNumber(){
        return accountNumber;
    }
}
class SavingsAccount extends BankAccount {
    // Constructor to initialize account number and balance
    public SavingsAccount(String accountNumber, double balance) {
        // Call the parent class constructor
        super(accountNumber,balance);
    }

    // Override the withdraw method from the parent class
    @Override
    public void withdraw(double amount) {
        // Check if the withdrawal would cause the balance to drop below $100
        if (getBalance() - amount< 100){
            // Print a message if the minimum balance requirement is not met
            System.out.println("Minimum balance of $100 required!");
        } else {
            // Call the parent class withdraw method
            super.withdraw(amount);
        }
    }
}

public class Main {

    public static void main(String[] args) {
        // Print message to indicate creation of a BankAccount object
        System.out.println("Create a Bank Account object (A/c No. BA1234) with
initial balance of $500:");
```

```
        // Create a BankAccount object (A/c No. "BA1234") with initial balance
of $500
        BankAccount BA1234 = new BankAccount("BA1234", 500);
        // Print message to indicate deposit action
        System.out.println("Deposit $1000 into account BA1234:");
        // Deposit $1000 into account BA1234
       BA1234.deposit(1000);
        // Print the new balance after deposit
       System.out.println("New balance after depositing $1000:
$"+BA1234.getBalance());
        // Print message to indicate withdrawal action
        System.out.println("Withdraw $600 from account BA1234:");
        // Withdraw $600 from account BA1234
      BA1234.withdraw(600);
        // Print the new balance after withdrawal
        System.out.println("New balance after withdrawing $600: $" +
BA1234.getBalance());
        // Print message to indicate creation of another SavingsAccount object
        System.out.println("Create a SavingsAccount object (A/c No. SA1000)
with initial balance of $300:");
        // Create a SavingsAccount object (A/c No. "SA1000") with initial
balance of $300
        SavingsAccount SA1000 = new SavingsAccount("SA1000", 300);

        // Print message to indicate withdrawal action
        System.out.println("Try to withdraw $250 from SA1000!");
        // Withdraw $250 from SA1000 (balance falls below $100)
        SA1000.withdraw(250);
        // Print the balance after attempting to withdraw $250
        System.out.println("Balance after trying to withdraw $250: $" +
SA1000.getBalance());
    }
}
```



Ex.No. 5.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Create a class Mobile with constructor and a method basicMobileO.

Create a subclass CameraMobile which extends Mobile class, with constructor and a method newFeatureO.

Create a subclass AndroidMobile which extends CameraMobile, with constructor and a method androidMobileO.

display the details of the Android Mobile class by creating the instance.

```
class Mobile{

}
class CameraMobile extends Mobile {

}

class AndroidMobile extends CameraMobile {

}
```

expected output:

```
Basic Mobile is Manufactured
Camera Mobile is Manufactured
Android Mobile is Manufactured
Camera Mobile with 5MG px
Touch Screen Mobile is Manufactured
```

For example:

Result
Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured

PROGRAM

```
class mob{
    mob(){
        System.out.println("Basic Mobile is Manufactured");
    }
    void basmob(){
        System.out.println("Basic Mobile is Manufactured");
    }
}
class cam extends mob{
    cam(){
        super();
        System.out.println("Camera Mobile is Manufactured");
    }
    void newm(){
        System.out.println("Camera Mobile with SMG px");

    }
}
class and extends cam{
    and(){
    super();
    System.out.println("Android Mobile is Manufactured");
    }
    void andmob(){
         System.out.println("Touch Screen Mobile is Manufactured");
    }
    }
public class Main{
    public static void main(String[]args){
        and andmob=new and();
        andmob.newm();
        andmob.andmob();
    }

}
```

	Expected	Got	
../	Basic Mobile is Manufactured	Basic Mobile is Manufactured	../
	Camera Mobile is Manufactured	Camera Mobile is Manufactured	
	Android Mobile is Manufactured	Android Mobile is Manufactured	
	Camera Mobile 1 ith SMG px	Camera Mobile with SMG px	
	Touch Screen Mobile is Manufactured	Touch Screen Mobile is tlanufactured	

Passed all tests!

06 - **STRING, STRINGBUFFER**

Ex.No. 6.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

You are provided a string of words and a 2-digit number. The two digits of the number represent the two words that are to be processed.

For example:

If the string is "Today is a Nice Day" and the 2-digit number is 41, then you are expected to process the 4th word ("Nice") and the 1st word ("Today").

The processing of each word is to be done as follows:

Extract the Middle-to-Begin part: Starting from the middle of the word, extract the characters till the beginning of the word.

Extract the Middle-to-End part: Starting from the middle of the word, extract the characters till the end of the word.

If the word to be processed is "Nice":

Its Middle-to-Begin part will be **"iN".**

Its Middle-to-End part will be "ce".

So, merged together these two parts would form "iNce".

Similarly, if the word to be processed is "Today":

Its Middle-to-Begin part will be "doT".

Its Middle-to-End part will be "day".

So, merged together these two parts would form "doTday".

Note: Note that the middle letter 'd' is part of both the extracted parts. So, for words whose length is odd, the middle letter should be included in both the extracted parts.

Expected output:

The expected output is a string containing both the processed words separated by a space "iNce doTday"

Example 1:

inputl ="Today is a Nice Day"

input2 = 41

output= "iNce doTday"

Example 2:

inputl ="Fruits like Mango and Apple are common but Grapes are rare"

input2 = 39

output= "naMngo arGpes"

Note: The input string inputl will contain only alphabets and a single space character separating each word in the string.

Note: The input string inputl will NOT contain any other special characters.

Note: The input number input2 will always be a 2-digit number (>=11 and <=99). One of its digits will never be 0. Both the digits of the number will always point to a valid word in the inputl string.

For example:

Input	Result
Today is a Nice Day 41	iNce doTday
Fruits like Mango and Apple are common but Grapes are rare 39	naMngo arGpes

PROGRAM

```
import java.util.*;
public class mix{
    public static void main(String[] args){
        Scanner scan= new Scanner(System.in);
        String g = scan.nextline();
        int n = scan.nextint(),ones,flag = 0;
        StringBuffer temp= new StringBuffer();
        StringBuffer templ = new StringBuffer();
        int space= 0;
        while (n > 0){
              ones= (n %10) - 1;
              for(int i = 0; i < g.length();i++){
                  if (g.charAt(i) == ' '){
                      space= space+ 1;
                  }
                  else if(space == ones && flag== 0){
                        temp.append(Character.toString(g.charAt(i)));
                  }
                  else if(space == ones && flag== 1){
                        templ.append(Character.toString(g.charAt(i)));
                  }
              }
              space= O;
              flag= 1;
              n = n /10;
        }
        rew m  new rew();
        System.out.println(m.r(templ.toString()) + " " +
m.r(temp.toString()));
    }
}
class rew{
    String r(String a){
        int le= a.length(),n,q;
        StringBuffer temp3 = new StringBuffer();
        if(le % 2 == 1){
```

