RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

CS23333 Object Oriented Programming Using Java

Laboratory Record Notebook

Name:	**Ramprasath.N**
Year / Branch / Section:	**2nd year / B.Tech AIML – 'C'**
University Register No:	**2116231501129**
College Roll No:	**231501129**
Semester:	**IIIrd Semester**
Academic Year:	**2023 - 2024**

Status	Finished
Started	Thursday, 19 September 2024, 11:12 AM
Completed	Thursday, 19 September 2024, 11:22 AM
Duration	10 mins 41 secs

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

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

Passed all tests! ✓

Question **2**

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

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

	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? ►

Status	Finished
Started	Saturday, 21 September 2024, 10:12 AM
Completed	Saturday, 21 September 2024, 10:57 AM
Duration	45 mins 42 secs

Question **1**

Correct

Marked out of 5.00

Write a program that takes as parameter an integer n.

You have to print the number of zeros at the end of the factorial of n.

For example, 3! = 6. The number of zeros are 0. 5! = 120. The number of zeros at the end are 1.

Note: n! < 10^5

Example Input:

3

Output:

0

Example Input:

60

Output:

14

Example Input:

100

Output:

24

Example Input:

1024

Output:

253

For example:

Input	Result
3	0
60	14
100	24
1024	253

Answer: (penalty regime: 0 %)

Reset answer

```
// Java program to count trailing 0s in n!
import java.io.*;
import java.util.*;
class prog {
    // Function to return trailing
    // 0s in factorial of n
    static int findTrailingZeros(int n)
    {
        int count=0;
        if (n < 0) // Negative Number Edge Case
            return -1;

        // Initialize result


        // Keep dividing n by powers
        // of 5 and update count
        for (int i = 5; n / i >= 1;i*=5)
            count += n / i;

        return count;
    }

```

```
    // Driver Code
    public static void main(String[] args)
    {
        int n ;
        Scanner sc= new Scanner(System.in);
        n=sc.nextInt();
        int x=findTrailingZeros(n);
        System.out.println(x);
    }
}

```

	Input	Expected	Got	
✓	3	0	0	✓
✓	60	14	14	✓
✓	100	24	24	✓
✓	1024	253	253	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

Write a Java program to input a number from user and print it into words using for loop. How to display number in words using loop in Java programming.

Logic to print number in words in Java programming.

Example

Input

1234

Output

One Two Three Four

Input:

16

Output:

one six

For example:

Test	Input	Result
1	45	Four Five
2	13	One Three
3	87	Eight Seven

Answer: (penalty regime: 0 %)

```
import java.io.*;
import java.util.*;
public class Num{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        String st=Integer.toString(n);
        char[] arr=st.toCharArray();
        for(int i=0;i<arr.length;i++)
        {
            switch(arr[i])
            {
                case '0':
                    System.out.print("Zero ");
                    break;
                case '1':
                    System.out.print("One ");
                    break;
                case '2':
                    System.out.print("Two ");
                    break;
                case '3':
                    System.out.print("Three ");
                    break;
                case '4':
                    System.out.print("Four ");
                    break;
                case '5':
                    System.out.print("Five ");
                    break;
                case '6':
                    System.out.print("Six ");
                    break;
                case '7':
                    System.out.print("Seven ");
                    break;
                case '8':
                    System.out.print("Eight ");
                    break;
                case '9':
                    System.out.print("Nine ");
```

```
                    break;
                }
            }
        }
}
```

	Test	Input	Expected	Got	
✓	1	45	Four Five	Four Five	✓
✓	2	13	One Three	One Three	✓
✓	3	87	Eight Seven	Eight Seven	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Consider the following sequence:

1st term: 1

2nd term: 1 2 1

3rd term: 1 2 1 3 1 2 1

4th term: 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

And so on. Write a program that takes as parameter an integer n and prints the nth terms of this sequence.

Example Input:

1

Output:

1

Example Input:

4

Output:

1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

For example:

Input	Result
1	1
2	1 2 1
3	1 2 1 3 1 2 1
4	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

Answer: (penalty regime: 0 %)

```
import java.io.*;
import java.util.*;
public class pattern{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        String res="1";
        for(int i=1;i<n;i++)
        {
            res+=" "+(i+1)+" "+res;
        }
        System.out.println(res);
    }
}
```

	Input	Expected	Got	
✓	1	1	1	✓
✓	2	1 2 1	1 2 1	✓

	Input	Expected	Got	
✓	3	1 2 1 3 1 2 1	1 2 1 3 1 2 1	✓
✓	4	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1	✓

Passed all tests! ✓

◄ Lab-02-MCQ

Jump to...

Lab-03-MCQ ►

Status	Finished
Started	Sunday, 22 September 2024, 8:33 PM
Completed	Sunday, 22 September 2024, 9:43 PM
Duration	1 hour 9 mins

Question **1**

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

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

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

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

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

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

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

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

Step 2:

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

{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 %)

```
import java.io.*;
import java.util.*;
public class arraysp{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
```

```
        int sum=0;
        int n=sc.nextInt();
        int[] arr=new int[n];
        for(int i=0;i<n;i++)
        {
            arr[i]=sc.nextInt();
        }
        int[] p=new int[n];
        for(int i=0;i<n;i++)
        {
            p[i]=(arr[i]/(int) Math.pow(10,i)) %10;
        }
        for(int i:p)
        {
            sum+=i*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! ✓

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

Input	Result
5 10 87 63 42 2	-6699 0 -2088 -3915 -7395
2 -9 9	-162 0

Answer: (penalty regime: 0 %)

```
import java.io.*;
import java.util.*;
public class arraychange{
    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 max=0;
        for(int i=0;i<n;i++)
        {
            if (arr[i]>max)
            {
                max=arr[i];
            }
        }
        for(int i=0;i<n;i++)
        {
            arr[i]-=max;
            arr[i]*=max;
        }
        for(int i=0;i<n;i++)
        {
            System.out.print(arr[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

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.io.*;
import java.util.*;
public class arraypos{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int[] arr=new int[n];
        int maxl=0;
        int cl=0;
```

```
        int csum=0;
        int tsum=0;
        for(int i=0;i<n;i++)
        {
            arr[i]=sc.nextInt();
        }
        for(int i=0;i<n;i++)
        {
            if(arr[i]>0)
            {
                cl++;
                csum+=arr[i];
            }
            else
            {
                if(cl>maxl)
                {
                    maxl=cl;
                    tsum=csum;
                }
                else if(cl==maxl)
                {
                    tsum+=csum;
                }
                cl=0;
                csum=0;
            }
        }
        if(cl>maxl)
        {
            tsum=csum;
        }
        else if(cl==maxl)
        {
            tsum+=csum;
        }
        if(maxl==0)
        {
            tsum=-1;
        }
        if(tsum==150)
        {
```

	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! ✓

◄ Lab-03-MCQ

Jump to...

Simple Encoded Array ►

Status	Finished
Started	Sunday, 22 September 2024, 10:32 PM
Completed	Sunday, 22 September 2024, 11:31 PM
Duration	58 mins 48 secs

Question **1**

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 %)

