RAJALAKSHMI ENGINEERING COLLEGE

[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE

CS23333 OBJECT ORIENTED PROGRAMING USING JAVA

Laboratory Record Note Book

Name : . srinithya A. . . . . .

Year / Branch / Section : . . II/IT/D. . . . . .

College Roll No. : . . . . . 2116231001212. . . . . .

Semester : . . . . . III. . . . . .

Academic Year : . . . . . 2024-2025 . . . . . .

RAJALAKSHMI ENGINEERING COLLEGE

[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name : . . . . .srinithya A. . . . . .

Academic Year : 2024-2025 Semester:. III Branch : IT-D

Register No. 2116231001212

Certified that this is the bonafide record of work done by the above student in the CS23333 –Object Oriented Programming using JAVA during the year 2024 - 2025.

Signature of Faculty in-charge

Submitted for the Practical Examination held on . . . . .27/11/2024. . . . . .

Internal Examiner **External Examiner**

INDEX

Lab Week	Date	Name of the Experiment	Page No	Signature
1	20.9.24	Java Architecture, Language Basics	1	
2	20.9.24	Flow Control Statements	5	
3	21.9.24	Arrays	11	
4	1.10.24	Classes And Objects	17	
5	1.10.24	Inheritance	23	
6	3.10.24	String, StringBuffer	29	
7	3.10.24	Interfaces	35	
8	6.10.24	Polymorphism, Abstract Classes, Final Keyword	41	
9	9.10.24	Exceptional Handling	47	
10	4.10.24	Collection - List	52	
11	10.11.24	Set, Map	57	
12	10.11.24	Introduction to I/O, I/O Operations, Object Serialization	63	
13	27.11.24	Java Project Report	72	

Question **1**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

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

	Input	Expected	Got	
✓	123	2	2	✓
✓	456	1	1	✓

Passed all tests! ✓

Question **1**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class lastDigit
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int a=n%10;
        int b=Math.abs(a);
        System.out.println(b);
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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 two digits should be 11

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Sum
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int a=sc.nextInt();
        int b=sc.nextInt();
        int c=Math.abs(a%10);
        int d=Math.abs(b%10);
        System.out.println(c+d);
    }
}
```

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

Passed all tests! ✓

◄ Lab-01-MCQ

Jump to...

Is Even? ►

Question **1**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

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

	Input	Expected	Got	
✓	5	4	4	✓
✓	8	24	24	✓
✓	11	149	149	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Sports
{
    public static void main(String[] main)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        boolean b=sc.nextBoolean();
        if( (n==6 || n==2 || n==3 || n==4|| n==5) && b==false )
        {
            System.out.println("5:00");
        }
        else if( (n==1 || n==7) && b==false)
        {
            System.out.println("6:00");
        }
```

```
        else if( (n==6 || n==2 || n==3 || n==4 || n==5) && b==true)
        {
            System.out.println("7:00");
        }
        else if( (n==1 || n==7) && b==true )
        {
            System.out.println("9:00");
        }
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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 0 to 10)

the critic's rating of the movie (range 0 to 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 is 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:

9 5

Output:

Yes

Example input:

1 9

Output:

No

Example input:

6 4

Output:

Maybe

For example:

Input	Result
9 5	Yes
1 9	No
6 4	Maybe

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Movie
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int acting=sc.nextInt();
        int rating=sc.nextInt();
        if(acting<2 || rating<2)
        {
            System.out.println("No");
        }
        else if(acting>8 || rating>8)
        {
            System.out.println("Yes");
```

```
        }
        else
        {
            System.out.println("Maybe");
        }
    }
}
```

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

Passed all tests! ✓

◄ Lab-02-MCQ

Jump to...

Lab-03-MCQ ►

Question **1**

Correct

Marked out of 5.00

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 0 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).

input1 represents the number of elements in the array.

input2 represents the array of integers.

Example 1:

input1 = 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:

input1 = 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:

input1 = 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 -79	62
11 -22 -24 -16 -1 -17 -19 -37 -25 -19 -93 -61	-1
16 -58 32 26 92 -10 -4 12 0 12 -2 4 32 -9 -7 78 -79	174

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Main
{
```

```java
    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 maxLen=0,currentLen=0,maxSum=0;
        int currentSum=0;
        boolean hasPositive=false;
        for(int i=0;i<n;i++)
        {
            if(arr[i]>=0)
            {
                hasPositive=true;
                currentLen++;
                currentSum+=arr[i];

            }
            else
            {
                if(currentLen>maxLen)
                {
                    maxLen=currentLen;
                    maxSum=currentSum;
                }
                else if(currentLen==maxLen)
                {
                    maxSum+=currentSum;
                }
                currentLen=0;
                currentSum=0;

            }
        }
        if(currentLen>maxLen)
        {
            maxSum=currentSum;
         }
        else if(currentLen==maxLen)
        {
            maxSum+=currentSum;

        }
        if(!hasPositive)
        {
```

	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! ✓

Question **2**

Correct

Marked out of 5.00

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:

input1 = 4 (represents the number of elements in the input1 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, 0}.

Example 2:

input1 = 5 (represents the number of elements in the input1 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:

input1 = 2 (represents the number of elements in the input1 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, 0}

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

{(-18 x 9), (0 x 9)} = {-162, 0}

So, the expected output is the resultant array {-162, 0}.

Note: The input array will contain not more than 100 elements

For example:

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

Answer: (penalty regime: 0 %)

```java
import java.util.Scanner;
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int input1=sc.nextInt();
        int[] input2=new int[input1];
        for(int i=0;i<input1;i++)
        {
            input2[i]=sc.nextInt();
        }
        int max=Integer.MIN_VALUE;
        for(int i=0;i<input1;i++)
        {
            if(input2[i]>max)
            {
                max=input2[i];
            }
        }
        for(int i=0;i<input1;i++)
        {
            input2[i]=(input2[i]-max)*max;
        }
        for(int i=0;i<input1;i++)
        {
            System.out.print(input2[i]+" ");
        }
    }
}
```

	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! ✓

Question **3**

Correct

Marked out of 5.00

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:

input1:5 and input2: {1, 51, 436, 7860, 41236}

Step 1:

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

0th index – pick up the units value of the number (in this case is 1).

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

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

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

4th 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.

{1, 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, input1[] is the array of numbers and input2 represents the number of elements in input1.

Example 2:

input1: 5 and input1: {1, 5, 423, 310, 61540}

Step 1:

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

{1, 0, 4, 0, 6}

In this case, the value in input1 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 51 436 7860 41236	107
5 1 5 423 310 61540	53

Answer: (penalty regime: 0 %)