```
            n   ((int)(le/2));
            q = ((int)(le/2));
        }
        else{
            n = ((int)(le/2)) - 1;
            q = ((int)(le/2));
        }
        for(int i  = n;i >= 0;i--){
                temp3.append(Character.toString(a.charAt(i)));
            }
        for(int i  = q;i < le;i++){
            temp3.append(Character.toString(a.charAt(i)));
        }
        return temp3.toString();
    }
}
```

	Input	Expected	Got	
✓	Today is a Nice Day 41	iNce doTday	iHce doTday	✓
	Fruits like Mango and Apple are co,,,.110n but Grapes are rare nallngo arGpes na/-lngo arGpes 39			

Passed all tests!

Ex.No. 6.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Given 2 strings inputl & input2.
Concatenate both the strings.
Remove duplicate alphabets & white spaces.
Arrange the alphabets in descending order.
Assumption 1:
There will either be alphabets, white spaces or null in both the inputs.
Assumption 2:
Both inputs will be in lower case.
Example 1:
Input 1: apple
Input 2: orange
Output: rponlgea
Example 2:
Input 1: fruits
Input 2: are good
Output: utsroigfeda
Example 3:
Input 1: ""
Input 2: ""
Output: null
For example:

Test	Input	Result
1	apple orange	rponlgea
2	fruits are good	utsroigfeda

PROGRAM

```
import java.util.*;


public class HelloWorld {
    public static void main(String[] args) {
        Scanner scan= new Scanner(System.in);
        String a= scan.nextline();
        String b = scan.nextline();
        StringBuffer ab= new StringBuffer();
```

```
        if(a.trim().isEmpty() && b.trim().isEmpty()){
            System.out.print("null");
        }
        else{
        for(int i = 0;i < a.length();i++){
            if (a.charAt(i) != ' ') {
                ab.append(Character.toString(a.charAt(i)));
            }
        }
        for(int i = 0;i < b.length();i++){
            if (b.charAt(i) != ' '){
                ab.append(Character.toString(b.charAt(i)));
            }
        }
        char[] d = ab.toString().toCharArray();
        Arrays.sort(d);
        for(int i = d.length - 1;i >= 1;i--){
            if(d[i] != d[i-1])
            System.out.print(d[i]);
        }
        System.out.print(d[0]);
        }

    }
}
```

	Test	Input	Expected	Got	
V	1	apple orange	rponlgea	rponlgea	V
V	2	fruits are good	utsroigfeda	utsroigfeda	V
V	3		null	null	

Passed all tests! v

Ex.No. 6.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Given a String input1, which contains many number of words separated by : and each word contains exactly two lower case alphabets, generate an output based upon the below 2 cases.

Note:

1. All the characters in input 1 are lowercase alphabets.
2. input 1 will always contain more than one word separated by:
3. Output should be returned in uppercase.

Case 1:

Check whether the two alphabets are same.

If yes, then take one alphabet from it and add it to the output.

Example 1:

inputl = ww:ii:pp:rr:oo

output= WIPRO

Explanation:

wordl is ww, both are same hence take w

word2 is ii, both are same hence take i

word3 is pp, both are same hence take p

word4 is rr, both are same hence taker

word5 is oo, both are same hence take o

Hence the output is WIPRO

Case 2:

If the two alphabets are not same, then find the position value of them and find maximum value - minimum value.

Take the alphabet which comes at this (maximum value - minimum value) position in the alphabet series.

Example 2"

inputl = zx:za:ee

output= BYE

Explanation

wordl is zx, both are not same alphabets

position value of z is 26

position value of xis 24

max - min will be 26 - 24 = 2

Alphabet which comes in 2nd position is b

Word2 is za, both are not same alphabets

position value of z is 26

position value of a is 1

max - min will be 26- 1 = 25

Alphabet which comes in 25th position is y

word3 is ee, both are same hence take e

Hence the output is BYE

For example:

Input	Result
ww:n:pp:rr:oo	WIPRO
zx:za:ee	BYE

PROGRAM

```
#include<stdio.h>
import java.util.*;
class diff{
    char different(char a, char b){
        if ((int)a != (int)b)
            return (char)((int)'a' + ((int)a-(int)b) - 1);
        return a;
        }
}
public class Main{
    public static void main(String[] args){
        Scanner scan= new Scanner(System.in);
        diff z = new diff();
        String q = scan.nextline();
        StringBuffer ans= new StringBuffer();
        StringBuffer temp= new StringBuffer();
        for(int i = 0;i < q.length();i++){
            if(q.charAt(i) == ':'){
                temp.append("");
            }
            else{
                temp.append(Character.toString(q.charAt(i)));}}
        String h = temp.toString();
        for(int i = 0;i < temp.length();i++){
            if(i%3 == 0){
                ans.append(Character.toString(z.different(h.charAt(i),h.charAt
(i+1))));}}
        System.out.print(ans.toString().toUpperCase());}}
```

Input	Expected	Got
""":ii: pp:rr: oo	l,JIPRO	HIPRO
zx:za:ee	BYE	BYE

Passed all tests!

07-INTERFACES

Ex.No. 7.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Create interfaces shown below.

```
 interface Sports {
public void setHomeTeam(String name);
public void setVisitingTeam(String name);
}
 interface Football extends Sports {
public void homeTeamScored(int points);
public void visitingTeamScored(int points);}
```

create a class College that implements the Football interface and provides the necessary functionality to the abstract methods.

sample Input:

Rajalakshmi
Saveetha
22
21

Output:

Rajalakshmi 22 scored
Saveetha **21** scored
Rajalakshmi is the Winner!

For example:

Test	Input	Result
1	Rajalakshmi Saveetha 22 **21**	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner!

PROGRAM

```
import java.util.Scanner;

interface Sports {
    void setHomeTeam(String name);
    void setVisitingTeam(String name);
}

interface Football extends Sports {
    void homeTeamScored(int points);
    void visitingTeamScored(int points);
}
```

```
class College implements Football {
    private String homeTeam;
    private String visitingTeam;
    private int homeTeamPoints = 0,·
    private int visitingTeamPoints = 0,·

    public void setHomeTeam(String name) {
        this.homeTeam = name;
    }

    public void setVisitingTeam(String name) {
        this.visitingTeam = name;
    }

    public void homeTeamScored(int points) {
        homeTeamPoints += points;
        System.out.println(homeTeam +" "+points+" scored");
    }

    public void visitingTeamScored(int points) {
        visitingTeamPoints += points;
        System.out.println(visitingTeam +" "+points+" scored");
    }

    public void winningTeam() {
        if (homeTeamPoints > visitingTeamPoints) {
            System.out.println(homeTeam +" is the winner!");
        } else if (homeTeamPoints < visitingTeamPoints) {
            System.out.println(visitingTeam + ''  is the winner!");
        } else {
            System.out.println("It's a tie match.");
        }
    }
}