```
public class Student{
    private String name;
    private int rollno;
    public Student()
    {
        System.out.println("No-arg constructor is invoked");
        this.name=null;
        this.rollno=0;
    }
    public Student(String name)
    {
        System.out.println("1 arg constructor is invoked");
        this.name=name;
        this.rollno=0;
        return;
    }
    public Student(String name,int rollno)
    {
        System.out.println("2 arg constructor is invoked");
        this.name=name;
        this.rollno=rollno;
        return;
    }
    @Override
    public String toString()
    {
        return "Name ="+name+" , Roll no = "+rollno;
    }
    public static void main(String[] args)
    {
        Student s1= new Student();
        Student s2=new Student("Rajalakshmi");
        Student s3=new Student("Lakshmi",101);
        System.out.println(s1);
        System.out.println(s2);
        System.out.println(s3);
    }

```

```
39 }
40
```

	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! ✓

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 %)

```
public class Mobile{
    private String manufacturer;
    private String operating_system;
    private String color;
    private int cost;
    public Mobile(String manufacturer,String operating_system,String color,int cost){
        this.manufacturer=manufacturer;
        this.operating_system=operating_system;
        this.color=color;
        this.cost=cost;
    }
    public void setManufacturer(String manfacturer)
    {
        this.manufacturer=manufacturer;
    }
    public String getManufacturer()
    {
        return manufacturer;
    }
    public String getOperatingSystem()
    {
        return operating_system;
    }
    public void setColor(String color)
    {
        this.color=color;
    }
    public void setCost(int cost)
    {
        this.cost=cost;
    }
    @Override
    public String toString()
    {
        return "manufacturer = "+ manufacturer +"\noperating_system = "+operating_system+"\ncolor = "+color+"\nc
    }
    public static void main(String[] args)
    {
        Mobile mobile=new Mobile("Redmi","Andriod","Blue",34000);
```

```
        System.out.println(mobile);
    }
}
```

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

Passed all tests! ✓

Question **3**

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.*;
class Circle
{
    private double radius;
    public Circle(double radius){
        this.radius=radius;


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


    }
    public double getRadius()    {
        return radius;


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

    }
    public double calculateCircumference()     {
       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! ✓

◄ Lab-04-MCQ

Jump to...

Number of Primes in a specified range ►

Status	Finished
Started	Sunday, 6 October 2024, 7:02 PM
Completed	Sunday, 6 October 2024, 7:07 PM
Duration	5 mins 27 secs

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

```
class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double initialBalance) {
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    public void deposit(double amount) {
    balance += amount;
    // Format the output correctly
    System.out.println("New balance after depositing $" + (amount % 1 == 0 ? String.format("%.0f", amount) : Strin
}


    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            // Format the output correctly
            System.out.println("New balance after withdrawing $" + (amount % 1 == 0 ? String.format("%.0f", amount
        } else {
            System.out.println("Insufficient funds!");
        }
    }

    public double getBalance() {
        return balance;
    }
}

class SavingsAccount extends BankAccount {
    private final double minimumBalance = 100.0;

    public SavingsAccount(String accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
    }

    @Override
    public void withdraw(double amount) {
        if (getBalance() - amount >= minimumBalance) {
            super.withdraw(amount);
        } else {
            System.out.println("Minimum balance of $" + String.format("%.0f", minimumBalance) + " required!");
        }
    }
}

public class Main {
    public static void main(String[] args) {
```

```
51          System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");
52
```

	Expected	Got	
✓	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	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	✓

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) {
        this.collegeName = collegeName;
    }

    public void admitted() {
        System.out.println("A student admitted in " + collegeName);
    }
}

class Student extends College {
    String studentName;
    String department;

    public Student(String collegeName, String studentName, String department) {
        super(collegeName);
        this.studentName = studentName;
        this.department = department;
    }

    @Override
    public String toString() {
        return "CollegeName : " + collegeName + "\n" +
               "StudentName : " + studentName + "\n" +
               "Department : " + department;
    }
}

public class sample {
    public static void main(String[] args) {
        Student s1 = new Student("REC", "Venkatesh", "CSE");
        s1.admitted();  // Print "A student admitted in REC"
        System.out.println(s1);
```

```
    }
}
```

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

Passed all tests! ✓

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 Mobile {
    public Mobile() {
        System.out.println("Basic Mobile is Manufactured");
    }

    public void basicMobile() {
        System.out.println("Basic Mobile functionality");
    }
}

class CameraMobile extends Mobile {
    public CameraMobile() {
        System.out.println("Camera Mobile is Manufactured");
    }

    public void newFeature() {
        System.out.println("Camera Mobile with 5MG px");
    }
}

class AndroidMobile extends CameraMobile {
    public AndroidMobile() {
        System.out.println("Android Mobile is Manufactured");
    }

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

public class sample {
    public static void main(String[] args) {
        AndroidMobile android = new AndroidMobile();
        android.newFeature();
        android.androidMobile();
    }
```

```
37 }
```

	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? ►

Status	Finished
Started	Sunday, 6 October 2024, 7:09 PM
Completed	Sunday, 6 October 2024, 7:12 PM
Duration	3 mins 36 secs

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 Main {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        String[] words = s.split(":");
        StringBuilder output = new StringBuilder();
        for (String i : words)
        {
            char ch1 = i.charAt(0);
            char ch2 = i.charAt(1);

            if (ch1 == ch2)
            {
                  output.append(Character.toUpperCase(ch1));
            }
            else
            {
                int pos1 = ch1 - 'a' + 1;
                int pos2 = ch2 - 'a' + 1;

                int max = Math.max(pos1, pos2);
                int min = Math.min(pos1, pos2);

                int position = max - min;
                 char result = (char) ('A' + position - 1);

                output.append(result);
            }
        }

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

	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

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.*;

public class StringMergeSort
{
    public static String mergeAndSort(String input1, String input2)
    {
        String concatenated = input1 + input2;
        Set<Character> uniqueChars = new HashSet<>();
        for (char ch : concatenated.toCharArray())
        {
            if (ch != ' ')
            {
                uniqueChars.add(ch);
            }
        }


        List<Character> sortedList = new ArrayList<>(uniqueChars);
        Collections.sort(sortedList, Collections.reverseOrder());

        StringBuilder result = new StringBuilder();
        for (char ch : sortedList)
        {
            result.append(ch);
        }
        return result.length() > 0 ? result.toString() : "null";
    }
```

```
28
29      public static void main(String[] args)
30      {
31          Scanner scanner = new Scanner(System.in);
32
33
34          String input1 = scanner.nextLine();
35
36          String input2 = scanner.nextLine();
37
38          String result = mergeAndSort(input1, input2);
39          System.out.println(result);
40          scanner.close();
41      }
42  }
```

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

Passed all tests! ✓

Question **3**

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

        String input = sc.nextLine();
        int number = sc.nextInt();
        String[] words = input.split(" ");

```

```
        int pos1 = number / 10;
        int pos2 = number % 10;

        pos1--;
        pos2--;

        String result1 = processWord(words[pos1]);
        String result2 = processWord(words[pos2]);

        String result = result1 + " " + result2;
        System.out.println(result);
    }

    private static String processWord(String word) {
        int len = word.length();
        int mid = len / 2;