```java
import java.util.Scanner;
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int input1=sc.nextInt();
        int[] input2=new int[input1];
        for(int i=0;i<input1;i++)
        {
            input2[i]=sc.nextInt();

        }
        int[] resultArray=new int[input1];
        for(int i=0;i<input1;i++)
        {
            String numStr=String.valueOf(input2[i]);
            int position=numStr.length()-1-i;
            if(position>=0)
            {
                resultArray[i]=Character.getNumericValue(numStr.charAt(position));
            }
            else
            {
                resultArray[i]=0;
            }
        }
        int sum=0;
        for(int i=0;i<input1;i++)
        {
            resultArray[i]*=resultArray[i];
            sum+=resultArray[i];
        }
        System.out.println(sum);
    }
}
```

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

Passed all tests! ✓

◄ Lab-03-MCQ

Jump to...

Simple Encoded Array ►

Question **1**

Correct

Marked out of 5.00

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 = πr^2

Circumference = $2\pi r$

Input:

2

Output:

Area = 12.57
Circumference = 12.57

For example:

Test	Input	Result
1	4	Area = 50.27 Circumference = 25.13

Answer: (penalty regime: 0 %)

Reset answer

```
import java.io.*;
import java.util.Scanner;
class Circle
{
    private double radius;
    public Circle(double radius){
        // set the instance variable radius
        this.radius=radius;


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


    }
    public double getRadius()    {
        // return the radius
        return 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()));
        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! ✓

Question **2**

Correct

Marked out of 5.00

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 getManufacturer(){

return manufacturer;}

Display the object details by overriding the toString() method.

For example:

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

Answer: (penalty regime: 0 %)

```
class prog
{
    private String manufacturer;
    private String operating_system;
    private String color;
    private int cost;
    public prog(String manufacturer,String operating_system,String color,int cost)
    {
        this.manufacturer=manufacturer;
        this.operating_system=operating_system;
        this.color=color;
        this.cost=cost;
    }
    public void setManufacturer(String manufacturer)
    {
        this.manufacturer=manufacturer;
    }
    public void setOperating_system(String operating_system)
    {
        this.operating_system=operating_system;
    }
    public void setColor(String color)
    {
        this.color=color;
    }
    public void setCost(int cost)
    {
        this.cost=cost;
    }
    public String getManufacturer()
    {
        return manufacturer;
    }
```

```
    }
    public String getOperating_system()
    {
        return operating_system;
    }
    public String getColor()
    {
        return color;
    }
    public int getCost()
    {
        return cost;
    }
    public String toString()
    {
        return "manufacturer = "+manufacturer+"\n"+
               "operating_system = "+operating_system+"\n"+
               "color = "+color+"\n"+
               "cost = "+cost;
    }
```

	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! ✓

Question **3**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

```java
class prog
{
    private String name;
    private int rollno;
    public prog()
    {
        this.name=null;
        this.rollno=0;
        System.out.println("No-arg constructor is invoked");
    }
    public prog(String name)
    {
        this.name=name;
        this.rollno=0;
        System.out.println("1 arg constructor is invoked");
    }
    public prog(String name,int rollno)
    {
        this.name=name;
        this.rollno=rollno;
        System.out.println("2 arg constructor is invoked");
    }
    public void display()
    {
        System.out.println("Name ="+name+" , Roll no = "+rollno);
    }
    public static void main(String[] args)
    {
        prog stu1=new prog();
        prog stu2=new prog("Rajalakshmi");
        prog stu3=new prog("Lakshmi",101);
        stu1.display();
```

```
        stu2.display();
        stu3.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! ✓

◄ Lab-04-MCQ

Jump to...

Number of Primes in a specified range ►

Question **1**

Correct

Marked out of 5.00

Create a class known as "BankAccount" with methods called deposit() and withdraw().

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 (A/c 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 (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0

Answer: (penalty regime: 0 %)

Reset answer

```java
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
    BankAccount(String ac,double bal){
        accountNumber = ac;
        balance = bal;
    }
    // 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;

    }
}

 class SavingsAccount extends BankAccount {
    // Constructor to initialize account number and balance
    public SavingsAccount(String accountNumber, double balance) {
```

```java
        // 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!");
```

	Expected	Got	
✓	Create a Bank Account object (A/c No. BA1234) 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 (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0	Create a Bank Account object (A/c No. BA1234) 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 (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

Answer: (penalty regime: 0 %)

Reset answer

```
class College
{
protected   String collegeName;

public College(String collegeName) {
    // initialize the instance variables
    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 depart) {
    super(collegeName);
   this.studentName = studentName;
   this.department = depart;


}

```

```
public String toString(){
    // return the details of the student
    return "CollegeName : "+collegeName+"\nStudentName : "+studentName+"\nDepartment : "+department;

}
}
// class cse extends Student{
//     private String department;
//     void course(){

//     }
// }
class prog {
public static void main (String[] args) {
        Student  s1 = new Student("REC","Venkatesh","CSE");
        s1.admitted();
                                        // invoke the admitted() method
        System.out.println(s1.toString());
}
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Create a class Mobile with constructor and a method basicMobile().

Create a subclass CameraMobile which extends Mobile class , with constructor and a method newFeature().

Create a subclass AndroidMobile which extends CameraMobile, with constructor and a method androidMobile().

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

Answer: (penalty regime: 0 %)

```
class Moblie{
    Moblie(){
        System.out.println("Basic Mobile is Manufactured");
    }
    // void basicMoblie(){
    // }
}
class CamaraMoblie extends Moblie{
    CamaraMoblie(){
        super();
        System.out.println("Camera Mobile is Manufactured");
    }
   void newFeature(){
        System.out.println("Camera Mobile with 5MG px");
    }
}
class AndroidMoblie extends CamaraMoblie{
    AndroidMoblie(){
        super();
        System.out.println("Android Mobile is Manufactured");

    }
    void androidMoblie(){
        System.out.println("Touch Screen Mobile is Manufactured");

    }
}
public class prog{
```

```
    public static void main(String A[]){
        AndroidMoblie a = new AndroidMoblie();
        a.newFeature();
        a.androidMoblie();
    }
}
```

	Expected	Got	
✓	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	✓

Passed all tests! ✓

◄ Lab-05-MCQ

Jump to...

