Status	Finished
Started	Friday, 20 September 2024, 7:41 PM
Completed	Friday, 20 September 2024, 7:59 PM
Duration	18 mins 6 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.util.Scanner;
public class main{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        if(n%2==0)
        System.out.print("1");
        else
        System.out.print("2");
    }
}
```

	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.util.Scanner;
public class main
{
    public static void main(String args[])
    {
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    n=Math.abs(n);
    int result=(n%10);
    System.out.print(result);
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

For example,

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

Below is the explanation:

Last digit of the 267 is 7

Last digit of the 154 is 4

Sum of 7 and 4 = 11

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

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

i.e.

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

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

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

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

For example:

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

Answer: (penalty regime: 0 %)

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

	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	Friday, 20 September 2024, 8:00 PM
Completed	Friday, 20 September 2024, 8:35 PM
Duration	35 mins 25 secs

Question **1**

Correct

Marked out of 5.00

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

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

Example Input:

5

Output:

4

Example Input:

8

Output:

24

Example Input:

11

Output:

149

For example:

Input	Result
5	4
8	24
11	149

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class main{
    public static int result(int n)
    {
        if(n==1)
        return 0;
        else if(n==2||n==3)
        return 1;
        int a=0,b=1,c=1,count=3,term=0;
        for(int i=3;i<n;i++)
        {
            term=a+b+c;
            a=b;
            b=c;
            c=term;
        }
        return term;

    }
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int output=result(n);
        System.out.print(output);
    }
}
```

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

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.util.*;
public class main{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        String s=String.valueOf(n);
        for(int i=0;i<s.length();i++)
        {
            switch(Character.getNumericValue(s.charAt(i)))
            {
                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

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.Scanner;
public class prog {
    // Function to return trailing
    // 0s in factorial of n
    public static int findTrailingZeros(int n)
    {
        if (n < 0) // Negative Number Edge Case
            return -1;

        // Initialize result
        int count=0;

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

```

```
        return count;
    }

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

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

Passed all tests! ✓

◄ Lab-02-MCQ

Jump to...

Lab-03-MCQ ►

Status	Finished
Started	Saturday, 21 September 2024, 3:58 PM
Completed	Saturday, 21 September 2024, 4:53 PM
Duration	54 mins 47 secs

Question **1**

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.util.Scanner;
public class Main
{
    static int maximum(int a[], int n)
    {
        int max=0;
        for(int i=0;i<n;i++)
        {
            if(a[i]>max)
            max=a[i];
        }
        return max;
    }
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        int a[]=new int[n];
        for(int i=0;i<n;i++)
        a[i]=s.nextInt();
        int max=maximum(a,n);
        for(int i=0;i<n;i++)
        {
        a[i]=a[i]-max;
        a[i]=a[i]*max;
        System.out.print(a[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 **2**

Correct

Marked out of 5.00

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

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

In this question's scope, the number 0 should be considered as positive.

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

input1 represents the number of elements in the array.

input2 represents the array of integers.

Example 1:

input1 = 16

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

Expected output = 62

Explanation:

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

Example 2:

input1 = 11

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

Expected output = -1

Explanation:

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

Example 3:

input1 = 16

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

Expected output = 174

Explanation:

The input array contains four sequences of POSITIVE numbers, i.e. "32, 26, 92", "12, 0, 12", "4, 32", and "78". The first and second sequences "32, 26, 92" and "12, 0, 12" are the longest of the four as they contain 4 elements each. Therefore, the expected output = sum of the longest sequence of POSITIVE numbers = (32 + 26 + 92) + (12 + 0 + 12) = 174.

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class Main{
    public static int sequence(int a[],int n)
    {
        int len=0,csum=0,maxlen=0,maxsum=0;
        for(int i=0;i<n;i++)
        {
            if(a[i]>=0)
            {
                len++;
```

```
                csum+=a[i];
            }

            else
            {
                if(len>maxlen)
                {
                    maxsum=csum;
                    maxlen=len;
                }
                else if(len==maxlen)
                {
                    maxsum+=csum;
                }
                csum=0;
                len=0;
            }
        }
        if(len>maxlen)
        {
            maxlen=len;
            maxsum=csum;
        }
        else if(len==maxlen)
        maxsum+=csum;
        if(maxlen<=0)
        return -1;
        else
        return maxsum;
    }
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        int a[]=new int[n];
        for(int i=0;i<n;i++)
        {
            a[i]=s.nextInt();
        }
        System.out.print(sequence(a,n));
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

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

This is explained below:

Example 1:

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

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

Step 1:

Starting from the 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.util.*;
public class Main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
```

```
        int a[]=new int[n];
        for(int i=0;i<n;i++)
        {
            a[i]=s.nextInt();
        }
        int b[]=new int[n],sum=0;
        for(int i=0;i<n;i++)
        {
            int count=0,temp=0;
            while(count!=i+1)
            {
                temp=a[i]%10;
                a[i]/=10;
                count++;
            }
            b[i]=temp*temp;
            sum+=b[i];
        }
        System.out.print(sum);
    }
}
```

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

Passed all tests! ✓

◄ Lab-03-MCQ

Jump to...

Simple Encoded Array ►

Status	Finished
Started	Saturday, 21 September 2024, 8:19 PM
Completed	Saturday, 21 September 2024, 9:19 PM
Duration	1 hour

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=null;
    private int rollno=0;
    public Student()
    {
        System.out.println("No-arg constructor is invoked");
    }
    public Student(String name)
    {
        this.name=name;
        System.out.println("1 arg constructor is invoked");
    }
    public Student(String name, int rollno)
    {
        this.name=name;
        this.rollno=rollno;
        System.out.println("2 arg constructor is invoked");
    }
    public String getName()
    {
        return name;
    }
    public int getrollno()
    {
        return rollno;
    }
    public static void main(String args[])
    {
        Student student1=new Student();
        Student student2=new Student("Rajalakshmi");
        Student student3=new Student("Lakshmi",101);
        System.out.println("Name ="+student1.getName()+" , Roll no = "+student1.getrollno());
        System.out.println("Name ="+student2.getName()+" , Roll no = "+student2.getrollno());
        System.out.println("Name ="+student3.getName()+" , Roll no = "+student3.getrollno());
    }

}

```

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

Area of Circle = πr^2

Circumference = $2\pi r$

Input:

2

Output:

Area = 12.57
Circumference = 12.57

For example:

Test	Input	Result
1	4	Area = 50.27 Circumference = 25.13

Answer: (penalty regime: 0 %)

Reset answer

```
import java.io.*;
import java.util.Scanner;
class Circle
{
    private double radius;
    public Circle(double radius2){

        setRadius(radius2);

    }
    public void setRadius(double radius2){

       radius=radius2;

    }
    public double getRadius(){
       return radius;


    }
    public double calculateArea(){
       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! ✓

Question **3**

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 %)

```
import java.util.Scanner;
public class Mobile{
    private String manufacturer;
    private String operating_system;
    private String color;
    private int cost;
    public Mobile(String m,String s,String c,int price)
    {
        manufacturer=m;
        operating_system=s;
        color=c;
        cost=price;
    }
    public String getmanufacturer()
    {
        return manufacturer;
    }
    public String getopsys()
    {
        return operating_system;
    }
    public String getcolour()
    {
        return color;
    }
    public int getcost()
    {
        return cost;
    }
    public String toString(){
        return "manufacturer ="+getmanufacturer()+
               "\noperating_system ="+getopsys()+
               "\ncolor ="+getcolour()+
               "\ncost = "+getcost();
    }
    public static void main(String args[])
    {
       Mobile mobile=new Mobile(" Redmi"," Andriod"," Blue",34000);
       System.out.print(mobile.toString());
```

```
40      }
41  }
```

	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! ✓

◄ Lab-04-MCQ

Jump to...

Number of Primes in a specified range ►

Status	Finished
Started	Monday, 30 September 2024, 4:05 PM
Completed	Monday, 30 September 2024, 4:40 PM
Duration	34 mins 47 secs

Question **1**

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 basicmethod(){
        System.out.println("Basic Mobile is Manufactured");
    }
}
class CameraMobile extends Mobile{
    public CameraMobile()
    {
        super();
        System.out.println("Camera Mobile is Manufactured");
    }
    public void newFeature()
    {
        System.out.println("Camera Mobile with 5MG px");
    }
}
class AndroidMobile extends CameraMobile{
    public AndroidMobile()
    {
        super();
        System.out.println("Android Mobile is Manufactured");
    }
    public void androidMobile()
    {
        System.out.println("Touch Screen Mobile is Manufactured");
    }
}
public class main{
    public static void main(String args[])
    {
        AndroidMobile a=new AndroidMobile();
```

```
        AndroidMobile a = new AndroidMobile();
        a.newFeature();
        a.androidMobile();
    }
}
```

	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! ✓

Question **2**

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 field to store the account number
    private String accountNumber;

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

    // Constructor to initialize account number and balance
    BankAccount(String s,double n)
    {
        accountNumber=s;
        balance=n;
    }




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

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

    // Method to get the current balance
    public double getBalance() {
        // Return the current balance
        return balance;
    }
}

 class SavingsAccount extends BankAccount {
    // Constructor to initialize account number and balance
    public SavingsAccount(String accountNumber, double balance) {
        // Call the parent class constructor
        super(accountNumber,balance);
    }

    // Override the withdraw method from the parent class
```

```
51      @Override
52      public void withdraw(double amount) {
```

	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 **3**

Correct

Marked out of 5.00

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

College:

String collegeName;

public College() { }

public admitted() { }

Student:

String studentName;

String department;

public Student(String collegeName, String studentName,String depart) { }

public toString()

Expected Output:

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

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

```
class College
{
protected   String collegeName;

public College(String collegeName) {
    // initialize the instance variables
    this.collegeName=collegeName;
    }

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

String studentName;
String department;

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

}

public String toString(){
    // return the details of the student
    return "CollegeName : "+collegeName+"\nStudentName : "+studentName+"\nDepartment : "+department;
}
}
public class Main
{
public static void main (String[] args)
{
```