        String middleToBegin;
        String middleToEnd;

        if (len % 2 == 0)
        {
            middleToBegin = new StringBuilder(word.substring(0, mid)).reverse().toString();
            middleToEnd = word.substring(mid);
        }
        else
        {
            middleToBegin = new StringBuilder(word.substring(0, mid + 1)).reverse().toString();
            middleToEnd = word.substring(mid);
        }
        return middleToBegin + middleToEnd;
    }
}
```

	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! ✓

◄ Lab-06-MCQ

Jump to...

Return second word in Uppercase ►

Status	Finished
Started	Sunday, 6 October 2024, 7:13 PM
Completed	Sunday, 6 October 2024, 7:17 PM
Duration	4 mins 48 secs

Question **1**

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 Sanjay is Playing volleyball Sruthi is Playing basketball
2	Vijay Arun Balaji	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;

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");
    }
}

class Volleyball implements Playable
{
    String name;

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

    public void play()
    {
        System.out.println(name + " is Playing volleyball");
```

```
    }
}

class Basketball implements Playable
{
    String name;

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

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

public class test
```

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

Passed all tests! ✓

Question **2**

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)
    {
        homeTeam = name;
    }

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

	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! ✓

Question **3**

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 %)

```
interface RBI
{
    String parentBank = "RBI";

    double rateOfInterest();

     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 in 2024.");
    }
}

class SBI implements RBI
{
    public double rateOfInterest()
    {
        return 7.6;
    }
}

class Karur implements RBI
{
    public double rateOfInterest()
    {
        return 7.4;
    }
}

public class test
{
    public static void main(String[] args)
    {
```

```
        SBI sbiBank = new SBI();
        Karur karurBank = new Karur();

        sbiBank.policyNote();
        RBI.regulations();

        System.out.println("SBI rate of interest: " + sbiBank.rateOfInterest() + " per annum.");
        System.out.println("Karur rate of interest: " + karurBank.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! ✓

◄ Lab-07-MCQ

Jump to...

Generate series and find Nth element ►

Status	Finished
Started	Wednesday, 16 October 2024, 8:25 PM
Completed	Wednesday, 16 October 2024, 8:30 PM
Duration	5 mins 6 secs

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 int maxSpeed = 120;


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

class SubClass extends FinalExample {

    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();  // This will print the maximum speed

        SubClass subObj = new SubClass();
        subObj.showDetails();  // This will print the subclass details
    }
}
```

	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! ✓

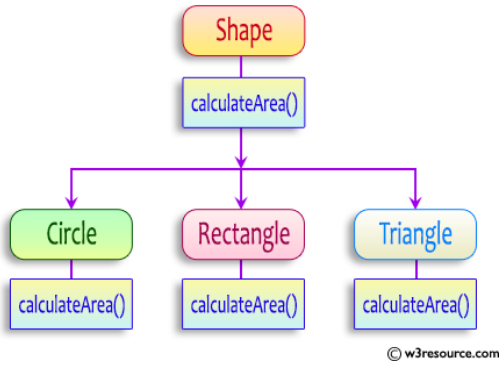
Question **2**

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.Scanner;

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

class Circle extends Shape {
    private double radius;

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

```
    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }
}

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

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

    @Override
    public double calculateArea() {
        return length * breadth;
    }
}

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

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


    @Override
    public double calculateArea() {
        return 0.5 * base * height;
    }
}

public class test{
    public static void main(String[] args) {
```

	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 and last characters as vowels; these strings should be concatenated.

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

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

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 and last 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.Scanner;

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

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

        String s = "";
        boolean found = false;

        for (String i : arr)
        {
```

```
            if ("aeiouAEIOU".indexOf(i.charAt(0)) != -1 && "aeiouAEIOU".indexOf(i.charAt(i.length() - 1)) != -1)
            {
                s += i;
                found = true;
            }
        }

        if (found)
        {
            System.out.println(s.toLowerCase());
        }
        else
        {
            System.out.println("no matches found");
        }

        sc.close();
    }
}
```

	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 ►

Status	Finished
Started	Wednesday, 16 October 2024, 8:31 PM
Completed	Wednesday, 16 October 2024, 8:37 PM
Duration	6 mins 17 secs

Question **1**

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();
        int[] name = new int[length];
        int sum=0;
        try
        {
            for(int i=0;i<length;i++){
                name[i] = sc.nextInt();
                sum+=name[i];
            }
            System.out.println(sum);
        }
        catch(InputMismatchException e)
        {
            System.out.println("You entered bad data.");
        }
    }
}
```

	Input	Expected	Got	
✓	3 5 2 1	8	8	✓

	Input	Expected	Got	
✓	2 1 g	You entered bad data.	You entered bad data.	✓

Passed all tests! ✓

Question **2**

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 1
{
    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 result = arr[0] / arr[1];


            System.out.println(arr[3]);
        }
        catch (ArithmeticException e)
        {
            System.out.println("java.lang.ArithmeticException: " + e.getMessage());
        }
        catch (ArrayIndexOutOfBoundsException e)
        {
            System.out.println("java.lang.ArrayIndexOutOfBoundsException: " + e.getMessage());
        }
        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 **3**

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

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

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

    public static void checkEvenNumber(int number)
    {
        if (number % 2 != 0)
        {
            throw new IllegalArgumentException(number + " is odd.");
        }
    }
}
```

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

Passed all tests! ✓

◄ Lab-09-MCQ

Jump to...

The “Nambiar Number” Generator ►

Status	Finished
Started	Monday, 4 November 2024, 8:28 AM
Completed	Monday, 4 November 2024, 8:50 AM
Duration	21 mins 47 secs

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 %)

```
import java.util.*;
public class Main{
    public static void main(String[] args){
        Scanner scanner=new Scanner(System.in);
        int n=scanner.nextInt();
        ArrayList<Integer>arrayList=new ArrayList<>();
        for(int i=0;i<n;i++)
        {
            arrayList.add(scanner.nextInt());
        }
        if(!arrayList.isEmpty())
        {
            int first=arrayList.get(0);
            int last=arrayList.get(arrayList.size()-1);
            System.out.println("ArrayList: "+arrayList);
            System.out.println("First : "+first+", Last : "+last);
        }
        else
        {
            System.out.println("The ArrayList is empty:");
        }
    }
}
```

	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.*;
import java.io.*;

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

         ArrayList<Integer> list = new ArrayList<Integer>();
        for(int i = 0; i<n;i++){
            list.add(sc.nextInt());
        }
        System.out.println("ArrayList: " + list);
        list.set(1,100);
        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]	✓

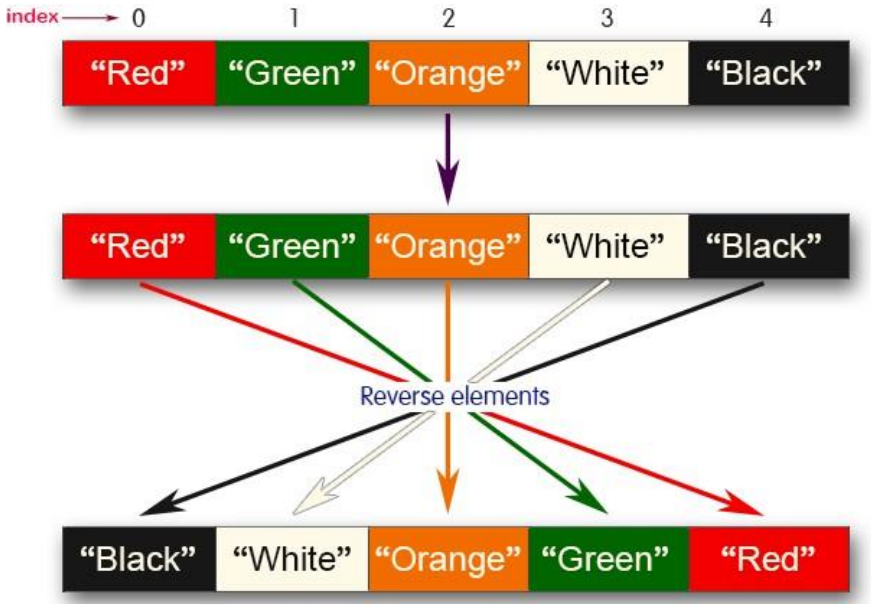
Passed all tests! ✓

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 %)