Is Palindrome Number? ►

Question **1**

Correct

Marked out of 5.00

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:

input1 = ww:ii:pp:rr:oo

output = WIPRO

Explanation:

word1 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 take r

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"

input1 = zx:za:ee

output = BYE

Explanation

word1 is zx, both are not same alphabets

position value of z is 26

position value of x is 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:ii:pp:rr:oo	WIPRO
zx:za:ee	BYE

Answer: (penalty regime: 0 %)

```java
import java.util.Scanner;

public class WordProcessor {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String input = scanner.nextLine();
        String[] words = input.split(":");

        StringBuilder output = new StringBuilder();

        for (String word : words) {
            if (word.length() == 2) {
                char firstChar = word.charAt(0);
                char secondChar = word.charAt(1);

                if (firstChar == secondChar) {
                    output.append(firstChar);
                } else {
                    int position1 = firstChar - 'a' + 1;
                    int position2 = secondChar - 'a' + 1;
                    int maxPos = Math.max(position1, position2);
                    int minPos = Math.min(position1, position2);
                    int newPos = maxPos - minPos;
                    char newChar = (char) ('a' + newPos - 1);
                    output.append(newChar);
                }
            }
        }
        System.out.println(output.toString().toUpperCase());

        scanner.close();
    }
}
```

	Input	Expected	Got	
✓	ww:ii:pp:rr:oo	WIPRO	WIPRO	✓
✓	zx:za:ee	BYE	BYE	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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:

input1 = "Today is a Nice Day"

input2 = 41

output = "iNce doTday"

Example 2:

input1 = "Fruits like Mango and Apple are common but Grapes are rare"

input2 = 39

output = "naMngo arGpes"

Note: The input string input1 will contain only alphabets and a single space character separating each word in the string.

Note: The input string input1 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 input1 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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;

public class WordProcessor {
```

```java
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String inputString = scanner.nextLine();
        int inputNumber = scanner.nextInt();

        String processedString = processWords(inputString, inputNumber);
        System.out.println(processedString);
    }

    public static String processWords(String inputString, int inputNumber) {
        String[] words = inputString.split(" ");

        if (words.length < 2 || inputNumber < 11 || inputNumber > 99) {
            return "Invalid input";
        }

        int word1Index = inputNumber / 10 - 1;
        int word2Index = inputNumber % 10 - 1;

        if (word1Index >= words.length || word2Index >= words.length) {
            return "Invalid word indices";
        }

        String word1 = words[word1Index];
        String word2 = words[word2Index];

        String processedWord1 = processWord(word1);
        String processedWord2 = processWord(word2);

        return processedWord1 + " " + processedWord2;
    }

    public static String processWord(String word) {
        int middleIndex = word.length() / 2;
        StringBuilder input = new StringBuilder();
        StringBuilder input1 = new StringBuilder();
        input.append(word.substring(0,middleIndex+1));
        input1.append(word.substring(0,middleIndex));
        if(word.length()%2==0){
        return input1.reverse()+word.substring(middleIndex);
        }
        else{
            return input.reverse()+word.substring(middleIndex);
        }
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Given 2 strings input1 & 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

Answer: (penalty regime: 0 %)

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;
import java.util.stream.Collectors;

public class StringProcessor {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String input1 = scanner.nextLine();
        String input2 = scanner.nextLine();
        String result = processStrings(input1, input2);
        System.out.println(result);
    }
    public static String processStrings(String input1, String input2) {
        if ((input1 == null || input1.trim().isEmpty()) && (input2 == null || input2.trim().isEmpty())) {
            return "null";
        }
        String concatenated = input1 + input2;
        Set<Character> uniqueChars = new HashSet<>();
        for (char c : concatenated.toCharArray()) {
```

```
            if (c != ' ') {
                uniqueChars.add(c);
            }
        }
        String sortedChars = uniqueChars.stream().map(String::valueOf).sorted((a, b) -> b.compareTo(a)).collect(Co

        return sortedChars;
    }
}
```

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

Passed all tests! ✓

◄ Lab-06-MCQ

Jump to...

Return second word in Uppercase ►

Question **1**

Correct

Marked out of 5.00

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 rateOfInterest().

RBI interface has two more methods default and static method.

default void policyNote() {

System.out.println("RBI has a new Policy issued in 2023.");

}

static void regulations(){

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.

Answer: (penalty regime: 0 %)

```java
interface RBI {
    String parentBank = "RBI";
    double rateOfInterest();
    default void policyNote() {
        System.out.println(parentBank + " has a new Policy issued in 2023");
    }
    static void regulations() {
        System.out.println(parentBank + " has updated new regulations in 2024.");
    }
}
class SBI implements RBI {
    @Override
    public double rateOfInterest() {
        return 7.6;
    }
}
class Karur implements RBI {
    @Override
    public double rateOfInterest() {
        return 7.4;
    }
}
public class BankPolicies {
    public static void main(String[] args) {
        RBI sbi = new SBI();
        RBI karur = new Karur();
        sbi.policyNote();
        RBI.regulations();
        System.out.println("SBI rate of interest: " + sbi.rateOfInterest() + " per annum.");
```

```
        System.out.println("Karur rate of interest: " + karur.rateOfInterest() + " per annum.");
    }
}
```

	Test	Expected	Got	
✓	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.	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.	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

```
interface Playable {
   void play();
}

class Football implements Playable {
   String name;
   public Football(String name){
      this.name=name;
   }
  public void play() {
   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 Playing volleyball Playing basketball
2	Vijay Arun Balaji	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball

Answer: (penalty regime: 0 %)

```java
import java.util.Scanner;
interface Playable {
    void play();
}
class Football implements Playable {
    String name;

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

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

class Volleyball implements Playable {
    String name;

    public Volleyball(String name) {
        this.name = name;
    }
    @Override
    public void play() {
        System.out.println(name + " is Playing volleyball");
```

```
        System.out.println(name + " is Playing volleyball");
    }
}
class Basketball implements Playable {
    String name;

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

    @Override
    public void play() {
        System.out.println(name + " is Playing basketball");
    }
}
public class SportsGame {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Playable[] players = new Playable[3];
        String[] sports = {"Football", "Volleyball", "Basketball"};
        for (int i = 0; i < players.length; i++) {
            String name = scanner.nextLine();
            switch (sports[i]) {
                case "Football":
                    players[i] = new Football(name);
                    break;
                case "Volleyball":
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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!

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.Scanner;
 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);

}
class College implements Football {
    String homeTeam;
    String visitingTeam;

    public void setHomeTeam(String name){
          this.homeTeam=name;
   }
public void setVisitingTeam(String name){
  this.visitingTeam=name;
}
public void homeTeamScored(int points){
    System.out.println(homeTeam+" "+points+" scored");
}
public void visitingTeamScored(int points){
  System.out.println(visitingTeam+" "+points+" scored");
}
```

```
public int winningTeam(int p1, int p2){
    if(p1>p2)
        return 1;
    else if(p1<p2)
        return 0;
    else
        return -1;
}
}
public class Main{
    public static void main(String[] args){
        Scanner sc= new Scanner(System.in);
        String  hname=sc.next();
        String  vteam=sc.next();
        int htpoints=sc.nextInt();
        int vtpoints=sc.nextInt();
    College s= new College();
    s.setHomeTeam(hname);
    s.setVisitingTeam(vteam);
    s.homeTeamScored(htpoints);
    s.visitingTeamScored(vtpoints);
    if((s.winningTeam(htpoints,vtpoints))==1){
        System.out.println(hname+" is the winner!");
    }
    else if((s.winningTeam(htpoints,vtpoints))==-1){
```

	Test	Input	Expected	Got	
✓	1	Rajalakshmi Saveetha 22 21	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner!	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi 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	SRM 20 scored VIT 21 scored VIT is the winner!	SRM 20 scored VIT 21 scored VIT is the winner!	✓

Passed all tests! ✓

◄ Lab-07-MCQ

Jump to...