```
36          Student  s1 = new Student("REC","Venkatesh","CSE");
37          s1.admitted(); // invoke the admitted() method
38          System.out.println(s1.toString());
39  }
40  }
```

	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! ✓

◄ Lab-05-MCQ

Jump to...

Is Palindrome Number? ►

Status	Finished
Started	Tuesday, 1 October 2024, 8:13 AM
Completed	Tuesday, 1 October 2024, 9:27 AM
Duration	1 hour 14 mins

Question **1**

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 main{
    public static String process(String s1,String s2)
    {
        StringBuilder s3= new StringBuilder(s1+s2);
        for(int i=0;i<s3.length()-1;i++)
        {
            for(int j=i+1;j<s3.length();j++)
            {
                if(s3.charAt(i)==s3.charAt(j)||s3.charAt(i)==' ')
                {
                    s3.deleteCharAt(i);
                    i--;
                    break;
                }
            }
        }
        String s=s3.toString();
        char c[]=s.toCharArray();
        Arrays.sort(c);
        for(int i=0,j=c.length-1;j>i;i++,j--)
        {
            char temp=c[i];
            c[i]=c[j];
            c[j]=temp;
        }
```

```
        return String.valueOf(c);

    }
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        String s1=s.nextLine();
        String s2=s.nextLine();
       //System.out.println(s1.length());
       int f=0;
       for(int i=0;i<s1.length();i++){
           if(s1.charAt(i)>='a' && s1.charAt(i)<='z')
           {
               f=1;
               break;
           }
       }
        if(f==0)
        System.out.println("null");
        else
        System.out.print(process(s1,s2));

    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

For example:

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

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

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

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

If the word to be processed is "Nice":

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

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

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

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

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

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

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

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

Expected output:

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

Example 1:

input1 = "Today is a Nice Day"

input2 = 41

output = "iNce doTday"

Example 2:

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

input2 = 39

output = "naMngo arGpes"

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

Note: The input string input1 will NOT contain any other special characters.

Note: The input number input2 will always be a 2-digit number (>=11 and <=99). One of its digits will never be 0. Both the digits of the number will always point to a valid word in the input1 string.

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
public class main{
    public static void process(String s1)
    {
        int mid=s1.length()/2;
        int midbegin=mid;
        //System.out.print(mid);
        if(s1.length()%2!=0)
        {
            for(int i=midbegin;i>=0;i--)
```

```
            {
                System.out.print(s1.charAt(i));
            }
            for(int i=midbegin;i<s1.length();i++)
            {
                System.out.print(s1.charAt(i));
            }
        }
        else{
            for(int i=midbegin-1;i>=0;i--)
            {
                System.out.print(s1.charAt(i));
            }
        for(int i=midbegin;i<s1.length();i++)
        {
            System.out.print(s1.charAt(i));
        }
        }
        System.out.print(" ");
    }
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String s=sc.nextLine();
        int n=sc.nextInt();
        String st[]=s.split(" ");
        String s1=st[n/10-1];
        String s2=st[n%10-1];
        process(s1);
        process(s2);
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Given 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.*;
public class main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        String sent=s.nextLine();
        String word[]=sent.split(":");
        String out=new String();
        for(int i=0;i<word.length;i++)
        {
            char c1=word[i].charAt(0);
            char c2=word[i].charAt(1);
            if(c1==c2)
            out+=Character.toString(c1);
            else
            {
                int j=Math.abs(c1-c2);
                char a=(char)(j+'a'-1);
                out+=Character.toString(a);
            }
        }
        out=out.toUpperCase();
        System.out.print(out);
    }
}
```

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

Passed all tests! ✓

◄ Lab-06-MCQ

Jump to...

Return second word in Uppercase ►

Status	Finished
Started	Monday, 30 September 2024, 7:38 PM
Completed	Monday, 30 September 2024, 8:09 PM
Duration	30 mins 55 secs

Question **1**

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 p1, int p2){
    if(p1>p2)
    System.out.println(homeTeam+" is the winner!");
    else if(p1<p2)
    System.out.println(visitingTeam+" is the winner!");
    else
    System.out.println("It's a tie match.");
}
```

```
}
public class Main{
    public static void main(String[] args){
        String hname;
        Scanner sc= new Scanner(System.in);
       hname=sc.nextLine();
        String  vteam=sc.next();
        int htpoints=sc.nextInt();
        int vtpoints=sc.nextInt();
   College s= new College();
    s.setHomeTeam(hname);
    s.setVisitingTeam(vteam);
    s.homeTeamScored(htpoints);
    s.visitingTeamScored(vtpoints);
    s.winningTeam(htpoints,vtpoints);

}
```

	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 **2**

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";
    default void policyNote(){
        System.out.println("RBI has a new Policy issued in 2023");
    };
    default void regulations(){
        System.out.println("RBI has updated new regulations in 2024.");
    }
    public void methodofinterest();
}
class sbi implements rbi{
    public void methodofinterest()
    {
        System.out.println("SBI rate of interest: 7.6 per annum.");
    }
}
class karur implements rbi{
    public void methodofinterest()
    {
        System.out.println("Karur rate of interest: 7.4 per annum.");
    }
}
public class main{
    public static void main(String args[])
    {
        sbi s=new sbi();
        karur k=new karur();
        s.policyNote();
        s.regulations();
        s.methodofinterest();
        k.methodofinterest();
    }
}

```

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

Passed all tests! ✓

Question **3**

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 play{
    void play();
}
class football implements play{
    String name;
    public football(String s)
    {
        name=s;
    }
    public void play()
    {
        System.out.println(name+" is Playing football");
    }
}
class volleyball implements play{
    String name;
    public volleyball(String s)
    {
        name=s;
    }
    public void play()
    {
        System.out.println(name+" is Playing volleyball");
    }
}
class basketball implements play{
    String name;
    public basketball(String s)
    {
        name=s;
    }
    public void play()
```

```
    {
        System.out.println(name+" is Playing basketball");
    }
}
public class main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        String s1=s.nextLine();
        String s2=s.nextLine();
        String s3=s.nextLine();
        football f=new football(s1);
        f.play();
        volleyball v=new volleyball(s2);
        v.play();
        basketball b=new basketball(s3);
        b.play();


```

	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! ✓

◄ Lab-07-MCQ

Jump to...

Generate series and find Nth element ►

Status	Finished
Started	Sunday, 6 October 2024, 6:27 PM
Completed	Monday, 7 October 2024, 12:17 PM
Duration	17 hours 49 mins

Question **1**

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 main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        String str[]=new String[n];
        for(int i=0;i<n;i++)
        str[i]=s.next();
        int f=0;
        String str2="";
        for(int i=0;i<n;i++)
        {
            str[i]=str[i].toLowerCase();

            char ch1=str[i].charAt(0);
            char ch2=str[i].charAt(str[i].length()-1);
            if((ch1=='a'||ch1=='e'||ch1=='i'||ch1=='o'||ch1=='u')&&(ch2=='a'||ch2=='e'||ch2=='i'||ch2=='o'||ch2=='
            {
```

```
                {
                    f=1;
                    str2+=str[i];
                }
            }
            if(f==0)
            System.out.print("no matches found");
            else
            System.out.print(str2);
        }
}
```

	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! ✓

Question **2**

Correct

Marked out of 5.00

1. Final Variable:

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

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

2. Final Method:

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

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

3. Final Class:

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

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

you should delete any piece of code.

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

```
class FinalExample {

    // Final variable
    final int maxSpeed = 120;

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

class SubClass extends FinalExample {

    public void displayMaxSpeed2() {
        System.out.println("Cannot override a final method");
    }

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

class prog {
    public static void main(String[] args) {
        FinalExample obj = new FinalExample();
        obj.displayMaxSpeed();
        System.out.print("\n");
        SubClass subObj = new SubClass();
        subObj.showDetails();
    }
}
```

```
32 }
33
```

	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! ✓

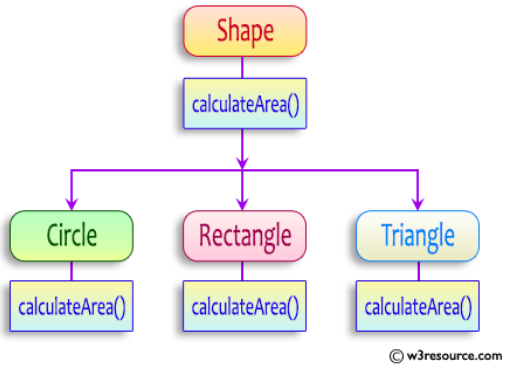
Question **3**

Correct

Marked out of 5.00

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

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




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

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

sample Input :

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

5 // length of the rectangle

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

4 // base of the triangle

3 // height of the triangle

OUTPUT:

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
abstract class shape{
    abstract void calculatearea();
}
class circle extends shape{
    float r;
    public circle(float radius){
        r=radius;
    }
    void calculatearea()
    {
        System.out.printf("Area of a circle: %.2f%n",(Math.PI*r*r));
```

```
    }
}
class rectangle extends shape{
    float l,b;
    public rectangle(float length,float breadth)
    {
        l=length;
        b=breadth;

    }
     void calculatearea()
     {
         System.out.printf("Area of a Rectangle: %.2f%n",(l*b));
     }
}
class triangle extends shape{
    float b,h;
    public triangle(float base,float height)
    {
        b=base;
        h=height;

    }
    void calculatearea()
    {
        System.out.printf("Area of a Triangle: %.2f%n",((0.5)*b*h));
    }
}
public class main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        float radius,length,breadth,base,height;
        radius=s.nextFloat();
        length=s.nextFloat();
        breadth=s.nextFloat();
        base=s.nextFloat();
        height=s.nextFloat();
        circle obj1=new circle(radius);
        rectangle obj2=new rectangle(length,breadth);
```

	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! ✓

◄ Lab-08-MCQ

Jump to...

FindStringCode ►

Status	Finished
Started	Sunday, 13 October 2024, 3:01 PM
Completed	Sunday, 13 October 2024, 3:46 PM
Duration	45 mins

Question **1**

Correct

Marked out of 5.00

Write a Java program to handle ArithmeticException and ArrayIndexOutOfBoundsException.

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

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

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

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

Input:

5

10 0 20 30 40

Output:

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

Input:

3
10 20 30

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

For example:

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

Answer: (penalty regime: 0 %)

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

        try
        {
            int n=s.nextInt();
            int a[]=new int[n];

            for(int i=0;i<a.length;i++)
            {
            a[i]=s.nextInt();
            }
            for(int i=0;i<a.length;i++)
            {
             int result=a[i]/a[i+1];
            }
        }
        catch(ArithmeticException e)
        {
            System.out.println(e);
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println(e);
        }
        finally
        {
            System.out.println("I am always executed");
        }
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

In the following program, an array of integer data is to be initialized.
During the initialization, if a user enters a value other than an integer, it will throw an InputMismatchException exception.
On the occurrence of such an exception, your program should print “You entered bad data.”
If there is no such exception it will print the total sum of the array.

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

Sample Input:

3
5 2 1

Sample Output:

8

Sample Input:

2

1 g

Sample Output:

```
You entered bad data.
```

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

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

    /* Define try-catch block to save user input in the array "name"
   If there is an exception then catch the exception otherwise print
   the total sum of the array. */
     try
      {
        for(int i=0;i<length;i++)
        {
            name[i]=sc.nextInt();
            sum+=name[i];
        }
        System.out.println(sum);
      }
      catch(InputMismatchException e)
     {

          System.out.println("You entered bad data.");
     }
```

```
34    }
35  }
```

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

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;
    // call the trynumber(n);
    trynumber(n);