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

        // Get home team name
        String hname = sc.nextline();

        // Get visiting team name
        String vteam = sc.nextline();

        // Create College object
        College match= new College();
        match.setHomeTeam(hname);
```

```
        match.setVisitingTeam(vteam);

        // Get points scored by home team
        int htpoints = sc.nextint();
        match.homeTeamScored(htpoints);

        // Get points scored by visiting team
        int vtpoints = sc.nextint();
        match.visitingTeamScored(vtpoints);

        // Determine and print the winning team
        match.winningTeam();

        sc.close();
    }
}
```

	Test	Input	Expected	Got	
✓	1	Rajalaksh111i Saveetha 22 21	Rajalaksh111i 22 scored Saveetha 21 scored Rajalaksh111i is the winner!	Rajalaksh111i 22 scored Saveetha 21 scored Rajalaksh111i is the winner!	✓
✓	2	Anna Balaji 21 21	Anna 21 scored Balaji 21 scored It's a tie match.	Anna 21 scored Balaji 21 scored It's a tie match.	✓
✓	3	SRM VIT 20 21	SRl-1 20 scored VIT 21 scored VIT is the winner!	SRM 20 scored VIT 21 scored VIT is the winner!	✓

Passed all tests!

Ex.No. 7.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

create an interface Playable with a method play0 that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the playO method to play the respective sports.

```
interface Playable {
   void play0;
}
class Football implements Playable {
   String name;
   public Football(String name){
      this.name:::name;
   }
  public void playO {
   System.out.println(name+" is Playing football");
  }
}
```

Similarly, create Volleyball and Basketball classes.

Sample output:

Sadhvin is Playing football
Sanjay is Playing volleyball
Sruthi is Playing basketball

For example:

Test	Input	Result
1	Sadhvin Sanjay Sruthi	Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball
2	Vijay Arun Balaji	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball

PROGRAM

```
import java.util.Scanner;

// Define the Playable interface
interface Playable {
    // Abstract method to play the respective sport
    void play();
}

// Football class implementing Playable interface
class Football implements Playable {
```

```
    String name;

    // Constructor
    public Football(String name){
        this.name= name;
    }

    // Override the play method
    public void play(){
        System.out.println(name +" is Playing football");
    }
}

// Volleyball class implementing Playable interface
class Volleyball implements Playable {
    String name;

    // Constructor
    public Volleyball(String name) {
        this.name= name;
    }

    // Override the play method
    public void play(){
        System.out.println(name +" is Playing volleyball");
    }
}

// Basketball class implementing Playable interface
class Basketball implements Playable {
    String name;

    // Constructor
    public Basketball(String name) {
        this.name= name;
    }

    // Override the play method
    public void play(){
        System.out.println(name +" is Playing basketball");
    }
}

// Main class to test the functionality
public class Main {
    public static void main(String[] args) {
        Scanner scanner= new Scanner(System.in);
```

```
        // Input for Football player

        String footballPlayerName = scanner.nextline();
        Football footballPlayer   new Football(footballPlayerName);

        // Input for Volleyball player

        String volleyballPlayerName = scanner.nextline();
        Volleyball volleyballPlayer = new Volleyball(volleyballPlayerName);

        // Input for Basketball player

        String basketballPlayerName   scanner.nextline();
        Basketball basketballPlayer   new Basketball(basketballPlayerName);

        // Call the play method for each player
        footballPlayer.play();
        volleyballPlayer.play();
        basketballPlayer.play();


        scanner.close();
    }
}
```

	Test	Input	Expected	Got	
✓	1	**Sadhvin**	**Sadhvin is Playing football**	**Sadhvin is Playing football**	✓
		Sanjay	Sanjay is Playing volleyball	Sanjay is Playing volleyball	
		Sruthi	Sruthi is Playing basketball	Sruthi is Playing basketball	
✓	2	Vijay	Vijay is Playing football	Vijay is Playing football	✓
		Arun	Arun is Playing volleyball	Arun is Playing volleyball	
		Balaji	Salaji is Playing basketball	Salaji is Playing basketball	

Passed all tests! ✓

Ex.No. 7.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

RBI issues all national banks to collect interest on all customer loans.
Create an RBI interface with a variable String parentBank="RBI" and abstract method rateOflnterestO.
RBI interface has two more methods default and static method.
default void policyNote0 {
System.out.println("RBI has a new Policy issued in 2023.");
}
static void regulationsO{
System.out.println("RBI has updated new regulations on 2024.");
}
Create two subclasses SBI and Karur which implements the RBI interface.
Provide the necessary code for the abstract method in two sub-classes.
Sample Input/Output:
RBI has a new Policy issued in 2023
RBI has updated new regulations in 2024.
SBI rate of interest: 7.6 per annum.
Karur rate of interest: 7.4 per annum.
For example:

Test	**Result**
1	RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum.

PROGRAM

```
interface RBI {
    II Variable declaration
    String parentBank = "RBI";

    II Abstract method
    double rateOfinterest();

    II Default method
    default void policyNote() {
        System.out.println("RBI has a new Policy issued in 2023");
    }

    II Static method
    static void regulations() {
        System.out.println("RBI has updated new regulations in 2024.");
    }
```

```
}

// SBI class implementing RBI interface
class SBI implements RBI {
    // Implementing the abstract method
    public double rateOfinterest() {
        return 7.6;
    }
}

// Karur class implementing RBI interface
class Karur implements RBI {
    // Implementing the abstract method
    public double rateOfinterest() {
        return 7.4;
    }
}

// Main class to test the functionality
public class Main {
    public static void main(String[] args) {
        // RBI policies and regulations
        RBI rbi = new SBI();  // Can be any class implementing RBI
        rbi.policyNote();    // Default method
        RBI.regulations();  // Static method

        // SBI bank details
        SBI sbi = new SBI();
        System.out.println("SBI rate of interest: " + sbi.rateOfinterest() + "
per annum.");