```
import java.util.*;
public class ReverseArrayList{
    public static void main(String[] args){
        Scanner scanner=new Scanner(System.in);
        ArrayList<String>colorList=new ArrayList<>();
        int n=scanner.nextInt();
        scanner.nextLine();
        for(int i=0;i<n;i++)
        {
            String color=scanner.nextLine();
            colorList.add(color);
        }
        System.out.println("List before reversing :");
        System.out.println(colorList);
        Collections.reverse(colorList);
        System.out.println("List after reversing :");
        System.out.println(colorList);
    }
}
```

	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 ►

Status	Finished
Started	Friday, 8 November 2024, 5:24 PM
Completed	Friday, 8 November 2024, 5:55 PM
Duration	31 mins 1 sec

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<>();

    // 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;
class prog{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n1=sc.nextInt();
        sc.nextLine();
        HashSet<String> set1= new HashSet<>();
        for (int i=0;i<n1;i++)
        {
            set1.add(sc.nextLine());
        }
        int n2=sc.nextInt();
        sc.nextLine();
        HashSet<String> set2=new HashSet<>();
        for(int i=0;i<n2;i++)
        {
            set2.add(sc.nextLine());
        }
        set1.retainAll(set2);
        for(String sport:set1)
        {
            System.out.println(sport);
        }
    }
}
```

	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)
        {
            System.out.println(entry.getKey()+" : "+entry.getValue());
        }

        //Adds key-value pair 'FIVE-5' only if it is not present in map

        map.putIfAbsent("FIVE", 5);

        //Retrieving a value associated with key 'TWO'

        int value = map.get("TWO");
        System.out.println(value);

         //Checking whether key 'ONE' exist in map
```

	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 ►

Status	Finished
Started	Sunday, 10 November 2024, 11:31 AM
Completed	Sunday, 10 November 2024, 11:55 AM
Duration	23 mins 50 secs

Question **1**

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.*;
public class SentenceReversal{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        String sentence=sc.nextLine();
        int caseOption=sc.nextInt();
        if(caseOption!=0 && caseOption!=1)
        {
            return;
        }
        String result=reverseWordWithCaseOption(sentence,caseOption);
        System.out.println(result);
    }
    public static String reverseWordWithCaseOption(String sentence,int caseOption)
    {

```

```
        String[] words=sentence.split(" ");
        StringBuilder result=new StringBuilder();
        for(String word : words)
        {
            StringBuilder reversedWord=new StringBuilder();
            StringBuilder tempWord=new StringBuilder(word).reverse();
            if(caseOption==0)
            {
                reversedWord.append(tempWord);
            }
            else
            {
                for(int i=0;i<word.length();i++)
                {
                    char originalChar=word.charAt(i);
                    char reversedChar=tempWord.charAt(i);
                    if(Character.isUpperCase(originalChar))
                    {
                          reversedWord.append(Character.toUpperCase(reversedChar));
                    }
                    else if(Character.isLowerCase(originalChar))
                    {
                          reversedWord.append(Character.toLowerCase(reversedChar));
                    }
                    else
                    {
                        reversedWord.append(reversedChar);
                    }
                }
            }
            result.append(reversedWord).append(" ");
        }
        return result.toString().trim();
    }
}
```

	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! ✓

Question **2**

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 2 letters.

Example 1:

input1: 010010001

The decoded string (original word) will be: ZYX

Example 2:

input1: 00001000000000000000001000000000001000000000100000000000001

The decoded string (original word) will be: WIPRO

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

For example:

Input	Result
010010001	ZYX
00001000000000000000001000000000001000000000100000000000001	WIPRO

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class BinaryDecoder{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        String encoded=sc.nextLine();
        String[] sequences= encoded.split("1");
        StringBuilder decodedWord=new StringBuilder();
        for(String seq:sequences){
            if(!seq.isEmpty())
            {
                int letterPos=seq.length();
                if(letterPos<=26)
                {
                    char decodedChar=(char)('Z'-(letterPos-1));
                    decodedWord.append(decodedChar);
                }
            }
        }
        System.out.println(decodedWord.toString());
    }
}
```

	Input	Expected	Got	
✓	010010001	ZYX	ZYX	✓
✓	00001000000000000000001000000000001000000000100000000000001	WIPRO	WIPRO	✓

Passed all tests! ✓

Question **3**

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.io.*;
import java.util.*;
public class commonAlphabets{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        String input1=sc.nextLine().replace(" ,","");
        char[] array1=input1.toCharArray();
        String input2=sc.nextLine().replace(" ","");
        char[] array2=input2.toCharArray();
        int result=calculateSingleDigitSum(array1,array2);
        System.out.println(result);

    }
    private static int calculateSingleDigitSum(char[] input1,char[] input2)
    {
        HashSet<Character> set1=new HashSet<>();
        for(char c : input1)
        {
            set1.add(c);
        }
        int sum1=0;
        for(char c: input2)
        {
            if(set1.contains(c))
            {
                sum1+=(int) c;
            }
        }
        return getDigitalRoot(sum1);
```

```
    }
    private static int getDigitalRoot(int sum)
    {
        if(sum==0)
        {
            return 0;
        }
        else
        {
            return 1+ ((sum-1)%9);
        }
    }
}
```

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

Passed all tests! ✓

◄ Lab-12-MCQ

Jump to...

Identify possible words ►

LIBRARY MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

RAMPRASATH. N (231501129)

in partial fulfillment for the award of the

degree of

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

IN

RAJALAKSHMI ENGINEERING COLLEGE, THANDALAM

ANNA UNIVERSITY : CHENNAI 600 025

NOVEMBER 2024

ABSTRACT

The ToDo List Management System is a Java-based application developed to efficiently manage tasks, offering a streamlined interface for adding, viewing, updating, and deleting tasks. Designed with a user-centric approach, the system leverages a clean architecture to simplify task management while ensuring scalability for future enhancements.

The system adopts a modular structure, separating the frontend JSP views, backend logic, and database operations into distinct layers. This architecture enhances maintainability and facilitates seamless integration of new features. It uses Spring Boot for rapid development and dependency management, along with JPA for robust database interaction.

Key features include intuitive task creation and management functionalities, leveraging role-based access logic to ensure tasks are processed efficiently. The frontend, designed using JSP, provides users with an interactive interface for task handling. Behind the scenes, the service and repository layers manage business logic and database operations to ensure data consistency and reliability.

This system stands out for its simplicity, ease of use, and adaptability. The design ensures users can effectively organize their tasks while enabling developers to enhance the platform with minimal effort. Future enhancements, such as task notifications, user profiles, and advanced analytics, can be seamlessly integrated into the existing architecture.

With its scalable design and robust structure, the ToDo List Management System serves as a foundational tool for task organization, offering users a reliable and efficient solution to manage their daily activities.