  }

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

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

```

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

Passed all tests! ✓

◄ Lab-09-MCQ

Jump to...

The “Nambiar Number” Generator ►

Status	Finished
Started	Sunday, 3 November 2024, 7:23 PM
Completed	Sunday, 3 November 2024, 8:00 PM
Duration	37 mins 2 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.Scanner;
import java.util.Collections;
import java.util.ArrayList;
public class main{
    public static void main(String args[])
    {
        ArrayList<Integer>l=new ArrayList<>();
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        for(int i=0;i<n;i++)
        {
            l.add(s.nextInt());
        }
        System.out.println("ArrayList: "+l+"\nFirst : "+l.get(0)+", Last : "+l.get(l.size()-1));

    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

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

list.set();

list.indexOf());

list.lastIndexOf())

list.contains()

list.size());

list.add();

list.remove();

The above methods are used for the below Java program.

Answer: (penalty regime: 0 %)

Reset answer

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

public class Prog {

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

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

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

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

//Replacing the element at index 1 with 100

list.set(1,100);
 //Getting the index of first occurrence of 100
System.out.println("Index of 100 = "+list.indexOf(100));

//Getting the index of last occurrence of 100
 System.out.println("LastIndex of 100 = "+  list.lastIndexOf(100));
// Check whether 200 is in the list or not
System.out.println(list.contains(200)); //Output : false
 // Print ArrayList size
 System.out.println("Size Of ArrayList = "+list.size());
//Inserting 500 at index 1
  list.add(1,500);                              // code here
 //Removing an element from position 3
  list.remove(2);
  list.set(2,100);// 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]	✓

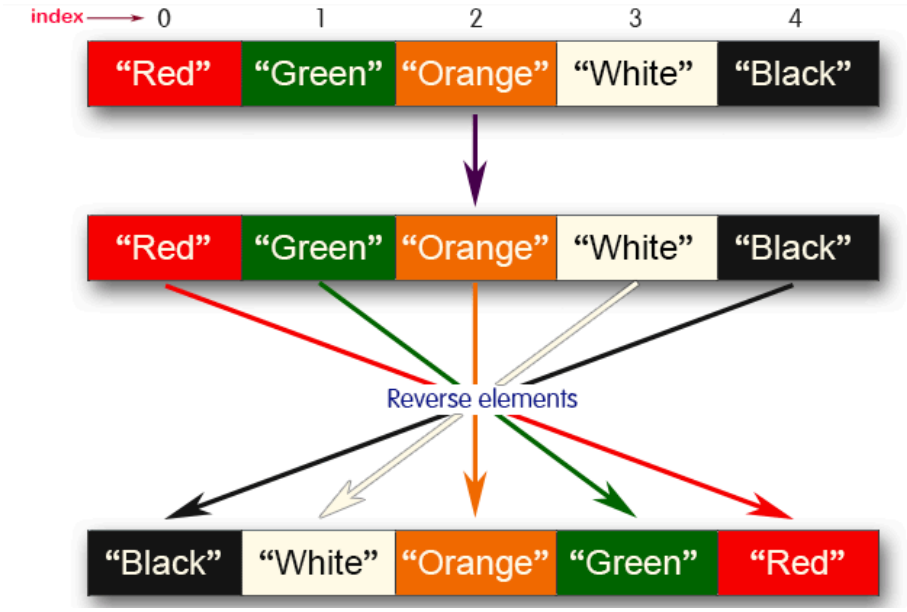
Passed all tests! ✓

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.Scanner;
import java.util.ArrayList;
public class main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        ArrayList<String>l=new ArrayList<>();
        for(int i=0;i<n;i++)
        l.add(s.next());
        ArrayList<String>l2=new ArrayList<>();
        System.out.println("List before reversing :\n"+l);
        for(int i=l.size()-1;i>=0;i--)
        {
            l2.add(l.get(i));
        }
        System.out.println("List after reversing :\n"+l2);
    }
}
```

	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	Saturday, 16 November 2024, 6:13 PM
Completed	Saturday, 16 November 2024, 6:33 PM
Duration	19 mins 59 secs

Question **1**

Correct

Marked out of 1.00

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

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

This class permits the null element.

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

Java HashSet Features

A few important features of HashSet are mentioned below:

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

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.HashSet;
import java.util.Scanner;
 class prog {
  public static void main(String[] args) {
    Scanner sc= new Scanner(System.in);
    int n = sc.nextInt();
    // Create a HashSet object called numbers
    HashSet<Integer>numbers = new HashSet<Integer>(n);

    // Add values to the set
    for(int i=0;i<n;i++)
    numbers.add(sc.nextInt());

  int skey=sc.nextInt();
  int f=0;
  for(int i: numbers)
  {
      if(i==skey)
      {
        System.out.println( skey + " was found in the set.");
        f=1;
        break;
      }
  }
  if(f==0)
  {
  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.*;
public class main{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Set<String>s1=new HashSet<String>();
        Set<String>s2=new HashSet<String>();
        int n1=sc.nextInt();
        for(int i=0;i<n1;i++)
        s1.add(sc.next());

        int n2=sc.nextInt();
        for(int i=0;i<n2;i++)
        s2.add(sc.next());

        Set<String>s3=new HashSet<String>(s1);
        s3.retainAll(s2);
        for(String st: s3)
        System.out.println(st);
    }
}
```

	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());
```

	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	Monday, 18 November 2024, 6:58 PM
Completed	Monday, 18 November 2024, 7:38 PM
Duration	40 mins 26 secs

Question **1**

Correct

Marked out of 5.00

Given two char arrays input1[] and input2[] containing only lower case alphabets, extracts the alphabets which are present in both arrays (common alphabets).

Get the ASCII values of all the extracted alphabets.

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

Return that single digit as output.

Note:

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

Example 1:

input1: {'a', 'b', 'c'}

input2: {'b', 'c'}

output: 8

Explanation:

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

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

98 + 99 = 197

1 + 9 + 7 = 17

1 + 7 = 8

For example:

Input	Result
a b c b c	8

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class main{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        String s1=s.nextLine();
        char a[]=s1.toCharArray();
        String s2=s.nextLine();
        char b[]=s2.toCharArray();
        HashSet<Character>h1=new HashSet<Character>();
        for(char i:a)
        h1.add(i);
        HashSet<Character>h2=new HashSet<Character>();
        for(char i:b)
        {
            if(h1.contains(i))
            h2.add(i);
        }
        int sum=0;
        for(char i: h2)
        {
            if(i>='a' && i<='z')
            sum+=(int)i;
        }

        while (sum > 9) {
            int sum1 = 0;
            while (sum != 0) {
                sum1 += sum % 10;
                sum /= 10;
```

```
            }
            sum = sum1;
        }
        System.out.println(sum);
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

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

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

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

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

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

NOTE:

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

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

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

Examples:

S. No.	input1	input2	output
1	Wipro Technologies Bangalore	0	orpiW seigolonhceT erolagnaB
2	Wipro Technologies, Bangalore	0	orpiW ,seigolonhceT erolagnaB
3	Wipro Technologies Bangalore	1	Orpiw Seigolonhcet Erolagnab
4	Wipro Technologies, Bangalore	1	Orpiw ,seigolonhceT Erolagnab

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;
import java.util.Arrays;

public class ReverseWords {
    public static String reverseWords(String sentence, int caseOption) {
        String[] words = sentence.split("\\s+");
        StringBuilder reversedSentence = new StringBuilder();

        for (String word : words) {
            char[] chars = word.toCharArray();
            char[] a=word.toCharArray();
            int start = 0, end = chars.length - 1;


            while (start < end) {
                char temp = chars[start];
                chars[start] = chars[end];
```

```
                chars[end] = temp;
                start++;
                end--;
            }
            int index=chars.length;
            if (caseOption == 1) {
                for (int i = 0; i < chars.length; i++) {
                    if (Character.isLetter(chars[i])&&i!=index) {
                        chars[i] = Character.isUpperCase(a[i]) ? Character.toUpperCase(chars[i]) : Character.toLow
                    }
                    else  if(!Character.isLetter(chars[i])&&!Character.isWhitespace(chars[i])){
                        char c=a[i];
                        String d=String.valueOf(chars);
                        index=d.indexOf(c);
                        if(index!=-1)
                        chars[index]=a[i];

                      }


                }
            }

            reversedSentence.append(String.valueOf(chars)).append(" ");
        }
        return reversedSentence.toString().trim();
    }

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

	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 **3**

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.Scanner;
public class Main{

    public static void main(String[] args) {
        Scanner s=new Scanner(System.in);
        String st=s.nextLine();
        String result= decode(st);
        System.out.println(result);
    }

    public static String decode(String s) {
        String[] s1 = s.split("1");

        StringBuilder sb= new StringBuilder();

        for (String i: s1) {
            if (i.length() > 0) {
                int n = 26 - i.length();
                sb.append((char) ('A' + n));
            }
        }