Generate series and find Nth element ►

Question **1**

Correct

Marked out of 5.00

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.`

Answer: (penalty regime: 0 %)

Reset answer

```
class FinalExample {

    // Final variable
    final int maxSpeed = 120;

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

class SubClass extends FinalExample {

    // public void displayMaxSpeed() {
    //     System.out.println("Cannot override a final method");
    // }

    // You can create new methods here
    public void showDetails() {
        System.out.println("This is a subclass of FinalExample.");
    }
}

class prog {
```

```
    public static void main(String[] args) {
        FinalExample obj = new FinalExample();
        obj.displayMaxSpeed();

        SubClass subObj = new SubClass();
        subObj.showDetails();
    }
}
```

	Test	Expected	Got	
✓	1	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! ✓

Question **2**

Correct

Marked out of 5.00

Create a base class Shape with a method called calculateArea(). Create three subclasses: Circle, Rectangle, and Triangle. Override the calculateArea() 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 calculateArea() ;
   }
}
```

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

sample Input :

4 // radius of the circle to calculate area PI*r*r

5 // length of the rectangle

6 // breadth of the rectangle to calculate the area of a rectangle

4 // base of the triangle

3 // 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

Answer: (penalty regime: 0 %)

```
import java.util.*;
abstract class Shape{
    abstract void calculatearea();
}
class Circle extends Shape{
```

```
    float rad;
    Circle(float rad){
        this.rad = rad;
    }
    void calculatearea(){
        System.out.format("Area of a circle: %.2f\n",3.14159*rad*rad);
    }
}
class Rectangle extends Shape{
    float l;
    float br;
    Rectangle(float l,float br){
        this.l = l;
        this.br = br;
    }
    void calculatearea(){
        System.out.format("Area of a Rectangle: %.2f\n",(l*br));

    }
}
class Triangle extends Shape{
    float ba;
    float h;
    Triangle(float ba ,float h){
        this.ba = ba;
        this.h = h;
    }
    void calculatearea(){
        System.out.format("Area of a Triangle: %.2f",0.5*ba*h);

    }
}
class prog{
    public static void main (String are[]){
        Scanner scan = new Scanner(System.in);
        float rad = scan.nextFloat();
        float l = scan.nextFloat();
        float br = scan.nextFloat();
        float ba = scan.nextFloat();
        float h = scan.nextFloat();
        Circle c = new Circle(rad);
        Rectangle r = new Rectangle(l,br);
        Triangle t = new Triangle(ba,h);
        c.calculatearea();
        r.calculatearea();
        t.calculatearea();

```

	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! ✓

Question **3**

Correct

Marked out of 5.00

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 andlast 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

input1: an integer representing the number of elements in the array.

input2: String array.

Example 1:

input1: 3

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

output: oreoapple

Example 2:

input1: 2

input2: {"Mango", "banana"}

output: no matches found

Explanation:

None of the strings has first andlast character as vowel.

Hence the output is no matches found.

Example 3:

input1: 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

Answer: (penalty regime: 0 %)

```
import java.util.*;
class prog{
    public static void main(String ae[]){
        Scanner scan = new Scanner(System.in);
        int n = scan.nextInt();
        String arr[] = new String[n];
        scan.nextLine();
        String str = scan.nextLine();
        String temp = "";
        int j=0;
        int l=str.length();
```

```
        for(int i = 0;i<l;i++){
            if(str.charAt(i)==' '){
                arr[j] = temp;
                temp ="";
                j++;
            }
            else{
                temp +=str.charAt(i);
            }
        }
        arr[j] = temp;
        String s = "";
        char [] cha ={'a','A','e','E','i','I','o','O','U','u'};
        for(int i=0;i<n;i++){
            int c=0;
            char [] ar = arr[i].toCharArray();
            char ch1 = ar[0];
            char ch2 = ar[ar.length -1];
            for(char k : cha){
                if(k==ch1){
                    c++;
                }
                if(k==ch2){
                    c++;
                }
            }
            if(c==2){
                s+=arr[i];
            }
        }
        if(s==""){
            System.out.print("no matches found");
        }
        else{
            System.out.print(s.toLowerCase());
        }
```

	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! ✓

◄ Lab-08-MCQ

Jump to...

FindStringCode ►

Question **1**

Correct

Marked out of 5.00

Write a Java program to handle ArithmeticException and ArrayIndexOutOfBoundsException.

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.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3
I am always executed

For example:

Test	Input	Result
1	6 1 0 4 1 2 8	java.lang.ArithmeticException: / by zero I am always executed

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Main
{
    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();
        }
        try
        {
            int a=arr[0]/arr[1];
            int b=arr[n];
        }
        catch(ArithmeticException ae)
        {
            System.out.println(ae.toString());
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println(e.toString());
        }
        finally
        {
            System.out.println("I am always executed");
        }
```

```
    }
}
```

	Test	Input	Expected	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 length 3 I am always executed	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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.

Answer: (penalty regime: 0 %)

Reset answer

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

```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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 InputMismatchException exception.
On the occurrence of such an exception, your program should print "You entered bad data."
If there is no such 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
5 2 1

Sample Output:

8

Sample Input:

2

1 g

Sample Output:

```
You entered bad data.
```

For example:

Input	Result
3 5 2 1	8
2 1 g	You entered bad data

Answer: (penalty regime: 0 %)

Reset answer

```
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;//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++)
           {
               name[i]=sc.nextInt();
               sum+=name[i];

           }
           System.out.println(sum);

      }
      catch(InputMismatchException ae )
```

```
    {
            System.out.println("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! ✓

◄ Lab-09-MCQ

Jump to...

The "Nambiar Number" Generator ►

Question **1**

Correct

Marked out of 1.00

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.