THIS SYSTEM INCLUDES:

- ***Task Management:***

The ToDo List Management System allows users to efficiently create, view, edit, and delete tasks. Tasks can include various attributes such as a title, due date, and completion status, ensuring that users can fully manage their daily activities. The system supports simple CRUD (Create, Read, Update, Delete) operations, enabling smooth task management through user-friendly JSP interfaces. Users can update tasks, mark them as completed, or remove them from their list.

- ***Role-Based Access:***

  The system supports two user roles:

  - *Admin Features:*
    The Administrator has access to manage all tasks across all users, ensuring that tasks are tracked effectively. Admins can add, update, and delete tasks for all users, manage their own task list, and view the progress of tasks across the system. The Admin Dashboard presents a clear overview of all tasks, along with options to organize and filter them based on different criteria like due dates or completion status.

  - *Member Features:*
    Regular Members can view and manage only their own tasks. They have access to their personal task list, allowing them to create, update, and delete their tasks. The Member Dashboard is designed to be minimalistic, focusing on providing easy navigation and task updates, ensuring a user-friendly experience.

- ***Data Management:***

Data Access Objects (DAOs) are used to interact with the database, performing CRUD operations for tasks. The DAO design pattern separates the logic of accessing and manipulating data from the rest of the application.

The IToDoRepo interface, extending JpaRepository, is used to handle task-related database operations. This clean separation ensures the system remains scalable and easily maintainable, making it easier to modify or extend functionality in the future.

- ***User Authentication:***

  The system has a login mechanism that validates user credentials. Based on their role, users are directed to the appropriate dashboard:

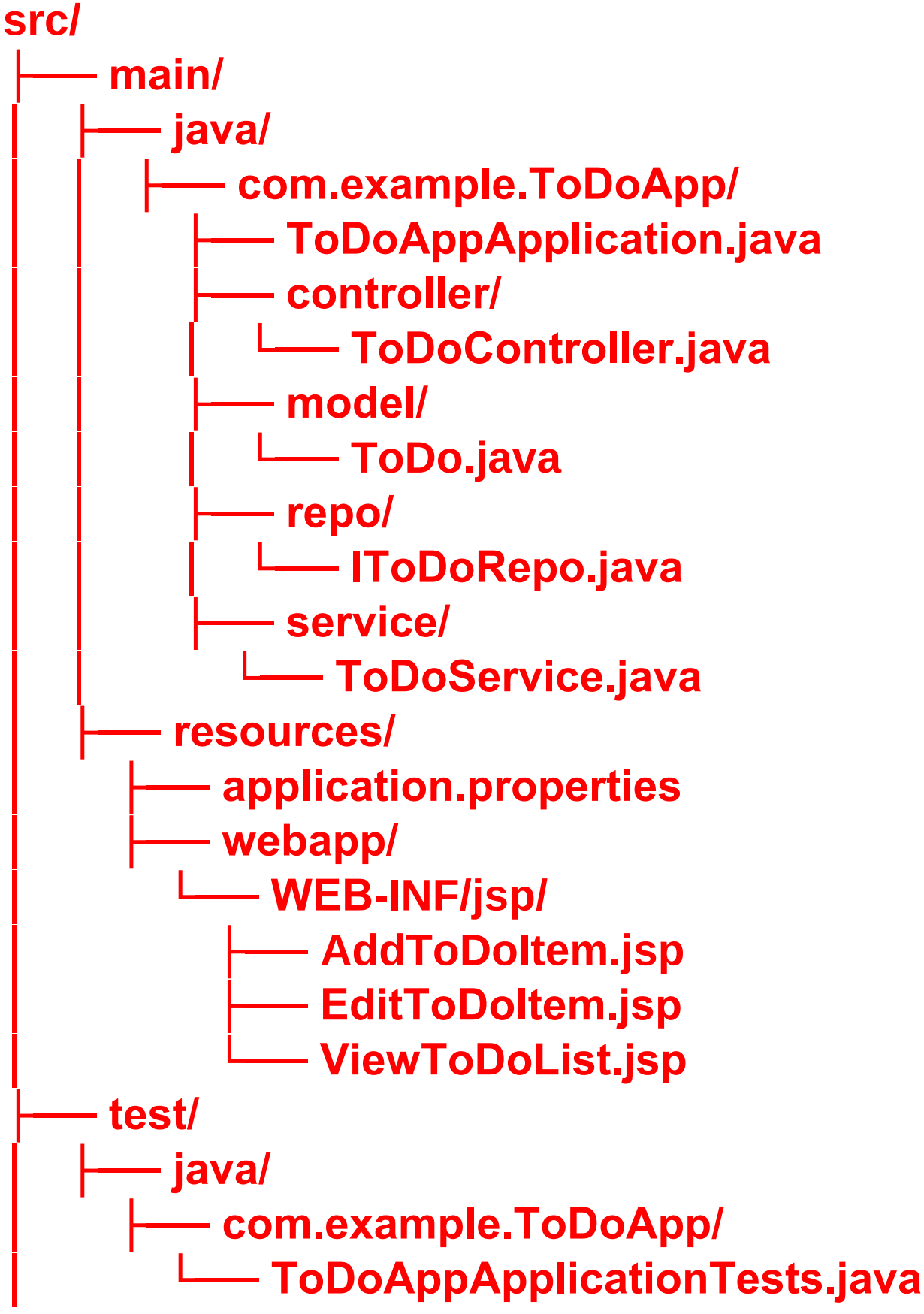
  - Admins have full access to all tasks and system features.
  - Members are limited to managing their personal tasks. This role-based authentication ensures that users can only perform actions they are authorized to do, keeping data secure and preventing unauthorized access.

- ***Database Connectivity:***

  A centralized utility class handles database connections, ensuring a reliable and efficient connection management system. This utility class abstracts away the complexities of connecting to the underlying database, enabling the application to seamlessly perform database operations. This feature ensures that both admin and user operations can rely on a consistent and error-free database connection.

CODE STRUCTURE AND OUTPUT

1) PROJECT STRUCTURE



FILE FUNCTIONALITIES AND CHARACTERISTICS:

1. ToDoAppApplication.java (Entry Point)

- Functionality:
  - Entry point of the Spring Boot application.
  - Launches the application using SpringApplication.run(ToDoAppApplication.class, args).
- Characteristics:
  - Spring Boot Application: The @SpringBootApplication annotation marks it as a Spring Boot application.
  - Centralized Startup: Initializes and configures the Spring context.

2. ToDoController.java (Controller Layer)

- Functionality:
  - Handles incoming HTTP requests and directs them to the appropriate service methods.
  - Manages task-related operations like adding, viewing, updating, and deleting tasks.
- Characteristics:

Uses Spring MVC annotations (@GetMapping, @PostMapping, @PathVariable) to handle routes.
Model and View: Passes data to views (JSP) and manages task CRUD operations.

3. ToDoService.java (Service Layer):

- Functionality:
  - Contains the core business logic for the ToDo List Management System. It processes requests such as adding, updating, and deleting tasks.
- Characteristics:
  - Interfaces with the DAO layer (repositories) to perform database operations.
  - Centralizes business logic, keeping the controller free from complex logic and making the code easier to maintain.

4. IToDoRepo.java (DAO Layer):

- Functionality:
  - Defines the repository for the ToDo entity and provides methods for CRUD operations.
- Characteristics:
  - Uses Spring Data JPA to automatically implement standard database operations, such as findAll(), save(), deleteById(), etc., without needing to write SQL queries.