        // Karur bank details
        Karur karur = new Karur();
        System.out.println("Karur rate of interest: " + karur.rateOfinterest()
+   per annum.");
    }
}
```

Test	Expected	Got
1	RBI has ıı ne,; Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. K11rur rate of interest: 7.4 per annum.	RBI has ıı new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum.

Passed all tests!

08 - POLYMORPHISM, ABSTRACT CLASSES, FINAL KEYWORD

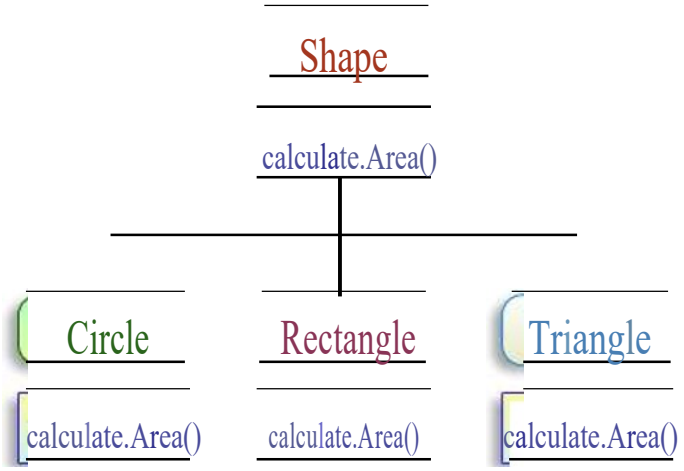
Ex.No. 8.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Create a base class Shape with a method called calculateArea0. Create three subclasses: Circle, Rectangle, and Triangle. Override the calculateArea0 method in each subclass to calculate and return the shape's area.

In the given exercise, here is a simple diagram illustrating polymorphism implementation:




```
abstract class Shape {
   public abstract double calculateArea0 ;
   }

}
```

System.out.printf("Area of a Triangle :%.2f0/on",((0.5)*base*height)); *II* use this statement

sample Input :

4 *II* radius of the circle to calculate area Pl*r*r

5 *II* length of the rectangle

6 *II* breadth of the rectangle to calculate the area of a rectangle

4 *II* base of the triangle

3 *II* height of the triangle

OUTPUT:

Area of a circle :50.27
Area of a Rectangle :30.00
Area of a Triangle :6.00

For example:

Test	Input	Result
1	4 5 6 4 3	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00
2	7 4.5 6.5 2.4 3.6	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32

PROGRAM

```
import java.util.Scanner;

// Abstract class Shape
abstract class Shape {
    public abstract double calculateArea();
}

// Circle class
class Circle extends Shape {
    private double radius;

    public Circle(double radius){
        this.radius= radius;
    }

    @Override
    public double calculateArea() {
        return Math.PI* radius* radius; // Area of circle: rrr²
    }
}

// Rectangle class
```

```
class Rectangle extends Shape {
    private double length;
    private double breadth;

    public Rectangle(double length, double breadth) {
        this.length= length;
        this.breadth= breadth;
    }

    @Override
    public double calculateArea() {
        return length* breadth; // Area of rectangle: length* breadth
    }
}

// Triangle class
class Triangle extends Shape {
    private double base;
    private double height;

    public Triangle(double base, double height) {
        this.base= base;
        this.height= height;
    }

    @Override
    public double calculateArea() {
        return 0.5 *base* height;// Area of triangle: 0.5 *base* height
    }
}

// Main class to test the shapes
public class ShapeTest {
    public static void main(String[] args) {
        Scanner scanner= new Scanner(System.in);

        // Input for Circle

        double radius= scanner.nextDouble();
        Circle circle= new Circle(radius);
        System.out.printf("Area of a circle: %.2f%n'', circle.calculateArea());

        // Input for Rectangle

        double length= scanner.nextDouble();

        double breadth= scanner.nextDouble();
        Rectangle rectangle= new Rectangle(length, breadth);
```

```
        System.out.printf("Area of a Rectangle: %.2f%n",
rectangle.calculateArea());

        // Input for Triangle

        double base= scanner.nextDouble();

        double height= scanner.nextDouble();
        Triangle triangle= new Triangle(base, height);
        System.out.printf("Area of a Triangle: %.2f%n",
triangle.calculateArea());

        scanner.close();
    }
}
```

	Test	Input	Expected	Got	
✓	1	4 5 6 4 3	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00	✓
✓	2	7 4.5 6.5 2.4 3.6	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32	✓

Passed all tests! ✓

Ex.No. 8.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

As a logic building learner you are given the task to extract the string which has vowel as the first and last characters from the given array of Strings.

Step1: Scan through the array of Strings, extract the Strings with first and last characters as vowels; these strings should be concatenated.

Step2: Convert the concatenated string to lowercase and return it.

If none of the strings in the array has first and last character as vowel, then return no matches found

inputl: an integer representing the number of elements in the array.

input2: String array.

Example 1:

inputl: 3

input2: {"oreo", "sirish", "apple"}

output: oreoapple

Example 2:

inputl: 2

input2: {"Mango", "banana"}

output: no matches found

Explanation:

None of the strings has first and last character as vowel.

Hence the output is no matches found.

Example 3:

inputl: 3

input2: {"Ate", "Ace", "Girl"}

output: ateace

For example:

Input	Result
3 oreo sirish apple	oreoapple
2 Mango banana	no matches found
3 Ate Ace Girl	ateace

PROGRAM

```
import java.util.Scanner;

public class VowelStringExtractor {

    // Method to extract strings with vowels as first and last characters
    public static String extractVowelStrings(String[] stringArray) {
        StringBuilder result= new StringBuilder();
        String vowels= "aeiouAEIOU''; // String containing all vowels

        // Iterate through the array of strings
        for (Strings : stringArray) {
            // Check if the string is not empty and if both the first and last
characters are vowels
            if (s.length() > 0 && vowels.indexOf(s.charAt(0)) != -1 &&
vowels.indexOf(s.charAt(s.length() - 1)) != -1) {
                result.append(s); // Append matching string to the result
            }
        }

        // Return the concatenated string in lowercase or "no matches found"
        return result.length()> 0? result.toString().tolowerCase() : "no
matches found";
    }

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

```
        // Input for the number of strings

        int n = scanner.nextint();
        scanner.nextline(); // Consume the newline character

        // Input for the strings in one line

        String input= scanner.nextline();
        String[] strings= input.split(" "); // Split input into an array

        // Process and output the result
        String result= extractVowelStrings(strings);
        System.out.println(result);

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

	Input	Expected	Got	
.../	3 oreo sirish apple	oreoapple	oreoapple	.../
.../	2 Mango banana	no matches found	no matches found	.../
.../	3 Ate Ace Girl	ateace	ateace	

Passed all tests! .../

Ex.No. **8.3** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

1. Final Variable:

- Once a variable is declared final, its value cannot be changed after it is initialized.
- It must be initialized when it is declared or in the constructor if it's not initialized at declaration.
- It can be used to define constants

```
final int MAX_SPEED = 120; // Constant value, cannot be changed
```

2. Final Method:

- A method declared final cannot be overridden by subclasses.
- It is used to prevent modification of the method's behavior in derived classes.