Answer: (penalty regime: 0 %)

```java
import java.util.*;
public class Solution
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        ArrayList<Integer> arr=new ArrayList<Integer>(n);
        while(sc.hasNextInt())
        {
            int a=sc.nextInt();
            arr.add(a);
        }

        int b=arr.size()-1;
        System.out.println("ArrayList: "+arr);
        System.out.println("First : "+arr.get(0)+", Last : "+arr.get(b));
    }
}
```

	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 25 35	ArrayList: [5, 15, 25, 35] First : 5, Last : 35	ArrayList: [5, 15, 25, 35] First : 5, Last : 35	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

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.set();

list.indexOf());

list.lastIndexOf())

list.contains()

list.size());

list.add();

list.remove();

The above methods are used for the below Java program.

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.ArrayList;
import java.util.Scanner;

public 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 1 2 3 100 5	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 1.00

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]
```

Answer: (penalty regime: 0 %)

```java
import java.util.*;
public class Reverse
{
    public static void main(String[] args)
    {
       Scanner sc=new Scanner(System.in);
       int n=sc.nextInt();
       ArrayList<String> list=new ArrayList<String>(n);
       for(int i=0;i<n;i++)
       {
           String str=sc.next();
           list.add(str);
       }
       ArrayList<String> list1=new ArrayList<String>(n);
       for(int i=n-1;i>=0;i--)
       {
           String str1=list.get(i);
           list1.add(str1);
       }
       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, White, 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, AIML, AIDS, CYBER] List after reversing : [CYBER, AIDS, AIML, CSE]	List before reversing : [CSE, AIML, AIDS, CYBER] List after reversing : [CYBER, AIDS, AIML, CSE]	✓

Passed all tests! ✓

◄ Lab-10-MCQ

Jump to...

Lab-11-MCQ ►

Question **1**

Correct

Marked out of 1.00

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.
```

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.HashSet;
import java.util.Scanner;
class prog
{
    public static void main(String[] args)
    {
        Scanner sc= new Scanner(System.in);
        int n = sc.nextInt();
        // Create a HashSet object called numbers
        HashSet<Integer> numbers= new HashSet<Integer>(n);
        // Add values to the set
        for(int i=0;i<n;i++)
        {
            numbers.add(sc.nextInt());
        }
        int skey=sc.nextInt();
        // Show which numbers between 1 and 10 are in the set
        if(numbers.contains(skey))
        {
            System.out.println(skey+ " was found in the set.");
```

```
        }
        else
        {
            System.out.println(skey + " was not found in the set.");

        }
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

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

Answer: (penalty regime: 0 %)

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.ArrayList;
public class Main
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        HashSet<String> set1 = new HashSet<>();
        HashSet<String> set2 = new HashSet<>();
        int numElementsSet1 = scanner.nextInt();
        scanner.nextLine();
        for (int i = 0; i < numElementsSet1; i++) {
            String num = scanner.nextLine();
            set1.add(num);
        }
        int numElementsSet2 = scanner.nextInt();
        scanner.nextLine();
        for (int i = 0; i < numElementsSet2; i++)
        {
            String num = scanner.nextLine();
            set2.add(num);
        }
        set1.retainAll(set2);
        ArrayList<String> list = new ArrayList<>(set1);
        for (int i = 0; i < list.size(); i++)
        {
            System.out.println(list.get(i));
```

```
        }
    }
}

```

	Test	Input	Expected	Got	
✓	1	5 Football Hockey Cricket Volleyball Basketball 7 Golf Cricket Badminton Football Hockey Volleyball Throwball	Cricket Hockey Volleyball Football	Cricket Hockey Volleyball Football	✓
✓	2	4 Toy Bus Car Auto 3 Car Bus Lorry	Bus Car	Bus Car	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 1.00

Java HashMap Methods

containsKey() Indicate if an entry with the specified key exists in the map

containsValue() Indicate if an entry with the specified value exists in the map

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

remove() Remove an entry from the map

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

size() Return the number of entries in the map

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

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Set;
import java.util.Scanner;
 class prog
{
    public static void main(String[] args)
    {
        //Creating HashMap with default initial capacity and load factor
        HashMap<String, Integer> map = new HashMap<String, Integer>();

        String name;
        int num;
        Scanner sc= new Scanner(System.in);
        int n=sc.nextInt();
         for(int i =0;i<n;i++)
         {
             name=sc.next();
             num= sc.nextInt();
             map.put(name,num);
         }

        //Printing key-value pairs


        Set<Entry<String, Integer>> entrySet = map.entrySet();

        for (Entry<String, Integer> entry : entrySet)
        {
            System.out.println(entry.getKey()+" : "+entry.getValue());
        }
         System.out.println("————————");
        //Creating another HashMap

        HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();

        //Inserting key-value pairs to anotherMap using put() method

        anotherMap.put("SIX", 6);

        anotherMap.put("SEVEN", 7);

        //Inserting key-value pairs of map to anotherMap using putAll() method

        anotherMap.putAll      ( map   );  // code here

        //Printing key-value pairs of anotherMap

        entrySet = anotherMap.entrySet();
```

```

        for (Entry<String, Integer> entry : entrySet)
        {
```

	Test	Input	Expected	Got	
✓	1	3 ONE 1 TWO 2 THREE 3	ONE : 1 TWO : 2 THREE : 3 ———————— SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	ONE : 1 TWO : 2 THREE : 3 ———————— SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	✓

Passed all tests! ✓

◄ Lab-11-MCQ

Jump to...

TreeSet example ►

Question **1**

Correct

Marked out of 5.00

You are provided with a string which has a sequence of 1's and 0'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 0s.

This is as mentioned below:

Z : 0

Y : 00

X : 000

W : 0000

V : 00000

U : 000000

T : 0000000

and so on upto A having 26 0's (00000000000000000000000000).

The sequence of 0's in the encoded form are separated by a single 1 which helps to distinguish between 2letters.

Example 1:

input1: 010010001

The decoded string (original word) will be: ZYX

Example 2:

input1: 00001000000000000000001000000000001000000000010000000000001

The decoded string (original word) will be: WIPRO

Note: The decoded string must always be in UPPER case.

For example:

Input	Result
010010001	ZYX
00001000000000000000001000000000001000000000010000000000001	WIPRO

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class BinaryStringDecoder
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        String encodedString = scanner.nextLine();
        String[] parts = encodedString.split("1");
        StringBuilder decodedString = new StringBuilder();
        for (String part : parts) {
            int zeroCount = part.length();
            if (zeroCount >= 1 && zeroCount <= 26) {
                char decodedChar = (char) ('Z' - zeroCount + 1);
                decodedString.append(decodedChar);
            }
        }
        System.out.println(decodedString.toString());
    }
}
```

```
20 }
```