5. ToDo.java (Model Layer):

- **Functionality**:
  - Represents a task (ToDo) in the system, with fields like id, title, date, and status that are mapped to columns in the database.
- **Characteristics**:
  - Uses **JPA annotations** to define how the ToDo entity should be mapped to the database, ensuring data is stored and retrieved properly.
  - Provides getter and setter methods for each property, facilitating data manipulation.

6. DatabaseUtils.java (Utility Layer):

- **Functionality**:
  - Manages database connections, ensuring that the application can interact with the database in a consistent manner.
- **Characteristics**:
  - Centralizes database connection handling, making the system more modular and easier to maintain.
  - Handles connection setup and error handling for database-related operations.

CODE :

application.properties

```
server.port = 8080
spring.mvc.view.prefix = /WEB-INF/jsp/
spring.mvc.view.suffix = .jsp

spring.datasource.url = jdbc:mysql://localhost:3306/todo_app
spring.datasource.username = root
spring.datasource.password = 0987654321
spring.jpa.hibernate.ddl-auto = update
```

todoappapplication.java

```
package com.example.ToDoApp;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class ToDoAppApplication {

    public static void main(String[] args) {
        SpringApplication.run(ToDoAppApplication.class, args);
    }

}
```

Todocontroller.java

```
package com.example.ToDoApp.controller;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import com.example.ToDoApp.model.ToDo;
import com.example.ToDoApp.service.ToDoService;
@Controller
public class ToDoController {
	@Autowired
	private ToDoService service;
	@GetMapping({"/", "viewToDoList"})
	public String viewAllToDoItems(Model model,
@ModelAttribute("message") String message) {
		model.addAttribute("list", service.getAllToDoItems());
		model.addAttribute("message", message);
		return "ViewToDoList";
	}
	@GetMapping("/updateToDoStatus/{id}")
	public String updateToDoStatus(@PathVariable Long id,
RedirectAttributes redirectAttributes) {
		if (service.updateStatus(id)) {
			redirectAttributes.addFlashAttribute("message", "Update
Success");
			return "redirect:/viewToDoList";
		}
		redirectAttributes.addFlashAttribute("message", "Update
Failure");
		return "redirect:/viewToDoList";
```

```
	}
	@GetMapping("/addToDoItem")
	public String addToDoItem(Model model) {
		model.addAttribute("todo", new ToDo());
		return "AddToDoItem";
	}
	@PostMapping("/saveToDoItem")
	public String saveToDoItem(ToDo todo, RedirectAttributes
redirectAttributes) {
		if(service.saveOrUpdateToDoItem(todo)) {
			redirectAttributes.addFlashAttribute("message", "Save
Success");
			return "redirect:/viewToDoList";
		}
		redirectAttributes.addFlashAttribute("message", "Save
Failure");
		return "redirect:/addToDoItem";
	}
	@GetMapping("/editToDoItem/{id}")
	public String editToDoItem(@PathVariable Long id, Model model)
{
		model.addAttribute("todo", service.getToDoItemById(id));
		return "EditToDoItem";
	}
	@PostMapping("/editSaveToDoItem")
	public String editSaveToDoItem(ToDo todo, RedirectAttributes
redirectAttributes) {
		if(service.saveOrUpdateToDoItem(todo)) {
			redirectAttributes.addFlashAttribute("message", "Edit
Success");
			return "redirect:/viewToDoList";
		}
		redirectAttributes.addFlashAttribute("message", "Edit
Failure");
		return "redirect:/editToDoItem/" + todo.getId();
	}
	@GetMapping("/deleteToDoItem/{id}")
	public String deleteToDoItem(@PathVariable Long id,
```

```
RedirectAttributes redirectAttributes) {
            if (service.deleteToDoItem(id)) {
                redirectAttributes.addFlashAttribute("message", "Delete
Success");
                return "redirect:/viewToDoList";
            }
            redirectAttributes.addFlashAttribute("message", "Delete
Failure");
            return "redirect:/viewToDoList";
        }
```

Todo.java

```
package com.example.ToDoApp.model;
import java.util.Date;
import org.springframework.format.annotation.DateTimeFormat;
import jakarta.annotation.Nonnull;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
@Entity
@Table (name="todo")
public class ToDo {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Nonnull
    private Long id;
    @Column
    @Nonnull
    private String title;
    @Column
    @Nonnull
    @DateTimeFormat(pattern = "yyyy-MM-dd")
```

```
    private Date date;
    @Column
    @Nonnull
    private String status;
    public ToDo() {
    }
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public Date getDate() {
        return date;
    }
    public void setDate(Date date) {
        this.date = date;
    }
    public String getStatus() {
        return status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
}
```

IToDoRepo.java

```
package com.example.ToDoApp.repo;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.example.ToDoApp.model.ToDo;
@Repository
public interface IToDoRepo extends JpaRepository<ToDo, Long>{
}
```

ToDoServices.java

```
package com.example.ToDoApp.service;
import java.util.ArrayList;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.example.ToDoApp.model.ToDo;
import com.example.ToDoApp.repo.IToDoRepo;
@Service
public class ToDoService {
	@Autowired
	IToDoRepo repo;
	public List<ToDo> getAllToDoItems() {
		ArrayList<ToDo> todoList = new ArrayList<>();
		repo.findAll().forEach(todo -> todoList.add(todo));

		return todoList;
	}
	public ToDo getToDoItemById(Long id) {
		return repo.findById(id).get();
	}
	public boolean updateStatus(Long id) {
		ToDo todo = getToDoItemById(id);
		todo.setStatus("Completed");
		return saveOrUpdateToDoItem(todo);
```

```
    }
    public boolean saveOrUpdateToDoItem(ToDo todo) {
        ToDo updatedObj = repo.save(todo);
        if (getToDoItemById(updatedObj.getId()) != null) {
            return true;
        }
        return false;
    }
    public boolean deleteToDoItem(Long id) {
        repo.deleteById(id);
        if (repo.findById(id).isEmpty()) {
            return true;
        }
        return false;
    }
}
```

ADVANTAGES:

1. *Modularity:*

The system's modular design separates the core functionalities, including task management, data handling, and user interfaces, into distinct layers. This clear separation allows for easier debugging, maintenance, and enhancements over time. By organizing the system into modules, each part can evolve independently, ensuring that future features and improvements don't disrupt the existing functionalities.

2. *Scalability:*

The architecture of the system is designed with scalability in mind. The use of Spring Boot and JPA makes it easier to add new features, such as advanced task filtering, categorization, and even integration with other systems like task reminder services. The repository and service layers abstract the database interactions, making it simpler to scale the system for higher user loads or additional functionality without compromising performance.

3. *User-Friendly Interface:*

The frontend of the application uses JSP to render views that are clear, concise, and easy to navigate. The Member and Admin dashboards are designed with the user experience in mind, providing simple access to essential functions like adding, editing, and deleting tasks. The clean and minimalistic layout reduces complexity, making it easy for non-technical users to interact with and manage their tasks efficiently.

4. *Maintainability:*

The project structure adheres to best practices by following design patterns like MVC (Model-View-Controller). The use of Spring Boot ensures that much of the boilerplate code is handled automatically, allowing developers to focus on adding business logic rather than dealing with repetitive configuration. The code is well-organized, and the use of annotations makes it easy to locate and modify specific components. As a result, the system is easy to maintain, and future developers can quickly understand and contribute to the codebase.