```
public final void display() {
   System.out.println("This is a final method.");
```

3. Final Class:

- A class declared as final cannot be subclassed (i.e., no other class can inherit from it).
- It is used to prevent a class from being extended and modified.
- public final class Vehicle {
  // class code

Given a Java Program that contains the bug in it, your task is to clear the bug to the output.

you should delete any piece of code.

For example:

Test	Result
1	The maximum speed is: 120 km/h This is a subclass of FinalExample.

PROGRAM

```
final class FinalExample {
    // Final variable
    final int MAX SPEED= 120; // Constant value

    // Final method
    public final void display() {
        System.out.println("The maximum speed is: "+ MAX_SPEED +" km/h");
    }
}

// Main class to test the final class
public class Test {
    public static void main(String[] args) {
        // Create an instance of FinalExample
        FinalExample example= new FinalExample();
        example.display();

        // Uncommenting the following line will result in a compile-time error
        // because FinalExample is a final class and cannot be subclassed.
        // class SubclassExample extends FinalExample {}

        System.out.println("This is a subclass of FinalExample.");
    }
}
```

Test	Expected	Got
	The maximum speed is: 120 km/h **This is a subclass of FinalExample.**	The maximum speed is: 120 km/h This is a subclass of FinalExample.

Passed all tests!

09 - EXCEPTION HANDLING

Ex.No. 9.1 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Write a Java program to create a method that takes an integer as a parameter and throws an exception if the number is odd.

Sample input and Output:

82 is even.

Error: 37 is odd.

Fill the preloaded answer to get the expected output.

For example:

Result
82 is even. Error: 37 is odd.

PROGRAM

```
class prog {
    public static void main(String[] args) {
      int n = 82;
      trynumber(n);
      n = 37;
      // call the trynumber(n);
      trynumber(n);

    }

    public static void trynumber(int n) {
      try {
          //call the checkEvenNumber()
          checkEvenNumber(n);
        System.out.println(n +" is even.");
      } catch(Exception e){
        System.out.println("Error: " + e.getMessage());
      }
    }

    public static void checkEvenNumber(int number) {
```

```
    if (number% 2 != 0) {
      throw new ArithmeticException(number +" is odd.");
    }
  }
}
```

Expected	Got
82 is even. Error: 37 is odd.	82 is even. Error: 37 is odd.

Passed all tests!

Ex.No. 9.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

In the following program, an array of integer data is to be initialized.
During the initialization, if a user enters a value other than an integer, it will throw an lnputMismatchException exception.
On the occurrence of such an exception, your program should print "You entered bad data."
If there is nosuch exception it will print the total sum of the array.

/* Define try-catch block to save user input in the array "name"
If there is an exception then catch the exception otherwise print the total sum of the array.*/

Sample Input:

3
521

Sample Output:

8

Sample Input:

2

1 g

Sample Output:

You entered bad data.

For example:

Input	Result
3 521	8
2 lg	You entered bad data.

PROGRAM

```
import java.util.Scanner;

import java.util.InputMismatchException;
 class prog {
  public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    int length= sc.nextint();
    // create an array to save user input
    int[] name= new int[length];
    int sum=0;
    int x=0;//save the total sum of the array.

    /* Define try-catch block to save user input in the array "name"
   If there is an exception then catch the exception otherwise print
   the total sum of the array. */
     try
      {
          for(int i=0;i<length;i++){
              x=sc.nextint();
              sum+=x;

          }
          System.out.println(sum);

         }
      catch(InputMismatchException e){
        System.out.print("You entered bad data.");

    }
  }
 }
```

input	Expected	Got
3 5 2 1	8	8
2 1 g	You entered bad data.	You entered bad data.

Passed all tests!

Ex.No. 9.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Write a Java program to handle ArithmeticException and ArraylndexOutOfBoundsException.

Create an array, read the input from the user, and store it in the array.

Divide the 0th index element by the 1st index element and store it.

if the 1st element is zero, it will throw an exception.

if you try to access an element beyond the array limit throws an exception.

Input:

5

10 0 20 30 40

Output:

java.lang.ArithmeticException: / by zero
I am always executed

Input:

3
10 20 30

Output
java.lang.ArraylndexOutOfBoundsException: Index 3 out of bounds for length 3
I am always executed

For example:

Test	Input	Result
1	6	java.lang.ArithmeticException: / by zero always executed

PROGRAM

```
import java.util.*;

public class main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n=sc.nextint();
        int[] name= new int[n];
        try{
            for(int i=0;i<n;i++){
            name[i]=sc.nextint();
            }
            if(name[1]==0){
                 throw new ArithmeticException("/ by zero");

            }
            else{
                throw new ArrayindexOutOfBoundsException("Index "+n+" out of
bounds for length "+n);
            }
        }
        catch(ArithmeticException e){
            System.out.println("java.lang.ArithmeticException:
"+e.getMessage());
        }
        catch(ArrayindexOutOfBoundsException g){
            System.out.println("java.lang.ArrayindexOutOfBoundsException:
"+g.getMessage());
        }
        finally{
            System.out.println("I am always executed");
        }



        }
    }
```

	Tttt	Input	Exp«t<d	Got
✓	1	6 1 0 4 1 2 8	java.lang.ArithmeticException: / by zero I am always executed	java.lang.ArithmeticException: / by zero I am always executed
✓	2	3 10 20 30	java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3 I am always executed	java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for 1 I am always executed

Passed all tests! ✓

10 - COLLECTION LIST

Ex.No. **10.1** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Given an ArrayList, the task is to get the first and last element of the ArrayList in Java.

Input: ArrayList = [1, 2, 3, 4]

Output: First= 1, Last = 4

Input: ArrayList= [12, 23, 34, 45, 57, 67, 89]

Output: First= 12,Last= 89

Approach:

1. Get the ArrayList with elements.
2. Get the first element of ArrayList using the get(index) method by passing index = 0.
3. Get the last element of ArrayList using the get(index) method by passing index = size - 1.

PROGRAM

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

class prog {

public static void main(String[] args)
{
 Scanner sc= new Scanner(System.in);
 int n = sc.nextint();

 Arraylist<Integer> list= new Arraylist<Integer>();

 for(int i = 0; i<n;i++)
 list.add(sc.nextint());
 System.out.println("Arraylist: "+list);
 System.out.println("First : "+list.get(0)+", Last    "+list.get(n-1));

 }
}
```

	Test	Input	Expected	Got	
✓	1	6 30 20 40 50 10 80	ArrayList: [30, 20, 40, 50, 10, 80] first : 30, last : 80	ArrayList: [30, 20, 40, 50, 10, 80) First : 30, Last : 80	✓
✓	2	4 5 15 2S 35	ArrayList: [5, 1S, 25, 35) first : 5, Last : 3S	Arraylist: [5, 15, 2S, 35) first : 5, Last : 3S	

Passed all tests! ✓

Ex.No. **10.2** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

The given Java program is based on the ArrayList methods and its usage. The Java program is partially filled. Your task is to fill in the incomplete statements to get the desired output.

list.set0;

list.indexOfO);

list.lastlndexOf0)

list.contains0

list.size0);

list.add◊;

list.remove0;

The above methods are used for the below Java program.