	Input	Expected	Got	
✓	010010001	ZYX	ZYX	✓
✓	00001000000000000000001000000000001000000000010000000000001	WIPRO	WIPRO	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

Given two char arrays input1[] and input2[] 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 sum1 and calculate single digit sum of sum1, i.e., keep adding the digits of sum1 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:

input1: {'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

For example:

Input	Result
a b c b c	8

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class ArrayCommonElementSum {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String[] array1 = scanner.nextLine().split(" ");
        String[] array2 = scanner.nextLine().split(" ");
        int asciiSum = 0;
        for (String element : array2) {
            for (String elem1 : array1) {
                if (element.equals(elem1)) {
                    asciiSum += element.charAt(0);
                    break;
                }
            }
        }
        int singleDigitSum = asciiSum % 9;
        if (singleDigitSum == 0 && asciiSum > 0) {
            singleDigitSum = 9;
        }
        System.out.println(singleDigitSum);
    }
```

```
22 }
```

	Input	Expected	Got	
✓	a b c b c	8	8	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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 and L.

Similarly, positions 1, 7, 11, 20 and 25 in the new string are uppercase O, S, O, 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.	input1	input2	output
1	Wipro Technologies Bangalore	0	orpiW seigolonhceT erolagnaB
2	Wipro Technologies, Bangalore	0	orpiW ,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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class WordReversal
{
    public static String reverseWords(String sentence, int caseOption) {
        String[] words = sentence.split(" ");
        StringBuilder modifiedSentence = new StringBuilder();
        for (String word : words) {
            String reversedWord = reverseWord(word, caseOption);
            if (modifiedSentence.length() > 0) {
```

```
                modifiedSentence.append(" ");            }
            modifiedSentence.append(reversedWord);
        }

        return modifiedSentence.toString();
    }

    private static String reverseWord(String word, int caseOption) {
        String reversedWord = new StringBuilder(word).reverse().toString();
        if (caseOption == 0) {
            return reversedWord;
        } else if (caseOption == 1) {
            StringBuilder casePreservedWord = new StringBuilder(reversedWord);

            for (int i = 0; i < word.length(); i++) {
                char originalChar = word.charAt(i);
                if (Character.isUpperCase(originalChar)) {
                    casePreservedWord.setCharAt(i, Character.toUpperCase(casePreservedWord.charAt(i)));
                } else if (Character.isLowerCase(originalChar)) {
                    casePreservedWord.setCharAt(i, Character.toLowerCase(casePreservedWord.charAt(i)));
                }
            }
            return casePreservedWord.toString();
        }
        return word;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String sentence = scanner.nextLine();
        int caseOption = scanner.nextInt();
        String result = reverseWords(sentence, caseOption);
        System.out.println( result);
    }
}
```

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

Passed all tests! ✓

◄ Lab-12-MCQ

Identify possible words ►

GYM MANAGEMENT SYSTEM

CS23333 – Object Oriented Programming using Java Project Report

Submitted by

SRIMATHI B.S -231001209
SRINITHYA A -231001212
SHREEKUMARAN S -231001195

Of

BACHELOR OF TECHNOLOGY
In

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING

COLLEGENOVEMBER-

2024

BONAFIDE CERTIFICATE

Certified that this project titled "GYM MANAGEMENT SYSTEM" is the bonafide work of "**SRIMATHI B.S(231001209), SRINITHYA A (231001212), SHREEKUMARAN S (231001195)**"who carried out the project work under my supervision.

SIGNATURE
Dr.P.Valarmathie
HEAD OF THE DEPARTMENT

Department of Information Technology
Rajalakshmi Engineering College

SIGNATURE
Mr.K.E NARAYANAN
COURSE INCHARGE

Department of Information Technology
Rajalakshmi Engineering College

This project is submitted for CS23333 – Object Oriented Programming usingJava held on _____

INTERNAL EXAMINAR

EXTERNAL EXAMINAR

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	List of Figures	4
	List of Tables	5
1	1.1 Abstract	6
	1.1 Introduction	6
	1.3 Purpose	6
	1.4 Scope of Project	7
	1.5 Software Requirement Specification	7
2	**System flow Diagrams**	12
	2.1 Use Case Diagram	12
	2.2 Entity-relationship Diagrams	13
	2.3 Data Flow Diagram	13
3	**Module Description**	14
4	4.1 Design	15
	4.2 Database Design	18
	4.3 Code	19
5	**Conclusion**	30
6	**Reference**	30

LIST OF FIGURES

Figure Numbers	Figure Captions	Pg.no
2.1	Use Case Diagram	12
2.2	Entity Relation diagram	13
2.3	Date Flow Diagram	13
4.1.1	Login Page	15
4.1.2	Registration Page	15
4.1.3	Home Page	16
4.1.4	Cart Selection	16
4.1.5	Payment Page	17
4.1.6	Order Message	17

LIST OF Table

Figure Numbers	Figure Captions	Pg.no
4.2.1	Login Table	18

1. Abstract

The Gym Management System is a Java-based desktop application developed to manage gym activities efficiently, such as member registration, payments, and trainer assignments. It uses Java Swing for the user interface and JDBC to connect with a MySQL database. By applying object-oriented principles like encapsulation and modularity, the system ensures scalability and maintainability. Features like date input using jdatepicker demonstrate Java's extensibility, making this project an effective solution for automating gym operations. It exemplifies Java's practicality in building small-scale, database-driven applications.

2. Introduction

The Gym Management System is a Java-based application designed to automate gym operations, such as managing member data, tracking payments, and assigning trainers. Built with Java Swing for the user interface and JDBC for MySQL database integration, it allows efficient handling of gym-related information. The system follows object-oriented principles, ensuring modularity and scalability. Key features include a date selection component using the jdatepicker library. This solution simplifies manual tasks, improves data accuracy, and is suitable for small to medium-sized gyms.

3. Purpose

The primary goal of the Gym Management System is to create an efficient and user-friendly platform that simplifies gym operations for administrators. The objectives of the project include:

- Automating Gym Operations: Streamlining tasks like member registration, payment tracking, and trainer assignments to reduce administrative workload.
- Providing a User-Friendly Interface: Creating an intuitive interface with Java Swing for easy use by gym staff, regardless of technical expertise.
- Ensuring Data Accuracy: Using JDBC for secure database access, ensuring consistent and reliable member, payment, and trainer data.
- Improving Efficiency: Minimizing errors and time spent on manual tasks like payment tracking and scheduling.
- Providing Scalability: Designing the system with object-oriented principles for easy expansion and maintenance as the gym grows.