        return sb.toString();
    }
}
```

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

Passed all tests! ✓

◄ Lab-12-MCQ

Jump to...

Identify possible words ►

ALPHA STATIONARY SHOP INVENTORY MANAGEMNT SYSTEM

A MINI PROJECT

SUBMITTED BY

LOGAPRIYA S G - 230701164

MADHUMITHA B M - 230701168

In partial fulfilment for the award of the Degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI- 602105

2024-25

BONAFIDE CERTIFIACTE

Certified that this project report "**Alpha Stationary Shop INVENTORY MANAGEMENT SYSTEM"** is a Bonafide work of "**B M MADHUMITHA (230701168) & S G LOGAPRIYA (230701164)"** who carried out the project work under the Supervision of **Mrs.B.S.Dharshini- Assistant Professor (SG)**

Submitted for the Practical Examination held on _____

Signature of

Mrs.B.S.Dharshini

Assistant Professor (SG)

Computer Science and Engineering

Rajalakshmi Engineering College

Thandalam, Chennai-602 105

ABSTRACT

The Alpha Stationary Shop Inventory Management System is a robust solution aimed at optimizing and automating the shop's inventory management processes through an advanced and user-friendly interface. Developed using JavaFX for a modern, intuitive graphical user experience and supported by PL/SQL for efficient database operations, the system effectively addresses the challenges of traditional manual inventory handling.

With JavaFX as the primary frontend technology, the application delivers a responsive, interactive, and visually appealing user interface. It provides a seamless experience for both administrators and customers, allowing for intuitive navigation and efficient task execution. Key components of the interface include real-time data visualization for inventory tracking, dynamic tables for product listings, and interactive forms for user input and management actions.

The backend, powered by PL/SQL, ensures secure and efficient data handling, supporting core functionalities such as inventory tracking, supplier management, customer data storage, and automated invoice generation. The integration between JavaFX and the PL/SQL database facilitates real-time updates, ensuring accurate monitoring of stock levels and enabling timely replenishment, which reduces the risks of overstocking and stockouts.

This scalable and modular system design ensures adaptability to future business needs and technological advancements, making it a reliable and future-proof investment for the shop. Overall, the Alpha Stationary Shop Inventory Management System represents a significant shift from manual processes to an automated, data-driven, and user-focused approach, leveraging the strengths of JavaFX for frontend development and PL/SQL for robust backend management.

TABLE OF CONTENTS

1. **INTRODUCTION**
   1.1 INTRODUCTION
   1.2 IMPLEMENTATION
   1.3 WEBSITE FEATURE

2. **SYSTEM SPECIFICATION**
   2.1 HARDWARE SPECIFICATION
   2.2 SOFTWARE SPECIFICATION

3. **SAMPLE CODE**
   3.1 HOME PAGE
   3.2 ADD ITEM PAGE
   3.3 ADD CUSTOMER PAGE
   3.4 CREATE INVOICE PAGE
   3.5 ADD SUPPLIER PAGE
   3.6 DISPLAY ITEMS PAGE
   3.7 DISPLAY CUSTOMER PAGE
   3.8 DISPLAY INVOICE PAGE
   3.9 DISPLAY SUPPLIERPAGE

4. **SNAPSHOTS**
   4.1 HOME PAGE
   4.2 ADD ITEM PAGE
   4.3 ADD CUSTOMER PAGE
   4.4 CREATE INVOICE PAGE
   4.5 ADD SUPPLIER PAGE
   4.6 DISPLAY ITEMS PAGE
   4.7 DISPLAY CUSTOMER PAGE
   4.8 DISPLAY INVOICE PAGE
   4.9 DISPLAY SUPPLIER PAGE

5. **CONCLUSION**

6. **REFERENCES**

INTRODUCTION

INTRODUCTION:

The project provides a comprehensive platform for handling various aspects of shop management, including inventory tracking, customer and supplier management, sales processing, and invoicing. The user-friendly interface, developed using JavaFX, delivers a responsive and visually appealing experience, simplifying complex tasks for both shop administrators and customers. By integrating PL/SQL, the system ensures efficient data handling, accurate record-keeping, and robust transaction processing.

IMPLEMENTTATION:

This project uses JAVA FX for building the frontend phase and PL SQL to handle the backend and databases.

WEBSITE FEATURE:

A responsive website is created using features of JAVA FX with an active home page.

The home page offers options like Add items, Add customers, Add suppliers, Add invoices and also the options to display the same details.

A responsive corresponding pages are linked with each options.

SYSTEM SPECIFICATION

1. Hardware Requirements

- Processor: Intel Core i5 (10th Gen or higher) / AMD Ryzen 5 or equivalent
- RAM: Minimum 8 GB (16 GB recommended for smoother development)
- Storage: 256 GB SSD (Solid State Drive) or higher.

2. Software Requirements

- Frontend: Java FX
- Backend: Pl Sql
- Operating System: Windows 10

SAMPLE CODE

Home Page:

```
package project;


import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.TableCell;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.image.Image;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.stage.Stage;


import java.sql.Connection;
import java.sql.Date;
import java.sql.ResultSet;
import java.sql.Statement;
```

```
public class App extends Application {


    @Override
    public void start(Stage primaryStage) {
        // Creating buttons for the actions
        Button addItemButton = new Button("Add Item");
        Button addSupplierButton = new Button("Add Supplier");
        Button addCustomerButton = new Button("Add Customer");
        Button addInvoiceButton = new Button("Create Invoice");
        Button addSupplyInvoiceButton = new Button("Add Supply Invoice");
        Button addItemsToInvoiceButton = new Button("Add Items to Invoice");
        Button viewSuppliersButton = new Button("View Suppliers");
        Button viewItemsButton = new Button("View Items");
        Button viewCustomersButton = new Button("View Customers");
        Button viewInvoicesButton = new Button("View Invoices");


        // Style buttons to make the text fully visible
        String buttonStyle = "-fx-font-size: 14px; -fx-padding: 10px 20px; -fx-
background-color: #555555; -fx-text-fill: white; -fx-border-radius: 5px; -fx-
background-radius: 5px; -fx-pref-width: 200px;";
        addItemButton.setStyle(buttonStyle);
        addSupplierButton.setStyle(buttonStyle);
        addCustomerButton.setStyle(buttonStyle);
        addInvoiceButton.setStyle(buttonStyle);
        addSupplyInvoiceButton.setStyle(buttonStyle);
        addItemsToInvoiceButton.setStyle(buttonStyle);
        viewSuppliersButton.setStyle(buttonStyle);
```

```
        viewItemsButton.setStyle(buttonStyle);
        viewCustomersButton.setStyle(buttonStyle);
        viewInvoicesButton.setStyle(buttonStyle);

        // Set button actions
        addItemButton.setOnAction(e -> showAddItemPage());
        addSupplierButton.setOnAction(e -> showAddSupplierPage());
        addCustomerButton.setOnAction(e -> showAddCustomerPage());
        addInvoiceButton.setOnAction(e -> showAddInvoicePage());
        addSupplyInvoiceButton.setOnAction(e -> showAddSupplyInvoicePage());
        addItemsToInvoiceButton.setOnAction(e ->
showAddItemsToInvoicePage());
        viewSuppliersButton.setOnAction(e -> displaySuppliers(primaryStage));
        viewItemsButton.setOnAction(e -> displayItems(primaryStage));
        viewCustomersButton.setOnAction(e -> displayCustomers(primaryStage));
        viewInvoicesButton.setOnAction(e -> displayInvoices(primaryStage));

        // Create VBox layout for buttons
        VBox leftColumn = new VBox(20);  // 20px spacing between buttons in the
left column
        leftColumn.setAlignment(Pos.CENTER_LEFT);

        VBox rightColumn = new VBox(20);  // 20px spacing between buttons in
the right column
        rightColumn.setAlignment(Pos.CENTER_LEFT);
```

```
        VBox centerColumn = new VBox(20);  // 20px spacing between buttons in the center
        centerColumn.setAlignment(Pos.CENTER);

        // Add buttons to the left, right, and center columns
        leftColumn.getChildren().addAll(
            addItemButton, addSupplierButton, addCustomerButton, addInvoiceButton,
            addSupplyInvoiceButton
        );

        rightColumn.getChildren().addAll(
            addItemsToInvoiceButton, viewSuppliersButton, viewItemsButton,
            viewCustomersButton, viewInvoicesButton
        );

        centerColumn.getChildren().addAll(
            // If you want to center specific buttons, you can add them here
        );

        // Create HBox layout to hold left, right, and center columns
        HBox hBox = new HBox(40); // 40px spacing between the columns
        hBox.setAlignment(Pos.CENTER);
        hBox.getChildren().addAll(leftColumn, centerColumn, rightColumn);

        // Create main layout with background image
        VBox mainLayout = new VBox();
```

```
        mainLayout.getChildren().add(hBox);

        // Set the background image
        BackgroundImage background = new BackgroundImage(
            new
Image(getClass().getResource("/project/Background/login.png").toExternalFor
m(), 800, 600, false, true),
            BackgroundRepeat.NO_REPEAT, BackgroundRepeat.NO_REPEAT,
            BackgroundPosition.CENTER, BackgroundSize.DEFAULT
        );
        mainLayout.setBackground(new Background(background));

        // Set up the scene and stage
        Scene scene = new Scene(mainLayout, 800, 600);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Inventory Management System");
        primaryStage.show();
    }

    // Navigation methods for different pages
    private void showAddItemPage() {
        AddItemPage itemPage = new AddItemPage();
        Stage itemStage = new Stage();
        itemPage.start(itemStage);
    }

    private void showAddSupplierPage() {
```

```
        AddSupplierPage supplierPage = new AddSupplierPage();
        Stage supplierStage = new Stage();
        supplierPage.start(supplierStage);
    }

    private void showAddCustomerPage() {
        AddCustomerPage customerPage = new AddCustomerPage();
        Stage customerStage = new Stage();
        customerPage.start(customerStage);
    }

    private void showAddInvoicePage() {
        AddInvoicePage invoicePage = new AddInvoicePage();
        Stage invoiceStage = new Stage();
        invoicePage.start(invoiceStage);
    }

    private void showAddSupplyInvoicePage() {
        AddSupplyInvoicePage supplyInvoicePage = new AddSupplyInvoicePage();
        Stage supplyInvoiceStage = new Stage();
        supplyInvoicePage.start(supplyInvoiceStage);
    }

    private void showAddItemsToInvoicePage() {
        AddItemsToInvoicePage itemsToInvoicePage = new
AddItemsToInvoicePage();
        Stage itemsToInvoiceStage = new Stage();
```

```
        itemsToInvoicePage.start(itemsToInvoiceStage);
    }

    // Go back to the home page
    private void goBackToHomePage(Stage primaryStage) {
        start(primaryStage); // This will navigate back to the home page by calling the start method
    }

    // Methods to display items, suppliers, customers, and invoices
    @SuppressWarnings("unchecked")
    public void displayItems(Stage primaryStage) {
        String query = "SELECT * FROM items";
        ObservableList<Item> items = FXCollections.observableArrayList();
        dbconnect d = new dbconnect();

        try (Connection conn = d.connect();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            while (rs.next()) {
                items.add(new Item(
                        rs.getInt("item_id"),
                        rs.getString("item_name"),
                        rs.getString("description"),
                        rs.getInt("quantity"),
                        rs.getInt("reorder_level"),
```

```
                rs.getBigDecimal("unit_price"),
                rs.getBigDecimal("total_amount"),
                rs.getDate("purchase_date")
            ));
        }
    } catch (Exception e) {
        e.printStackTrace();
    }

    // Create a TableView
    TableView<Item> tableView = new TableView<>(items);
    TableColumn<Item, Integer> idColumn = new TableColumn<>("Item ID");
    TableColumn<Item, String> nameColumn = new TableColumn<>("Item
Name");
    TableColumn<Item, String> descColumn = new
TableColumn<>("Description");
    TableColumn<Item, Integer> quantityColumn = new
TableColumn<>("Quantity");
    TableColumn<Item, Integer> reorderColumn = new
TableColumn<>("Reorder Level");
    TableColumn<Item, String> priceColumn = new TableColumn<>("Unit
Price");

    idColumn.setCellValueFactory(new PropertyValueFactory<>("itemId"));
    nameColumn.setCellValueFactory(new
PropertyValueFactory<>("itemName"));
    descColumn.setCellValueFactory(new
PropertyValueFactory<>("description"));
```

```
        quantityColumn.setCellValueFactory(new
PropertyValueFactory<>("quantity"));
        reorderColumn.setCellValueFactory(new
PropertyValueFactory<>("reorderLevel"));
        priceColumn.setCellValueFactory(new
PropertyValueFactory<>("unitPrice"));

        tableView.getColumns().addAll(idColumn, nameColumn, descColumn,
quantityColumn, reorderColumn, priceColumn);