PROGRAM

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

class prog {

public static void main(String[] args)
{
 Scanner sc= new Scanner(System.in);
 int n = sc.nextint();

 Arraylist<Integer> list   new Arraylist<Integer>();

 for(int i = 0; i<n;i++)
 list.add(sc.nextint());

 // printing initial value Arraylist
 System.out.println("Arraylist: "+ list);

//Replacing the element at index 1 with 100
    list.set(1,100);

 //Getting the index of first occurrence of 100
```

```
System.out.println("Index of 100 = "+list.indexOf(100));

//Getting the index of last occurrence of 100
 System.out.println("Lastindex of 100 = "+ list.lastindexOf(100));
// Check whether 200 is in the list or not
System.out.println(list.contains(200)); //Output : false
 // Print Arraylist size
 System.out.println("Size Of Arraylist = ''+list.size());
//Inserting 500 at index 1
      list.add(1,500);                          // code here
 //Removing an element from position 3
      list.remove(3);                          // code here
  System.out.print("Arraylist: " + list);
 }
}
```

Test	Input	Expected	Got
1	5	Arraylist: [1, 2, 3, 100, SJ	Arraylist: [1, 2, 3., 100, SJ
	1	Index of 100 1	Index of 100 • 1
	2	Lastlndex of 100 • 3	Lastindex of 100 • 3
	3	false	false
	100	Si:e Of Arraylist s 5	Size Of ArrayList s 5
	5	Arraylist: |1, 500, 100, 100, SJ	Arraylist:[1, 500, 100, 100, SJ

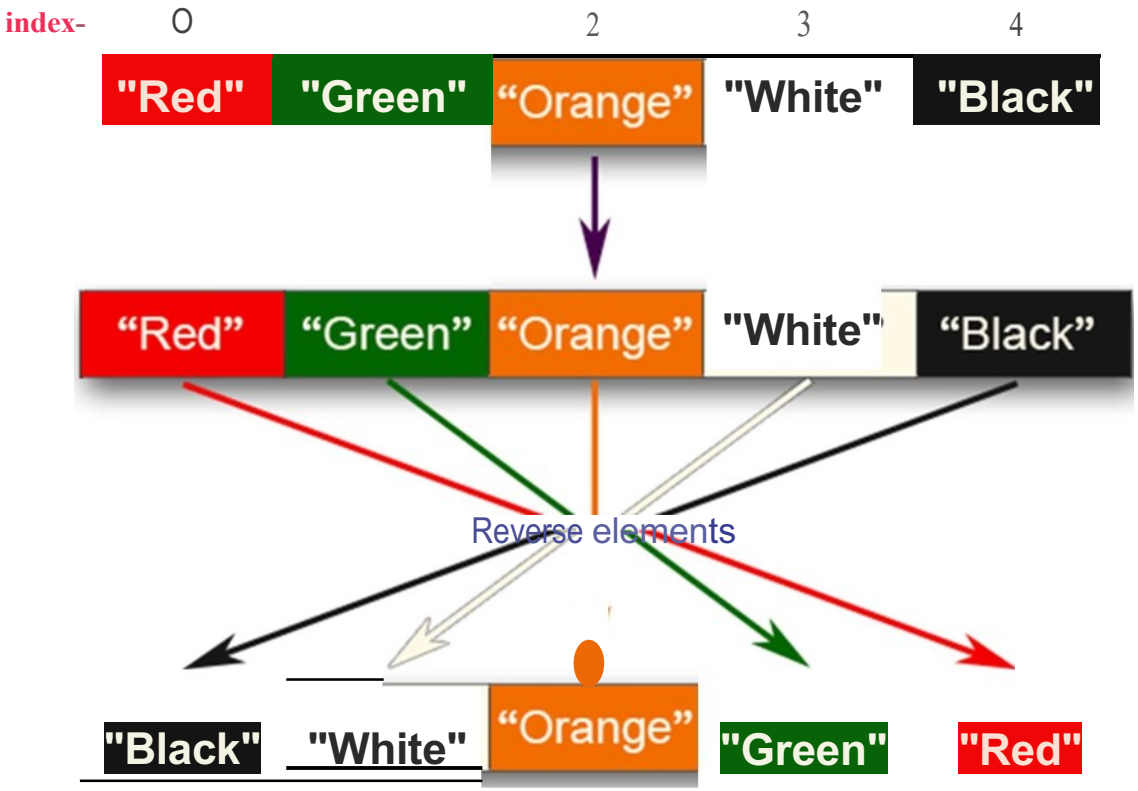
Passed all tests!

Ex.No. 10.3 Date:

Register No.: 231501048 Name: Girivasanth V

Problem 3

Write a Java program to reverse elements in an array list.



Sample input and Output:
Red
Green
Orange
White
Black
Sample output
List before reversing :

[Red, Green, Orange, White, Black]

List after reversing :

[Black, White, Orange, Green, Red]

PROGRAM

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

class prog {

public static void main(String[] args)
{
 Scanner sc= new Scanner(System.in);
 int n = sc.nextint();

 Arraylist<String> list= new Arraylist<String>();

 for(int i = 0; i<n;i++)
 list.add(sc.next());
 Arraylist<String> list1 = new Arraylist<String>();
 for(int i=list.size()-1;i>=0;i--){
     listl.add(list.get(i));
 }
 System.out.println("List before reversing  .").,
 System.out.println(list);

 System.out.println("List after reversing:");
 System.out.println(list1);

 }
}
```

Test	Input	Expected	Got
1	5 Red Green Orange White Black	List before reversing [Red, Green, Orange, White, Black) List after reversing: [Black, 1-lhite, Orange, Green, Red]	List before reversing [Red, Green, Orange, White, Black] List after reversing: [Black, White, Orange, Green, Red]
2	4 CSE AIML AIDS CYBER	List before reversing: [CSE, AHIL, AIDS, CYBER] List after reversing: [CYBER, AIDS, AIML, CSE)	List before reversing: [CSE, Alf·ll, AIDS, CYBER) List after reversing: [CYBER, AIDS, AUil, CSE)

Passed all tests!

11- SET, MAP

Ex.No. **11.1** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Java HashSet class implements the Set interface, backed by a hash table which is actually a HashMap instance.

No guarantee is made as to the iteration order of the hash sets which means that the class does not guarantee the constant order of elements over time.

This class permits the null element.