4. Scope of the Project:

The Gym Management System aims to automate and streamline various operations within a gym, providing a comprehensive solution for member management, payment tracking, and trainer assignments. The system includes features such as membership registration, payment tracking, scheduling, and trainer assignments, ensuring efficient data handling and reducing administrative workload. By utilizing Java Swing for the user interface and JDBC for secure database connectivity, the system ensures a smooth and user-friendly experience for gym staff. The database integration with MySQL ensures secure and consistent data management. The system is scalable, allowing for future expansion, such as adding features like online booking or enhanced reporting tools, to meet evolving gym needs.

5. Software Requirement Specification:

Introduction

This section defines the hardware and software requirements for the Gym Management System, ensuring smooth operation, scalability, and efficient user interaction.

Product Scope

The system automates key gym operations such as member registration, payment tracking, and trainer assignments. Developed with Java for logic and MySQL for database management, it offers a seamless user experience for both gym staff and members. The system is designed to be scalable, supporting future features like online booking and advanced reporting.

References and Acknowledgements

- Java Development Kit (JDK) for application development.
- MySQL documentation for database integration.
- Java Swing for the user interface.
- JDBC API for secure database connectivity.

Overall Description

The Gym Management System is a software solution developed using Java and MySQL to automate and streamline gym operations. It manages tasks such as member registration, payment tracking, trainer assignments, and scheduling. The system integrates a secure MySQL database via JDBC for reliable data storage, and the Java Swing interface ensures ease of use for gym staff. Scalable and maintainable, the system is designed to handle the evolving needs of small to medium-sized gyms, with the potential for future enhancements like online booking and advanced reporting.

Product Perspective

The Gym Management System is a Java and MySQL-based solution that automates and streamlines gym operations. It allows administrators to manage member registrations, payments, trainer assignments, and class schedules efficiently. The system uses JDBC for secure database connectivity and features a Java Swing interface for ease of use. Scalable and cost-effective, it is designed for small to medium-sized gyms and can be expanded with future features like online booking or advanced reporting. This system can be easily integrated into existing gym management processes, offering a modern solution for operational efficiency.

Product Functionality

- Member Registration: Allows members to register and update their profiles.
- Payment Tracking: Tracks member payments and dues in real-time.
- Trainer Assignment: Assigns trainers to members and manages schedules.
- Scheduling: Enables booking and managing gym classes or sessions.
- Database Integration: Uses MySQL for secure data storage and access.
- User Interface: Provides an intuitive Java Swing interface for easy navigation.
- Scalability: Supports future features like online booking and reporting.

User and Characteristics

- **Qualification**: Users should have at least basic educational qualifications, such as matriculation, and be comfortable with English for understanding system instructions and managing data effectively.
- **Experience**: Familiarity with gym operations or membership management is beneficial, though not required. Administrators will benefit from prior experience in managing schedules, payments, and memberships.
- **Technical Experience**: Users are expected to have elementary knowledge of computers and should be comfortable with operating basic software tools (like navigating windows, entering data into forms, etc.) to interact with the system efficiently.

Operating Environment

Hardware Requirements

- Processor: Intel i3 or higher (or equivalent AMD processor)
- Operating System: Windows 8,10, 11
- Processor Speed: 2.0 GHz
- RAM: 4GB
- Hard Disk: 500GB

Software Requirements

- System Requirements
- Software Requirements
- Database Requirements
- Development Tools
- Libraries and Frameworks
- Network Requirements

Constraints

- **System Access: Access is limited to authorized personnel only, ensuring that sensitive data is** protected and that the system is used by designated staff members only.
- Data Volume: The system is designed to handle a moderate volume of data, such as member details, payments, and schedules. However, for very large datasets (e.g., large gym chains), additional optimizations may be required to maintain system performance.
- Internet Connectivity: The system requires a stable internet connection for updates, remote access, and future integration of online features, which could be a limitation in regions with unreliable internet service.
- User Knowledge: Users should have basic computer literacy and some familiarity with gym operations. Those with limited technical expertise might require some training to navigate the system effectively.
- Hardware Limitations: The system is optimized for computers with at least 4 GB RAM and 2 GB disk space. Performance may degrade on systems with lower specifications.

User Interface

The Medical E-Commerce Store provides user-friendly, menu-driven interfaces for:

- **Register**: Allows new users to create an account by providing essential information like name, contact details, and membership preferences.
- **Login**: Enables existing users (administrators, trainers, or members) to log in using their credentials (username and password) for personalized access.
- **Member Dashboard**: Displays personalized details like membership status, upcoming sessions, payment history, and profile settings.
- **Trainer Assignment**: Admins can assign trainers to specific members and track their schedules.
- **Payment Tracking**: Admins can view and manage member payments, ensuring that dues are up to date.
- **Scheduling**: Allows users (administrators and members) to view and manage gym sessions, including booking and canceling classes or training sessions.
- **Order/Transaction History**: Members can view and track previous transactions, payments, and appointments, providing transparency and easy access to their gym-related activities.

Hardware Interface

- Screen resolution of at least 640 x 480 or above.
- Compatible with any version of Windows 8, 10, 11.

Software Interface

- Login and Registration Screens
- Admin Dashboard
- Member Dashboard
- Trainer Dashboard
- Scheduling Interface
- Payment Interface

Functional Requirements

User Registration and Authentication:

- Allow users (administrators, trainers, members) to register by providing details like name, email, password, and contact information.
  Provide secure login/logout functionality with encrypted password storage to ensure safe access.
- Gym Member Management:
  Enable administrators to view, update, and manage gym member profiles, including membership status, payment history, and session details.
- Trainer Assignment:
  Allow administrators to assign trainers to members and manage trainer schedules.
  Trainers should be able to view and update their assigned sessions and manage progress tracking.
- Session Scheduling:
  Allow members to book or cancel gym classes and personal training sessions through a scheduling interface.
  Admins can manage and update the class schedule.
- Payment Tracking:
  Enable tracking of member payments for membership and training services.
  Allow members to view their payment status and history.
- Admin Functions:

Provide admins with tools to manage member, trainer, and session data.Allow for reports on payments, member activity, and session attendance.

Non-functional Requirements

- Performance:
  The system should load within 2-3 seconds and handle a high volume of simultaneous users without delays or crashes.

- Scalability:
  The system must be able to accommodate increasing user data, transactions, and gym schedules without significant performance issues.

- Security:
  Ensure that all user data is securely stored and transmitted, using encryption methods like SSL for login and payment processes.

- Availability:
  The system should be available 99.9% of the time, with minimal downtime for scheduled maintenance.

- Usability:
  The system should offer a user-friendly interface that is intuitive for administrators, trainers, and members, ensuring ease of navigation and efficient task completion.