        // Create "Back" button to return to the home page
        Button backButton = new Button("Back to Home Page");
        backButton.setStyle("-fx-font-size: 14px; -fx-padding: 10px 20px; -fx-
background-color: #555555; -fx-text-fill: white;");
        backButton.setOnAction(e -> goBackToHomePage(primaryStage));

        // Create a VBox to hold the TableView and Back button
        VBox layout = new VBox(20, tableView, backButton);
        Scene scene = new Scene(layout, 800, 600);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Items List");
        primaryStage.show();
    }

    public void displaySuppliers(Stage primaryStage) {
        String query = "SELECT * FROM supplier";
        ObservableList<Supplier> suppliers = FXCollections.observableArrayList();
        dbconnect d = new dbconnect();
```

```
        try (Connection conn = d.connect();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            while (rs.next()) {
                suppliers.add(new Supplier(
                        rs.getInt("supplier_id"),
                        rs.getString("supplier_name"),
                        rs.getString("contact_no"),
                        rs.getString("email"),
                        rs.getString("address")
                ));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        // Create TableView for Suppliers
        TableView<Supplier> tableView = new TableView<>(suppliers);
        TableColumn<Supplier, Integer> idColumn = new
TableColumn<>("Supplier ID");
        TableColumn<Supplier, String> nameColumn = new
TableColumn<>("Supplier Name");
        TableColumn<Supplier, String> contactColumn = new
TableColumn<>("Contact No");
```

```
        TableColumn<Supplier, String> emailColumn = new
TableColumn<>("Email");
        TableColumn<Supplier, String> addressColumn = new
TableColumn<>("Address");


        idColumn.setCellValueFactory(new PropertyValueFactory<>("supplierId"));
        nameColumn.setCellValueFactory(new
PropertyValueFactory<>("supplierName"));
        contactColumn.setCellValueFactory(new
PropertyValueFactory<>("contactNo"));
        emailColumn.setCellValueFactory(new PropertyValueFactory<>("email"));
        addressColumn.setCellValueFactory(new
PropertyValueFactory<>("address"));


        tableView.getColumns().addAll(idColumn, nameColumn, contactColumn,
emailColumn, addressColumn);


        // Create the "Back" button
        Button backButton = new Button("Back to Home Page");
        backButton.setStyle("-fx-font-size: 14px; -fx-padding: 10px 20px; -fx-
background-color: #555555; -fx-text-fill: white;");
        backButton.setOnAction(e -> goBackToHomePage(primaryStage));


        // Create the layout and set the scene
        VBox layout = new VBox(20, tableView, backButton);
        Scene scene = new Scene(layout, 800, 600);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Suppliers List");
```

```
        primaryStage.show();
    }

    public void displayCustomers(Stage primaryStage) {
        String query = "SELECT * FROM customer";  // Modify according to your database schema
        ObservableList<Customer> customers = FXCollections.observableArrayList();
        dbconnect d = new dbconnect();

        try (Connection conn = d.connect();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            while (rs.next()) {
                customers.add(new Customer(
                        rs.getInt("customer_id"),
                        rs.getString("customer_name"),
                        rs.getString("contact_no"),
                        rs.getString("email"),
                        rs.getString("address"), 0,
                        rs.getDate("last_purchase_date")
                ));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        // Create TableView for Customers
        TableView<Customer> tableView = new TableView<>(customers);
        TableColumn<Customer, Integer> idColumn = new
TableColumn<>("Customer ID");
        TableColumn<Customer, String> nameColumn = new
TableColumn<>("Customer Name");
        TableColumn<Customer, String> contactColumn = new
TableColumn<>("Contact No");
        TableColumn<Customer, String> emailColumn = new
TableColumn<>("Email");
        TableColumn<Customer, String> addressColumn = new
TableColumn<>("Address");
        TableColumn<Customer, Date> dateColumn = new TableColumn<>("Last
Purchase Date");

        idColumn.setCellValueFactory(new
PropertyValueFactory<>("customerId"));
        nameColumn.setCellValueFactory(new
PropertyValueFactory<>("customerName"));
        contactColumn.setCellValueFactory(new
PropertyValueFactory<>("contactNo"));
        emailColumn.setCellValueFactory(new PropertyValueFactory<>("email"));
        addressColumn.setCellValueFactory(new
PropertyValueFactory<>("address"));
        dateColumn.setCellValueFactory(new
PropertyValueFactory<>("lastPurchaseDate"));

        tableView.getColumns().addAll(idColumn, nameColumn, contactColumn,
emailColumn, addressColumn,dateColumn);
```

```
        // Create the "Back" button
        Button backButton = new Button("Back to Home Page");
        backButton.setStyle("-fx-font-size: 14px; -fx-padding: 10px 20px; -fx-background-color: #555555; -fx-text-fill: white;");
        backButton.setOnAction(e -> goBackToHomePage(primaryStage));

        // Create the layout and set the scene
        VBox layout = new VBox(20, tableView, backButton);
        Scene scene = new Scene(layout, 800, 600);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Customers List");
        primaryStage.show();
    }

    public void displayInvoices(Stage primaryStage) {
        String query = "SELECT * FROM invoice";  // Modify according to your database schema
        ObservableList<Invoice> invoices = FXCollections.observableArrayList();
        dbconnect d = new dbconnect();

        try (Connection conn = d.connect();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            while (rs.next()) {
                invoices.add(new Invoice(
```

```
                rs.getInt("invoice_id"),
                rs.getInt("customer_id"),
                rs.getBigDecimal("bill_amount"),
                rs.getDate("invoice_date")
            ));
        }
    } catch (Exception e) {
        e.printStackTrace();
    }

    // Create TableView for Invoices
    TableView<Invoice> tableView = new TableView<>(invoices);
    TableColumn<Invoice, Integer> idColumn = new TableColumn<>("Invoice ID");
    TableColumn<Invoice, Integer> customerIdColumn = new TableColumn<>("Customer ID");
    TableColumn<Invoice, String> dateColumn = new TableColumn<>("Invoice Date");
    TableColumn<Invoice, String> totalAmountColumn = new TableColumn<>("Bill Amount");

    idColumn.setCellValueFactory(new PropertyValueFactory<>("invoiceId"));
    customerIdColumn.setCellValueFactory(new PropertyValueFactory<>("customerId"));
    dateColumn.setCellValueFactory(new PropertyValueFactory<>("invoiceDate"));
    totalAmountColumn.setCellValueFactory(new PropertyValueFactory<>("totalAmount"));
```

```
        tableView.getColumns().addAll(idColumn, customerIdColumn,
dateColumn, totalAmountColumn);


        // Create the "Back" button
        Button backButton = new Button("Back to Home Page");
        backButton.setStyle("-fx-font-size: 14px; -fx-padding: 10px 20px; -fx-
background-color: #555555; -fx-text-fill: white;");
        backButton.setOnAction(e -> goBackToHomePage(primaryStage));


        // Create the layout and set the scene
        VBox layout = new VBox(20, tableView, backButton);
        Scene scene = new Scene(layout, 800, 600);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Invoices List");
        primaryStage.show();
    }
 public static void main(String[] args) {
        launch(args);
    }
}
```

Add Items Page:

```
package project;


import javafx.application.Application;
import javafx.geometry.Insets;
```

```
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;

import java.sql.Connection;
import java.sql.Date;
import java.time.LocalDate;

public class AddItemPage extends Application {

    @SuppressWarnings("unused")
    @Override
    public void start(Stage primaryStage) {
        // Labels and TextFields
        Label itemNameLabel = new Label("Item Name:");
        TextField itemNameField = new TextField();

        Label descriptionLabel = new Label("Description:");
        TextField descriptionField = new TextField();

        Label quantityLabel = new Label("Quantity:");
        TextField quantityField = new TextField();

        Label reorderLevelLabel = new Label("Reorder Level:");
        TextField reorderLevelField = new TextField();
```

```
        Label unitPriceLabel = new Label("Unit Price:");
        TextField unitPriceField = new TextField();

        Label purchaseDateLabel = new Label("Purchase Date:");
        DatePicker purchaseDatePicker = new DatePicker(LocalDate.now());

        // Submit button
        Button submitButton = new Button("Add Item");

        submitButton.setOnAction(e -> {
            String itemName = itemNameField.getText();
            String description = descriptionField.getText();
            int quantity = Integer.parseInt(quantityField.getText());
            int reorderLevel = Integer.parseInt(reorderLevelField.getText());
            float unitPrice = Float.parseFloat(unitPriceField.getText());
            Date purchaseDate = Date.valueOf(purchaseDatePicker.getValue());

            // Create dbconnect object and insert the item
            dbconnect db = new dbconnect();
            Connection conn = db.connect();
            db.addNewItem(conn, itemName, description, quantity, reorderLevel,
unitPrice, purchaseDate);
            primaryStage.close();
        });

        // Layout setup
```

```
    GridPane grid = new GridPane();
    grid.setPadding(new Insets(10, 10, 10, 10));
    grid.setVgap(8);
    grid.setHgap(10);

    // Add components to grid
    grid.add(itemNameLabel, 0, 0);
    grid.add(itemNameField, 1, 0);
    grid.add(descriptionLabel, 0, 1);
    grid.add(descriptionField, 1, 1);
    grid.add(quantityLabel, 0, 2);
    grid.add(quantityField, 1, 2);
    grid.add(reorderLevelLabel, 0, 3);
    grid.add(reorderLevelField, 1, 3);
    grid.add(unitPriceLabel, 0, 4);
    grid.add(unitPriceField, 1, 4);
    grid.add(purchaseDateLabel, 0, 5);
    grid.add(purchaseDatePicker, 1, 5);
    grid.add(submitButton, 1, 6);

    // Scene setup
    Scene scene = new Scene(grid, 400, 350);
    primaryStage.setScene(scene);
    primaryStage.setTitle("Add Item");
    primaryStage.show();
}
```

```
}
```

Add Customers Page:

```
package project;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;

import java.sql.Connection;
import java.sql.Date;
import java.time.LocalDate;

public class AddCustomerPage extends Application {

    @SuppressWarnings("unused")
    @Override
    public void start(Stage primaryStage) {
        // Labels and TextFields
        Label customerNameLabel = new Label("Customer Name:");
        TextField customerNameField = new TextField();

        Label contactNoLabel = new Label("Contact No:");
        TextField contactNoField = new TextField();
```

```
        Label emailLabel = new Label("Email:");
        TextField emailField = new TextField();

        Label addressLabel = new Label("Address:");
        TextField addressField = new TextField();

        Label loyaltyPointsLabel = new Label("Loyalty Points:");
        TextField loyaltyPointsField = new TextField();

        Label lastPurchaseDateLabel = new Label("Last Purchase Date:");
        DatePicker lastPurchaseDatePicker = new DatePicker(LocalDate.now());