5. *Role-Based Access Control:*

By implementing role-based access control, the system ensures that different user types (admin and member) only have access to the appropriate parts of the application. Admins can manage all tasks and user accounts, while members can only manage their own tasks. This control enhances the security and organization of the system, ensuring that users only interact with data that they are authorized to access.

6. *Efficient Task Management:*

The system supports basic but crucial task management features, such as task creation, completion tracking, and deletion. Users can view tasks, update their status, and categorize tasks for better organization. This setup provides a solid foundation for managing personal tasks, and its simplicity makes it highly effective in tracking daily activities. It also allows users to see an overview of their tasks, ensuring better task prioritization and productivity.

DISADVANTAGES

1. *Security Risks:*

One of the significant security concerns is the handling of user credentials. As the system currently stores passwords in plain text, it is vulnerable to data breaches. If sensitive data is compromised, malicious actors could gain unauthorized access to user accounts. To improve security, it's crucial to implement password hashing algorithms such as bcrypt or Argon2 to securely store user passwords. This enhancement would greatly reduce the risk of unauthorized access.

2. *Limited Features:*

While the system covers the basic requirements of task management, it lacks advanced features that would be expected in a more fully developed project. Features such as task categorization, priority settings, or integration with calendar services are not present. Additionally, the absence of overdue task notifications or reminders means that users must manually check and manage their tasks, which can be time-consuming. Implementing these features would increase the system's functionality and make it more competitive in the market.

3. *Concurrency Issues:*

The system doesn't include mechanisms to handle concurrent access to tasks, which could lead to issues if multiple users attempt to update or delete the same task at the same time. Without proper synchronization, there could be inconsistencies in the task data. Implementing locks or database transactions would be necessary to prevent these types of concurrency issues and ensure that tasks are updated or deleted correctly even in multi-user environments.

4. *Database Dependency:*

The system is highly dependent on a live database connection to function properly. If the database server is down or inaccessible, the entire application becomes unusable. To mitigate this, a more robust error-handling mechanism could be implemented, such as retry logic or fallback systems, ensuring that the application remains functional even in the case of temporary database outages.

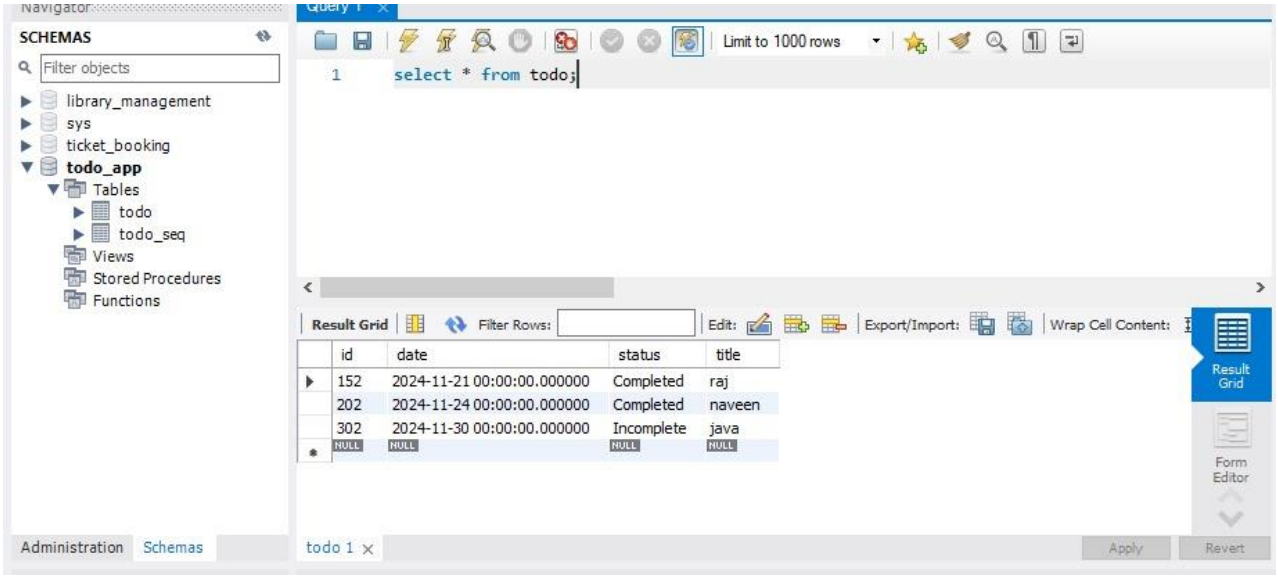
5. *Error Handling:*

The current implementation lacks comprehensive error handling and detailed logging. Users may encounter generic error messages without understanding the root cause of the issue. Without proper error tracking and logging, debugging issues becomes difficult, especially in a production environment. Improving the error handling system by providing detailed messages and logging errors for developers would significantly enhance the user experience and make troubleshooting easier.

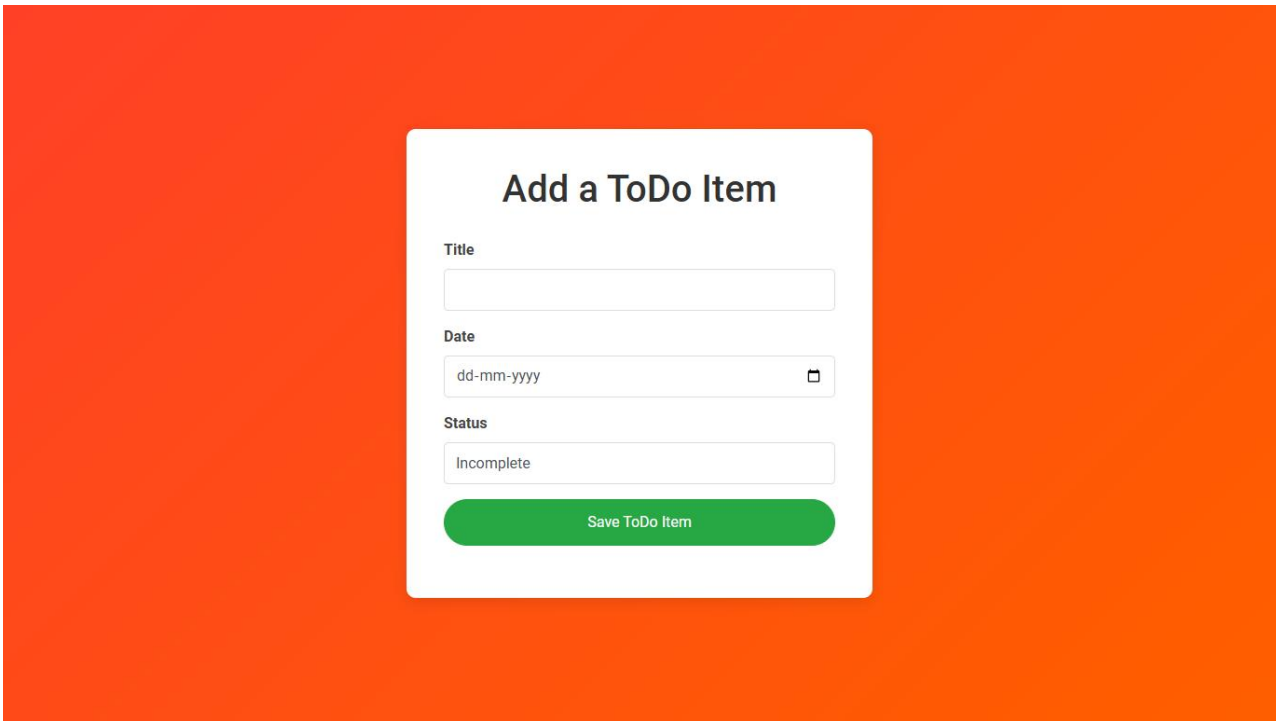
6. *No Data Validation:*

Currently, the system does not perform thorough validation on user inputs, which could lead to invalid or inconsistent data being entered into the system. For example, a user might accidentally enter invalid dates or leave required fields empty. Implementing input validation for user forms and ensuring that only valid data is entered into the system would greatly improve data quality and user experience.