The class also offers constant time performance for the basic operations like add, remove, contains, and size assuming the hash function disperses the elements properly among the buckets.

Java HashSet Features

A few important features of HashSet are mentioned below:

- Implements Set Interface.
- The underlying data structure for HashSet is Hashtable.
- As it implements the Set Interface, duplicate values are not allowed.
- Objects that you insert in HashSet are not guaranteed to be inserted in the same order. Objects are inserted based on their hash code.
- NULL elements are allowed in HashSet.
- HashSet also implements Serializable and Cloneable interfaces.
- public class HashSet<E> extends AbstractSet<E> implements Set<E>, Cloneable, Serializable
  Sample Input and Output:
  5
  90
  56
  45
  78
  25
  78
  Sample Output:
  78 was found in the set.
  Sample Input and output:
  3
  2
  7
  9

5

Sample Input and output:

5 was not found in the set.

PROGRAM

```
import java.util.HashSet;
import java.util.Scanner;
public class Prog {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextint();
        HashSet<Integer> numbers= new HashSet<>();
        for (inti= 0; i < n; i++) {
            numbers.add(sc.nextint());
        }
        int skey = sc.nextint();
        if (numbers.contains(skey)) {
            System.out.println(skey +"was found in the set.");
        } else {
            System.out.println(skey +"was not found in the set.");
        }
        sc.close();
    }
}
```

Test	Input	Expected	Got
1	5 90 56 45 78 25 78	78 tias found in the set.	78,as found in the set.
1	3 -1 2 4 5	5 was not found in the set.	5 was not found in the set.

Passed all tests!

Ex.No. 11.2 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Write a Java program to compare two sets and retain elements that are the same.

Sample Input and Output:

5

Football

Hockey

Cricket

Volleyball

Basketball

7 // HashSet 2:

Golf

Cricket

Badminton

Football

Hockey

Volleyball

Handball

SAMPLE OUTPUT:

Football

Hockey

Cricket

Volleyball

Basketball

PROGRAM

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;
public class CompareSets {
    public static void main(String[] args) {
```

```
        Scanner scanner= new Scanner(System.in);
        int sizel = Integer.parseint(scanner.nextline());
        Set<String> setl = new HashSet<>();
        for (inti= 0; i < sizel; i++) {
            setl.add(scanner.nextline());
        }
        int size2 = Integer.parseint(scanner.nextline());
        Set<String> set2 = new HashSet<>();
        for (inti= 0; i < size2; i++) {
            set2.add(scanner.nextline());
        }
        setl.retainAll(set2);
        for (String element : setl) {
            System.out.println(element);
        }
        scanner.close();
    }
}
```

	Test	Input	Expected	Got	
V	1	5 Football Hockey Cricket Volleyball Basketball 7 Golf Cricket Badminton Football Hockey Volleyball Thro,vball	Cricket Hockey Volleyball Football	Cricket Hockey Volleyball Football	V
V	2	4 Toy Bus Car Auto 3 Car Bus Lorry	Bus Car	Bus Car	V

Passed all tests! V

Ex.No. **11.3** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

Java HashMap Methods

containsKeyO Indicate if an entry with the specified key exists in the map

containsValueO Indicate if an entry with the specified value exists in the map

putifAbsentO Write an entry into the map but only if an entry with the same key does not already exist

removeO Remove an entry from the map

replace◊ Write to an entry in the map only if it exists

sizeO Return the number of entries in the map

Your task is to fill the incomplete code to get desired output

PROGRAM

```
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Scanner;
import java.util.Set;
public class Prag {
    public static void main(String[] args) {
        HashMap<String, Integer> map= new HashMap<String, Integer>();
        String name;
        int num;
        Scanner sc = new Scanner(System.in);
        int n = sc.nextint();
        for (inti= 0; i < n; i++) {
            name= sc.next();
            num = sc.nextint();
            map.put(name, num);
        }
        Set<Entry<String, Integer>> entrySet = map.entrySet();
        for (Entry<String, Integer> entry : entrySet) {
            System.out.println(entry.getKey() + " : " + entry.getValue());
        }
        System.out.println("------ ");
        HashMap<String, Integer> anotherMap    new HashMap<String, Integer>();
        anotherMap.put("SIX", 6);
        anotherMap.put("SEVEN", 7);
        anotherMap.putAll(map);
```

```
        entrySet = anotherMap.entrySet();
        for (Entry<String, Integer> entry   entrySet) {
            System.out.println(entry.getKey() + "" + entry.getValue());
        }
        map.putifAbsent("FIVE",  5);
        int value= map.get("TWO");
        System.out.println(value);
        System.out.println(map.containsKey("ONE"));
        boolean valueExists = map.containsValue(3);
        System.out.println(valueExists);
        System.out.println(map.size());
    }
}
```

Test	Input	Expected	Got
1	3	ONE : 1	ONE : 1
	ONE	THO : 2	TL-J0 : 2
	1	THREE : 3	THREE : 3
	TI-10	---	---
	2	SIX : 6	SIX : 6
	THREE	ONE : 1	ONE : 1
	3	n/0 : 2	mo : 2
		SEVEN : 7	SEVEN : 7
		THREE : 3	THREE : 3
		2	2
		true	true
		true	true
		4	4

Passed all tests!

12 - INTRODUCTION TO I/O, I/O OPERATIONS, OBJECT SERIALIZATION

Ex.No. **12.1** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 1

Given two char arrays inputlD and input2D containing only lower case alphabets, extracts the alphabets which are present in both arrays (common alphabets).

Get the ASCII values of all the extracted alphabets.

Calculate sum of those ASCII values. Lets call it suml and calculate single digit sum of sum1, i.e., keep adding the digits of suml until you arrive at a single digit.

Return that single digit as output.

Note:

1. Array size ranges from 1 to 10.
2. All the array elements are lower case alphabets.
3. Atleast one common alphabet will be found in the arrays.

Example 1:

inputl: {'a', 'b', 'c'}

input2: {'b', 'c'}

output: 8

Explanation:

'b' and 'c' are present in both the arrays.

ASCII value of 'b' is 98 and 'c' is 99.

98 + 99 = 197

1+9+7=17

1+7=8

Forexample:

Input	Result
abc	8

PROGRAM