        // Submit button
        Button submitButton = new Button("Add Customer");

        submitButton.setOnAction(e -> {
            String customerName = customerNameField.getText();
            String contactNo = contactNoField.getText();
            String email = emailField.getText();
            String address = addressField.getText();
            int loyaltyPoints = Integer.parseInt(loyaltyPointsField.getText());
            Date lastPurchaseDate =
    Date.valueOf(lastPurchaseDatePicker.getValue());

            // Create dbconnect object and insert the customer
            dbconnect db = new dbconnect();
```

```
            Connection conn = db.connect();
            db.addCustomer(conn, customerName, contactNo, email, address,
loyaltyPoints, lastPurchaseDate);
            primaryStage.close();
        });

        // Layout setup
        GridPane grid = new GridPane();
        grid.setPadding(new Insets(10, 10, 10, 10));
        grid.setVgap(8);
        grid.setHgap(10);

        // Add components to grid
        grid.add(customerNameLabel, 0, 0);
        grid.add(customerNameField, 1, 0);
        grid.add(contactNoLabel, 0, 1);
        grid.add(contactNoField, 1, 1);
        grid.add(emailLabel, 0, 2);
        grid.add(emailField, 1, 2);
        grid.add(addressLabel, 0, 3);
        grid.add(addressField, 1, 3);
        grid.add(loyaltyPointsLabel, 0, 4);
        grid.add(loyaltyPointsField, 1, 4);
        grid.add(lastPurchaseDateLabel, 0, 5);
        grid.add(lastPurchaseDatePicker, 1, 5);
        grid.add(submitButton, 1, 6);
```

```
        // Scene setup
        Scene scene = new Scene(grid, 400, 350);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Add Customer");
        primaryStage.show();
    }
}
```

Add Invoice Page:

```
package project;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;

import java.sql.Connection;
import java.sql.Date;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.time.LocalDate;

public class AddInvoicePage extends Application {

    @Override
```

```
public void start(Stage primaryStage) {
    // Labels and TextFields
    Label invoiceDateLabel = new Label("Invoice Date:");
    DatePicker invoiceDatePicker = new DatePicker(LocalDate.now());

    Label customerIdLabel = new Label("Customer ID:");
    TextField customerIdField = new TextField();

    Label totalAmountLabel = new Label("Total Amount:");
    TextField totalAmountField = new TextField();

    // Submit button
    Button submitButton = new Button("Create Invoice");

    submitButton.setOnAction(e -> {
        // Get form input values
        try {
            int customerId = Integer.parseInt(customerIdField.getText());
            float totalAmount = Float.parseFloat(totalAmountField.getText());
            Date invoiceDate = Date.valueOf(invoiceDatePicker.getValue());

            // Create dbconnect object and insert the invoice
            dbconnect db = new dbconnect();
            Connection conn = db.connect();

            // SQL query to insert the new invoice
```

```
            String sqlInsertInvoice = "INSERT INTO invoice (invoice_date,
customer_id, bill_amount) VALUES (?, ?, ?)";
            PreparedStatement stmtInsertInvoice =
conn.prepareStatement(sqlInsertInvoice);

            // Set parameters for the prepared statement
            stmtInsertInvoice.setDate(1, invoiceDate);  // Set invoice_date
            stmtInsertInvoice.setInt(2, customerId);    // Set customer_id
            stmtInsertInvoice.setFloat(3, totalAmount); // Set bill_amount

            // Execute the query to insert the invoice
            int rowsAffected = stmtInsertInvoice.executeUpdate();
            if (rowsAffected > 0) {
                System.out.println("Invoice created successfully.");

                // Update the last_purchase_date in the customer table
                String sqlUpdateCustomer = "UPDATE customer SET
last_purchase_date = ? WHERE customer_id = ?";
                PreparedStatement stmtUpdateCustomer =
conn.prepareStatement(sqlUpdateCustomer);

                // Set parameters for the prepared statement
                stmtUpdateCustomer.setDate(1, invoiceDate);  // Set the invoice
date as the last purchase date
                stmtUpdateCustomer.setInt(2, customerId);    // Set customer_id

                // Execute the update query
                int updateRowsAffected = stmtUpdateCustomer.executeUpdate();
```

```
                if (updateRowsAffected > 0) {
                    System.out.println("Customer's last purchase date updated
successfully.");
                    primaryStage.close();
                } else {
                    System.out.println("Failed to update the customer's last purchase
date.");
                }

                // Close the window after success
                primaryStage.close();
            } else {
                System.out.println("Failed to create invoice.");
            }

            // Close the connection
            conn.close();
        } catch (SQLException sqlEx) {
            System.out.println("Database error: " + sqlEx.getMessage());
            sqlEx.printStackTrace();
        } catch (Exception ex) {
            System.out.println("Error: " + ex.getMessage());
            ex.printStackTrace();
        }
    });

    // Layout setup
```

```
        GridPane grid = new GridPane();
        grid.setPadding(new Insets(10, 10, 10, 10));
        grid.setVgap(8);
        grid.setHgap(10);

        // Add components to grid
        grid.add(invoiceDateLabel, 0, 0);
        grid.add(invoiceDatePicker, 1, 0);

        grid.add(customerIdLabel, 0, 2);
        grid.add(customerIdField, 1, 2);

        grid.add(totalAmountLabel, 0, 5);
        grid.add(totalAmountField, 1, 5);
        grid.add(submitButton, 1, 6);

        // Scene setup
        Scene scene = new Scene(grid, 400, 350);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Create Invoice");
        primaryStage.show();
    }
}
```

Add Supplier Page:

```
package project;
```

```
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;

import java.sql.Connection;

public class AddSupplierPage extends Application {

    @SuppressWarnings("unused")
    @Override
    public void start(Stage primaryStage) {
        // Labels and TextFields
        Label supplierNameLabel = new Label("Supplier Name:");
        TextField supplierNameField = new TextField();

        Label contactNoLabel = new Label("Contact No:");
        TextField contactNoField = new TextField();

        Label emailLabel = new Label("Email:");
        TextField emailField = new TextField();

        Label addressLabel = new Label("Address:");
```

```
TextField addressField = new TextField();

// Submit button
Button submitButton = new Button("Add Supplier");

submitButton.setOnAction(e -> {
    String supplierName = supplierNameField.getText();
    String contactNo = contactNoField.getText();
    String email = emailField.getText();
    String address = addressField.getText();

    // Create dbconnect object and insert the supplier
    dbconnect db = new dbconnect();
    Connection conn = db.connect();
    db.addSupplier(conn, supplierName, contactNo, email, address);
    primaryStage.close();
});

// Layout setup
GridPane grid = new GridPane();
grid.setPadding(new Insets(10, 10, 10, 10));
grid.setVgap(8);
grid.setHgap(10);

// Add components to grid
grid.add(supplierNameLabel, 0, 0);
```

```
        grid.add(supplierNameField, 1, 0);
        grid.add(contactNoLabel, 0, 1);
        grid.add(contactNoField, 1, 1);
        grid.add(emailLabel, 0, 2);
        grid.add(emailField, 1, 2);
        grid.add(addressLabel, 0, 3);
        grid.add(addressField, 1, 3);
        grid.add(submitButton, 1, 4);

        // Scene setup
        Scene scene = new Scene(grid, 400, 350);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Add Supplier");
        primaryStage.show();
    }
}
```

Display Items Page:

```
package project;

import java.math.BigDecimal;
import java.util.Date;

public class Item {
    private int itemId;
    private String itemName;
    private String description;
```

```
    private int quantity;
    private int reorderLevel;
    private BigDecimal unitPrice;
    private BigDecimal totalAmount;
    private Date purchaseDate;

    // Constructor
    public Item(int itemId, String itemName, String description, int quantity, int
reorderLevel,
            BigDecimal unitPrice, BigDecimal totalAmount, Date purchaseDate) {
        this.itemId = itemId;
        this.itemName = itemName;
        this.description = description;
        this.quantity = quantity;
        this.reorderLevel = reorderLevel;
        this.unitPrice = unitPrice;
        this.totalAmount = totalAmount;
        this.purchaseDate = purchaseDate;
    }

    // Getters and setters
    public int getItemId() {
        return itemId;
    }

    public void setItemId(int itemId) {
        this.itemId = itemId;
```

```
}

public String getItemName() {
   return itemName;
}

public void setItemName(String itemName) {
   this.itemName = itemName;
}

public String getDescription() {
   return description;
}

public void setDescription(String description) {
   this.description = description;
}

public int getQuantity() {
   return quantity;
}

public void setQuantity(int quantity) {
   this.quantity = quantity;
}
```

```
public int getReorderLevel() {
    return reorderLevel;
}

public void setReorderLevel(int reorderLevel) {
    this.reorderLevel = reorderLevel;
}

public BigDecimal getUnitPrice() {
    return unitPrice;
}

public void setUnitPrice(BigDecimal unitPrice) {
    this.unitPrice = unitPrice;
}

public BigDecimal getTotalAmount() {
    return totalAmount;
}

public void setTotalAmount(BigDecimal totalAmount) {
    this.totalAmount = totalAmount;
}

public Date getPurchaseDate() {
    return purchaseDate;
```

```
    }

    public void setPurchaseDate(Date purchaseDate) {
        this.purchaseDate = purchaseDate;
    }
}
```

Display Customers Page:

```
package project;

import java.util.Date;

public class Customer {
    private int customerId;
    private String customerName;
    private String contactNo;
    private String email;
    private String address;
    private int loyaltyPoints;
    private Date lastPurchaseDate;

    // Constructor
    public Customer(int customerId, String customerName, String contactNo,
String email, String address,
                int loyaltyPoints, Date lastPurchaseDate) {
        this.customerId = customerId;
        this.customerName = customerName;
```

```
        this.contactNo = contactNo;
        this.email = email;
        this.address = address;
        this.loyaltyPoints = loyaltyPoints;
        this.lastPurchaseDate = lastPurchaseDate;
    }

    // Getters and setters
    public int getCustomerId() {
        return customerId;
    }

    public void setCustomerId(int customerId) {
        this.customerId = customerId;
    }

    public String getCustomerName() {
        return customerName;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public String getContactNo() {
        return contactNo;
```

```
}

public void setContactNo(String contactNo) {
   this.contactNo = contactNo;
}

public String getEmail() {
   return email;
}

public void setEmail(String email) {
   this.email = email;
}

public String getAddress() {
   return address;
}

public void setAddress(String address) {
   this.address = address;
}

public int getLoyaltyPoints() {
   return loyaltyPoints;
}
```

```
    public void setLoyaltyPoints(int loyaltyPoints) {
        this.loyaltyPoints = loyaltyPoints;
    }

    public Date getLastPurchaseDate() {
        return lastPurchaseDate;
    }

    public void setLastPurchaseDate(Date lastPurchaseDate) {
        this.lastPurchaseDate = lastPurchaseDate;
    }
}
```

Display Invoice Page:

```
package project;

import java.math.BigDecimal;
import java.util.Date;

public class Invoice {
    private int invoiceId;
    private int customerId;
    private BigDecimal totalAmount;
    private Date invoiceDate;

    // Constructor
```

```
    public Invoice(int invoiceId, int customerId, BigDecimal totalAmount, Date
invoiceDate) {
        this.invoiceId = invoiceId;
        this.customerId = customerId;
        this.totalAmount = totalAmount;
        this.invoiceDate = invoiceDate;
    }

    // Getters and setters
    public int getInvoiceId() {
        return invoiceId;
    }

    public void setInvoiceId(int invoiceId) {
        this.invoiceId = invoiceId;
    }

    public int getCustomerId() {
        return customerId;
    }

    public void setCustomerId(int customerId) {
        this.customerId = customerId;
    }

    public BigDecimal getTotalAmount() {
        return totalAmount;
```

```
    }

    public void setTotalAmount(BigDecimal totalAmount) {
        this.totalAmount = totalAmount;
    }

    public Date getInvoiceDate() {
        return invoiceDate;
    }

    public void setInvoiceDate(Date invoiceDate) {
        this.invoiceDate = invoiceDate;
    }
}
```

Display Supplier Page:

```
package project;

public class Supplier {
    private int supplierId;
    private String supplierName;
    private String contactNo;
    private String email;
    private String address;

    // Constructor
```

```
    public Supplier(int supplierId, String supplierName, String contactNo, String email, String address) {
        this.supplierId = supplierId;
        this.supplierName = supplierName;
        this.contactNo = contactNo;
        this.email = email;
        this.address = address;
    }

    // Getters and setters
    public int getSupplierId() {
        return supplierId;
    }

    public void setSupplierId(int supplierId) {
        this.supplierId = supplierId;
    }

    public String getSupplierName() {
        return supplierName;
    }

    public void setSupplierName(String supplierName) {
        this.supplierName = supplierName;
    }

    public String getContactNo() {
```

```
        return contactNo;
    }

    public void setContactNo(String contactNo) {
        this.contactNo = contactNo;
    }

    public String getEmail() {
        return email;
    }

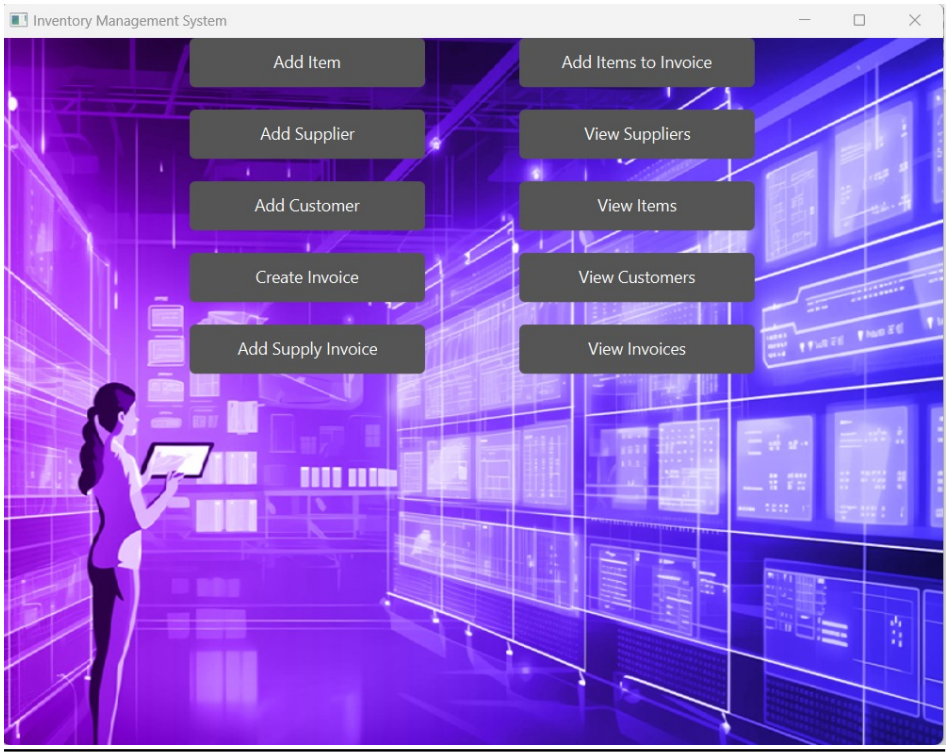
    public void setEmail(String email) {
        this.email = email;
    }