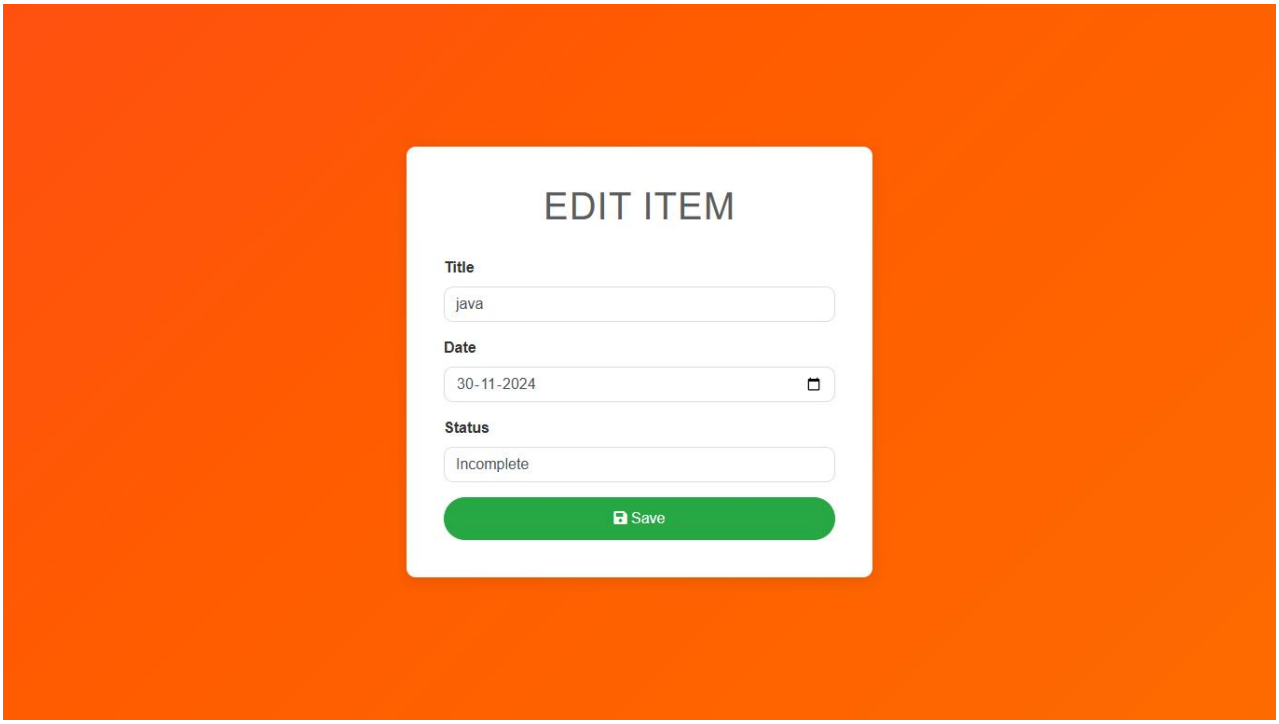
OUTPUT:
SQL Table:



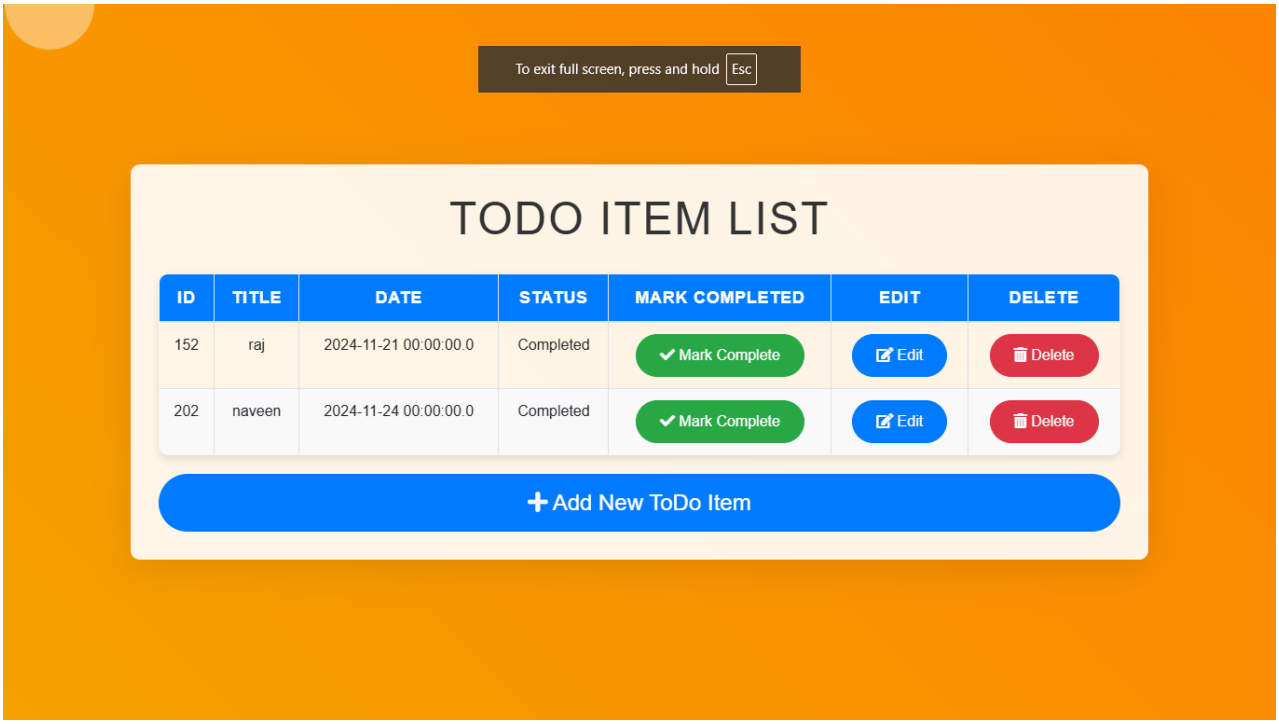
Add In List:



Edit To do item:



To do List:



OVERALL SUMMARY:

The **ToDo List Management System** is a Java-based web application built with **Spring Boot**, **JSP**, and **JPA**. It provides users with a platform to create, view, edit, and delete tasks, offering basic task management functionality. The system follows a **Model-View-Controller (MVC)** architecture, separating task data, views, and logic. Users interact with **JSP** pages for task creation and management, while the **ToDoController** handles routing and operations like adding, updating, and deleting tasks. **ToDoService** provides business logic and interacts with the database through **JPA**. The system uses **DatabaseUtils** for database connectivity. It is designed to be modular, making it easy to maintain and scale. Future enhancements could include features like task prioritization, notifications, and improved security. The system serves as a solid foundation for personal task management.