```
import java.util.HashSet;
import java.util.Set;
public class CommonAlphabetSum {
    public static int singleDigitSum(int num) {
        int sum= 0;
        while (num > 0) {
            sum+= num % 10;
            num /= 10;
        }
        if (sum> 9) {
            return singleDigitSum(sum);
        }
        return sum;
    }
    public static int calculateCommonAlphabetSum(char[J inputl, char[] input2)
{
        Set<Character> setl = new HashSet<>();
        for (char c : inputl) {
            setl.add(c);
        }
        int sum= 0;
        for (char c : input2) {
            if (setl.contains(c)) {
                sum+= c;
            }
        }
        return singleDigitSum(sum);
    }
    public static void main(String[] args) {
        char[] inputl = {'a', 'b', 'c'};
        char[] input2 = {'b', 'c', 'd'};
        int result= calculateCommonAlphabetSum(inputl, input2);
        System.out.println(result);
    }
}
```

Input	Expected	Got	
ab c b c	8	8	✓

Passed all tests!

Ex.No. **12.2** **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 2

Write a function that takes an input String (sentence) and generates a new String (modified sentence) by reversing the words in the original String, maintaining the words position.

In addition, the function should be able to control the reversing of the case (upper or lowercase) based on a case_option parameter, as follows:

If case_option = 0, normal reversal of words i.e., if the original sentence is "Wipro TechNologies BangaLore", the new reversed sentence should be "orpiW seigoloNhceT eroLagnaB".

If case_option = 1, reversal of words with retaining position's case i.e., if the original sentence is "Wipro TechNologies BangaLore", the new reversed sentence should be "Orpiw SeigOlonhcet ErolaGnab".

Note that positions 1, 7, 11, 20 and 25 in the original string are uppercase W, T, N, B andL.

Similarly, positions 1, 7, 11, 20 and 25 in the new string are uppercase 0, S, 0, E and G.

NOTE:

1. Only space character should be treated as the word separator i.e., "Hello World" should be treated as two separate words, "Hello" and "World". However, "Hello,World", "Hello;World", "Hello-World" or "Hello/World" should be considered as a single word.

2. Non-alphabetic characters in the String should not be subjected to case changes. For example, if case option= 1 and the original sentence is "Wipro TechNologies, Bangalore" the new reversed sentence should be "Orpiw ,seiGolonhceT Erolagnab". Note that comma has been treated as part of the word "Technologies," and when comma had to take the position of uppercase T it remained as a comma and uppercase T took the position of comma. However, the words "Wipro and Bangalore" have changed to "Orpiw" and "Erolagnab".

3. Kindly ensure that no extra (additional) space characters are embedded within the resultant reversed String.

Examples:

S.No.	inputl	input2	output
1	Wipro Technologies Bangalore	0	'3rpiW seigolonhceT erolagnaB

2	Wipro Technologies, Bangalore	0	prpiW ,seigolonhceT erolagnaB
3	Wipro Technologies Bangalore	1	Orpiw Seigolonhcet Erolagnab
4	Wipro Technologies, Bangalore	1	Orpiw ,seigolonhceT Erolagnab

For example:

Input	Result
Wipro Technologies Bangalore 0	orpiW seigolonhceT erolagnaB
Wipro Technologies, Bangalore 0	orpiW ,seigolonhceT erolagnaB
Wipro Technologies Bangalore 1	Orpiw Seigolonhcet Erolagnab
Wipro Technologies, Bangalore 1	Orpiw ,seigolonhceT Erolagnab

PROGRAM

```
import java.util.Scanner;
public class WordReverser {
    public static String reverseWordsWithCase(String sentence, int caseOption)
{
        String[] words= sentence.split(" ");
        StringBuilder result= new StringBuilder();
        for (String word : words) {
            String reversedWord = new
StringBuilder(word).reverse().toString();
            if (caseOption == 0){
                result.append(reversedWord).append(" ");
            } else if (caseOption == 1) {
                result.append(applyCaseConversion(reversedWord,
word)).append(" ");
            }
        }
        return result.toString().trim();
    }
```

```
    private static String applyCaseConversion(String reversedWord, String
originalWord) {
        StringBuilder adjustedWord = new StringBuilder();
        for (inti= 0; i < reversedWord.length(); i++) {
            char reversedChar = reversedWord.charAt(i);
            char originalChar = originalWord.charAt(i);
            if (Character.islowerCase(originalChar)) {
                adjustedWord.append(Character.toLowerCase(reversedChar));
            } else if (Character.isUpperCase(originalChar)) {
                adjustedWord.append(Character.toUpperCase(reversedChar));
            } else {
                adjustedWord.append(reversedChar);
            }
        }
        return adjustedWord.toString();
    }
    public static void main(String[] args) {
        Scanner scanner= new Scanner(System.in);
        String sentence= scanner.nextline();
        int caseOption = scanner.nextint();
        if (caseOption != 0 && caseOption != 1) {
            System.out.println("Invalid case option. Please enter 0 or 1.");
        } else {
            String result= reverseWordsWithCase(sentence, caseOption);
            System.out.println(result);
        }
        scanner.close();
    }
}
```

Input	Expected	Got
Wipro Technologies Bangalore 0	orpiW seigolonhceT erolagnaB	orpiW seigolonhceT erolagnaB
Wipro Technologies, Bangalore 0	orpiW ,seigolonhceT erolagnaB	orpiW ,seigolonhceT erolagnaB
IHpro Technologies Bangalore 1	Orpiw Seigolonhcet Erolagnab	Orpiw Seigolonhcet Erolagnab
Wipro Technologies, Bangalore 1	Orpiw ,seigolonhceT Erolagnab	Orpiw ,seigolonhceT Erolagnab

Passed all tests!

Ex.No. 12.3 **Date:**

Register No.: 231501048 **Name: Girivasanth V**

Problem 3

You are provided with a string which has a sequence of l's and O's.
This sequence is the encoded version of a English word. You are supposed write a program to decode the provided string and find the original word.
Each alphabet is represented by a sequence of Os.
This is as mentioned below:
Z : O
Y: 00
X: 000
W: 0000
V: 00000
U: 000000
T: 0000000
and so on upto A having 26 O's (00000000000000000000000000).
The sequence of O's in the encoded form are separated by a single 1 which helps to distinguish between 2 letters.
Example 1:
inputl: 010010001
The decoded string (original word) will be: ZYX
Example 2:
inputl:00001000000000000000001000000000001000000000100000000000001
The decoded string (original word) will be: WIPRO
Note: The decoded string must always be in UPPER case.

For example:

Input	Result
010010001	ZYX
00001000000000000000001000000000001000000000100000000000001	**WIPRO**

PROGRAM

```
import java.util.Scanner;

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

        StringBuilder decodedString = new StringBuilder();
        int count= 0;

        for (inti= 0; i < encodedString.length(); i++) {
```

```
            if (encodedString.charAt(i)        'O') {
                count++;
            } else {

                char decodedChar = (char)('Z' - count+ 1);
                decodedString.append(decodedChar);
                count= 0;
            }
        }

        System.out.println(decodedString.toString());
    }
}
```

	Input	Expected	Got	
✓	010010001	ZYX	ZYX	
✓	00001000000000000000000100000000001000000000100000000000001	I IPRO	I IPRO	✓

Passed all tests!