    public String getAddress() {
        return address;
    }

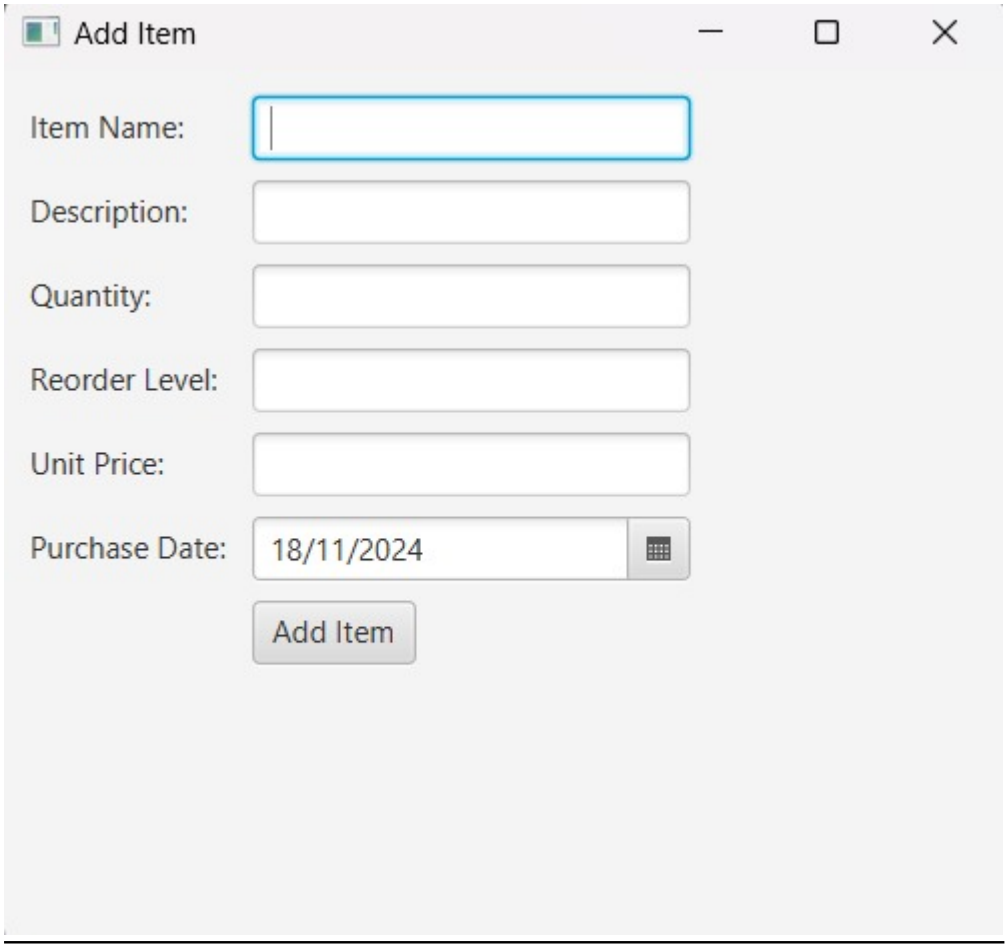
    public void setAddress(String address) {
        this.address = address;
    }
}
```

SNAPSHOTS

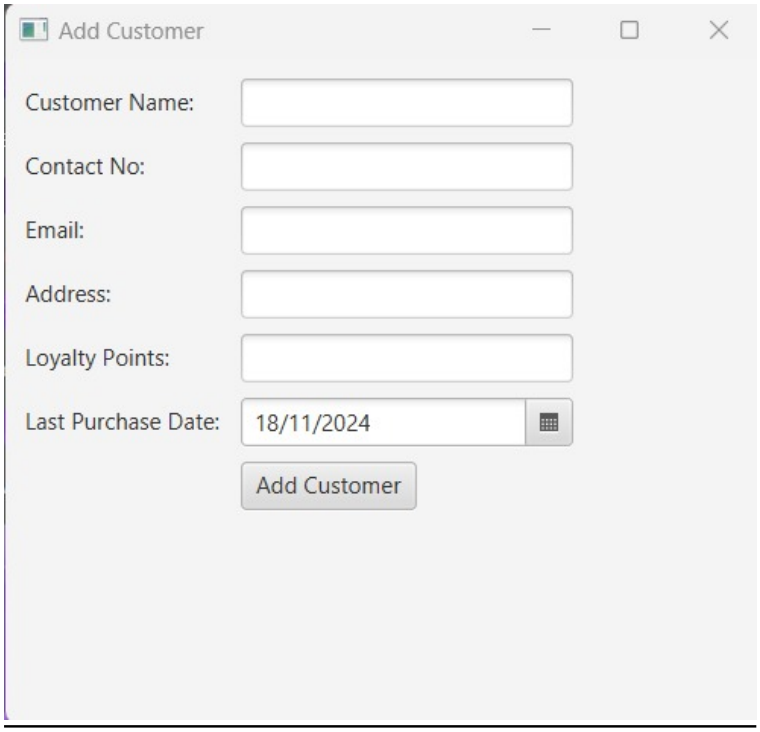
Home Page:



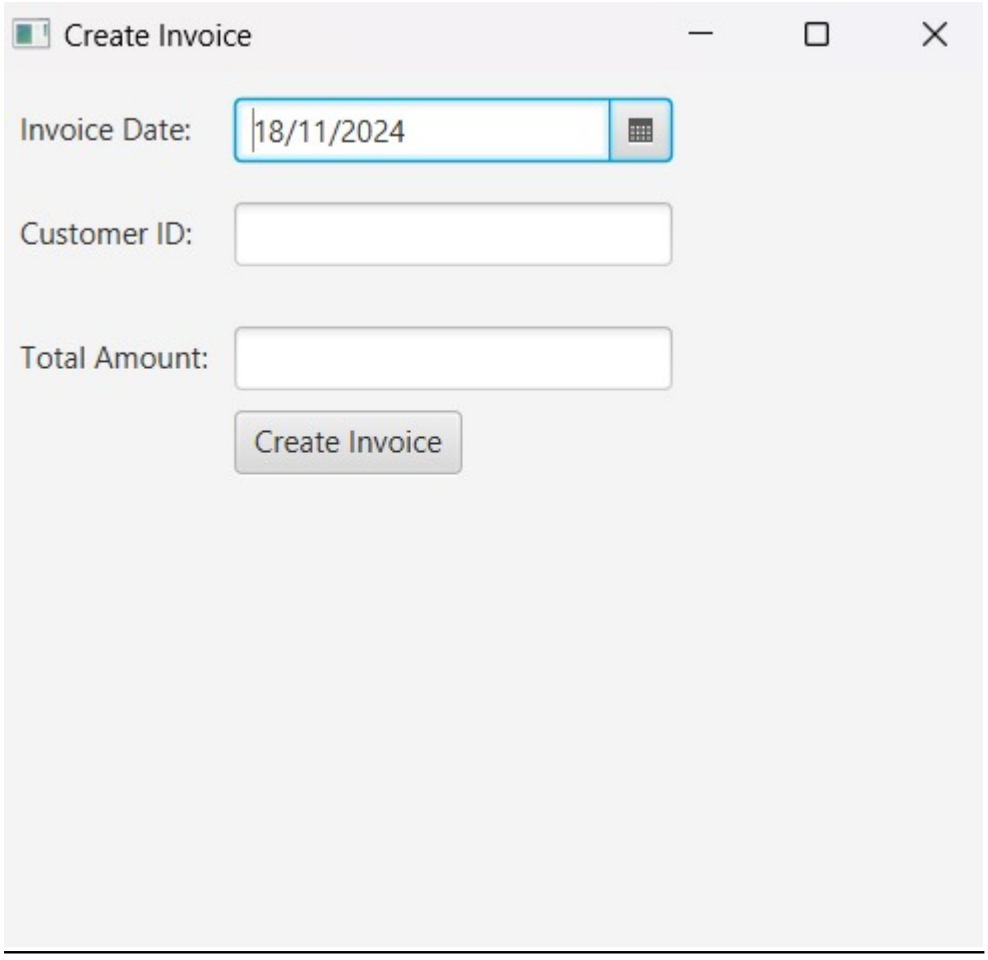
Add Items Page:



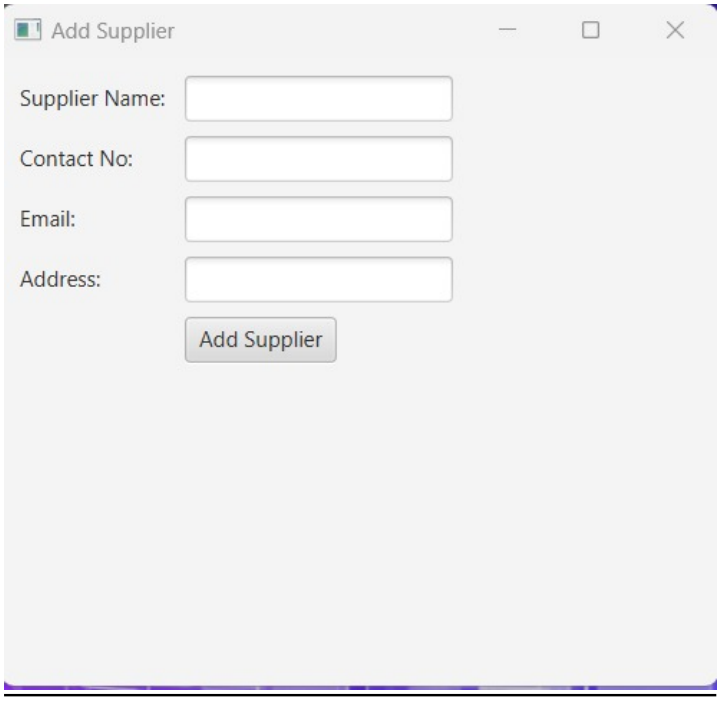
Add Customer Page:



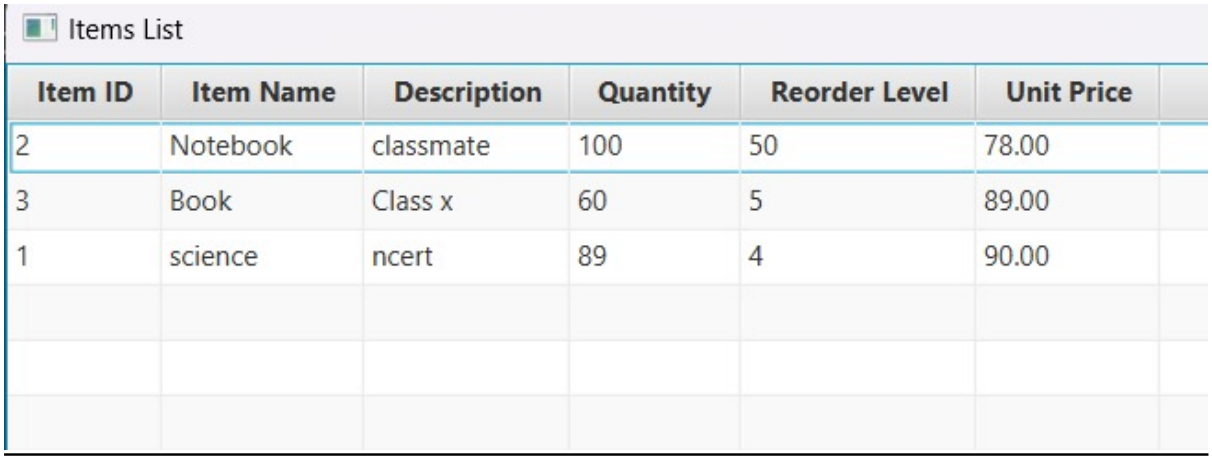
Add Invoice Page:



Add Supplier Page:

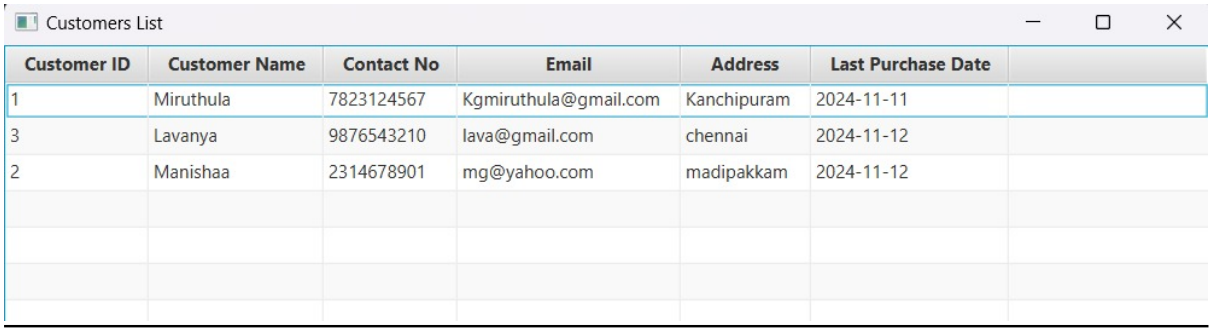


Display Items Page:



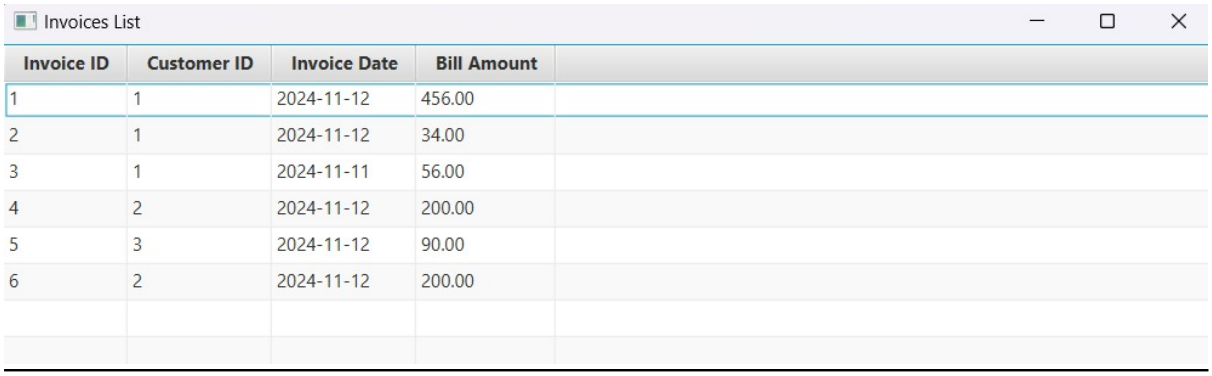
Item ID	Item Name	Description	Quantity	Reorder Level	Unit Price
2	Notebook	classmate	100	50	78.00
3	Book	Class x	60	5	89.00
1	science	ncert	89	4	90.00

Display Customer Page:



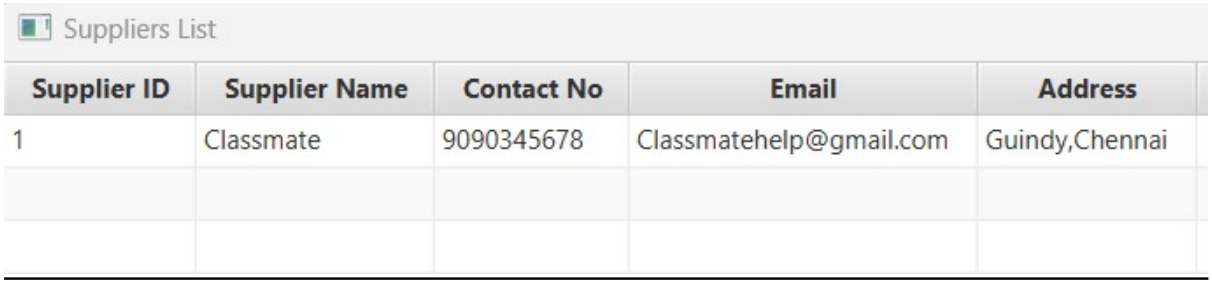
Customer ID	Customer Name	Contact No	Email	Address	Last Purchase Date
1	Miruthula	7823124567	Kgmiruthula@gmail.com	Kanchipuram	2024-11-11
3	Lavanya	9876543210	lava@gmail.com	chennai	2024-11-12
2	Manishaa	2314678901	mg@yahoo.com	madipakkam	2024-11-12

Display Invoice Page:



Invoice ID	Customer ID	Invoice Date	Bill Amount
1	1	2024-11-12	456.00
2	1	2024-11-12	34.00
3	1	2024-11-11	56.00
4	2	2024-11-12	200.00
5	3	2024-11-12	90.00
6	2	2024-11-12	200.00

Display Supplier Page:



Supplier ID	Supplier Name	Contact No	Email	Address
1	Classmate	9090345678	Classmatehelp@gmail.com	Guindy,Chennai

CONCLUSION

The Alpha Stationary Shop Inventory Management System successfully addresses the challenges of manual inventory handling and retail operations by providing an efficient, automated solution tailored to the specific needs of a stationery shop. Through the use of JavaFX for an interactive, user-friendly graphical interface and PL/SQL for secure and robust database management, the project demonstrates the practical integration of frontend and backend technologies.