- Reliability:
  The system should be stable and consistently perform as expected with minimal downtime or errors.

- Compatibility:
  The system should be compatible with major browsers (Chrome, Firefox, Safari) and devices (desktops, tablets, smartphones) to ensure broad accessibility.

2. SYSTEM FLOW DIAGRAMS

Figure 2.1 Use Case Diagrams

Figure 2.2 Entity Relationship Diagram

Figure 2.3 Data-flow diagram

3. MODULE DESCRIPTION

- **User Management**:
  Handles user registration, login, and profile management for both gym members and administrators. Ensures secure authentication and role-based access.
- **Member Management**:
  Allows admins to manage member profiles, including personal details, membership status, payment history, and session bookings.
- **Session Scheduling and Trainer Assignment**:
  Admins can assign trainers to members, manage schedules, and track session attendance. Members can book and view sessions.
- **Payment and Checkout**:
  Integrates secure payment gateways for processing transactions, including membership and session fees, using various payment methods.
- **Admin Dashboard**:
  Provides admins with an interface to manage sales, track inventory, monitor member activity, and generate reports.
- **Inventory and Equipment Management**:
  Tracks gym equipment and resources, allowing admins to manage stock levels and ensure necessary supplies are available for members.

DESIGN:

Figure 4.1.1 Login Page

Figure 4.1.3 Home page

Figure 4.1.4 Add member page

Figure 4.1.5 Payment page

4.1 Database Design

The database design for the **Gym Management System** includes several key tables that store and manage data across different modules:

- **Users Table**: Stores user details for members, trainers, and administrators, including personal information, roles, and login credentials.
- **Sessions Table**: Manages details about gym sessions, such as time, trainer assignments, and member bookings.
- **Payments Table**: Tracks member payments for memberships, sessions, and other services, ensuring financial data is accurately stored.
- **Inventory Table**: Tracks gym equipment and resources, including stock levels and updates, to ensure availability for members.

4.2 IMPLEMENTATIONS (CODE)

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.gymmanagementsystem;

import database.ConnectionProvider;
import java.awt.Color;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.ImageIcon;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

/**
 *
 * @author quach
 */
public class benefitsPanel extends javax.swing.JPanel {


    /**
     * Creates new form NewJPanel
     */

    public benefitsPanel() {
        initComponents();

        basic1.setSelected(false);
        basic2.setSelected(false);
        plus1.setSelected(false);
        plus2.setSelected(false);
        plus3.setSelected(false);
        premium1.setSelected(false);
        premium2.setSelected(false);
        premium3.setSelected(false);
        premium4.setSelected(false);

    }
```

```
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        plus1 = new javax.swing.JCheckBox();
        basic1 = new javax.swing.JCheckBox();
        basic2 = new javax.swing.JCheckBox();
        plus2 = new javax.swing.JCheckBox();
        plus3 = new javax.swing.JCheckBox();
        premium1 = new javax.swing.JCheckBox();
        premium2 = new javax.swing.JCheckBox();
        premium3 = new javax.swing.JCheckBox();
        premium4 = new javax.swing.JCheckBox();
        jComboBox1 = new javax.swing.JComboBox<>();

        setPreferredSize(new java.awt.Dimension(451, 571));

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 36)); // NOI18N
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/icons/benefiticon.png"))); // NOI18N
        jLabel1.setText("BENEFITS");

        plus1.setBackground(new java.awt.Color(204, 204, 204));
        plus1.setText("ALL GROUP EXERCISE CLASSES");

        basic1.setBackground(new java.awt.Color(204, 204, 204));
        basic1.setText("USE OF ALL STRENGTH EQUIPMENT");

        basic2.setBackground(new java.awt.Color(204, 204, 204));
        basic2.setText("USE OF ALL CARDIO EQUIPMENT");

        plus2.setBackground(new java.awt.Color(204, 204, 204));
  plus2.setText("1 PERSONAL TRAINING SESSION");

        plus3.setBackground(new java.awt.Color(204, 204, 204));
        plus3.setText("1 GROUP TRAINING SESSION");
```

```
        premium1.setBackground(new java.awt.Color(204, 204, 204));
        premium1.setText("USE OF BASKETBALL COURTS");

        premium2.setBackground(new java.awt.Color(204, 204, 204));
        premium2.setText("USE OF RACQUETBALL COURTS");

        premium3.setBackground(new java.awt.Color(204, 204, 204));
        premium3.setText("UNLIMITED STUDIO CYCLING");

        premium4.setBackground(new java.awt.Color(204, 204, 204));
        premium4.setText("UP TO TWO GUESTS PER VISIT");

        jComboBox1.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] "Basic",
"Plus", "Premium" }));
        jComboBox1.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                jComboBox1ItemStateChanged(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                    .addComponent(basic1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(plus1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(basic2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(plus2, javax.swing.GroupLayout.DEFAULT_SIZE,
                .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(29, 29, 29)
                .addComponent(basic1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(basic2)
                .addGap(18, 18, 18)
```

```
            .addComponent(plus1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(plus2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(plus3)
            .addGap(18, 18, 18)
            .addComponent(premium1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(premium2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(premium3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(premium4)
            .addContainerGap(151, Short.MAX_VALUE))
    );
}// </editor-fold>//GEN-END:initComponents
```

CONCLUSION

The Gym Management System is an efficient and comprehensive solution designed to automate and streamline the day-to-day operations of a gym. It simplifies tasks such as user management, session scheduling, trainer assignments, payment tracking, and inventory management. By leveraging Java and MySQL, the system ensures data integrity, secure transactions, and easy scalability to accommodate future growth.

With its user-friendly interface and powerful backend, this system improves operational efficiency, reduces administrative workload, and enhances the overall gym experience for both members and staff. Additionally, its flexible design allows for easy expansion and integration of new features, making it a reliable solution for managing small to medium-sized gyms. The system not only meets the current needs of gym operations but also provides a foundation for future advancements in the fitness industry.

5. REFERENCES

Books:
Head First Java by Kathy Sierra and Bert Bates – Beginner-friendly Java guide.
Java: The Complete Reference by Herbert Schildt – Comprehensive Java reference.
Database System Concepts by Abraham Silberschatz et al. – Covers database design.
User Interface Design and Evaluation by Debbie Stone et al. – Focuses on UI/UX principles.
Websites:
Oracle Java Tutorials: https://docs.oracle.com/javase/tutorial/

SQL Basics (W3Schools): https://www.w3schools.com/sql/

Java Swing (Javatpoint): https://www.javatpoint.com/java-swing

Software Development Life Cycle (GeeksforGeeks):
https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/