Status	Finished
Started	Sunday, 22 September 2024, 3:45 PM
Completed	Sunday, 22 September 2024, 3:58 PM
Duration	12 mins 37 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.*;
public class Main{
     public static void main(String args[])
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        int n1=Math.abs(n);
        if(n1==0||n1%2==0)
        {
            System.out.println("1");
        }
        else
        {
           System.out.println("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.*;
public class last{
    public static void main(String args[])
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        System.out.print(Math.abs(n%10));
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

For example,

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

Below is the explanation:

Last digit of the 267 is 7

Last digit of the 154 is 4

Sum of 7 and 4 = 11

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

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

i.e.

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

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

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

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

For example:

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

Answer: (penalty regime: 0 %)

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

	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	Sunday, 22 September 2024, 3:11 PM
Completed	Sunday, 22 September 2024, 3:45 PM
Duration	34 mins 4 secs

Question **1**

Correct

Marked out of 5.00

Consider the following sequence:

1st term: 1

2nd term: 1 2 1

3rd term: 1 2 1 3 1 2 1

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

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

Example Input:

1

Output:

1

Example Input:

4

Output:

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class sequence{
    public static void main(String args[])
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        System.out.println(gennth(n));
    }
    public static String  gennth(int n)
    {
        if(n==1)
        {
            return "1";
        }
        String prev=gennth(n-1);
        StringBuilder nthterm=new StringBuilder();
        nthterm.append(prev).append(" ").append(n).append(" ").append(prev);
        return nthterm.toString();

    }
}
```

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

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

Passed all tests! ✓

Question **2**

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.*;
public class fib{
    public static void main(String args[])
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        int zerbased=n-1;
        System.out.println(findnth(zerbased));
    }
    public static int findnth(int n)
    {
        if(n==0)
        {
            return 0;
        }
        else if(n==1)
        {
            return 1;
        }
        else if(n==2)
        {
            return 1;
        }
        int a=0,b=1,c=1;
        for(int i=3;i<=n;i++)
        {
            int nt=a+b+c;
            a=b;
            b=c;
            c=nt;
        }
        return c;
    }
}
```

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

Passed all tests! ✓

Question **3**

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 now{
    public static void main(String args[])
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        String result=ctw(n);
        System.out.println(result);
    }
    private static String ctw(int n)
    {
        if(n<0||n>99)
        {
            return "numberv oy=ut of range";
        }
        String[] words={"Zero","One","Two","Three","Four","Five","Six","Seven","Eight","Nine"};
        if(n<10)
        {
            return words[n];
        }
        int tend=n/10;
        int unitd=n%10;
        String result=words[tend];
        if(unitd>0)
        {
            result+=" "+words[unitd];
        }
       return result;
}
}
```

	Test	Input	Expected	Got	
✓	1	45	Four Five	Four Five	✓
✓	2	13	One Three	One Three	✓

	Test	Input	Expected	Got	
✓	3	87	Eight Seven	Eight Seven	✓

Passed all tests! ✓

◄ Lab-02-MCQ

Jump to...

Lab-03-MCQ ►

Status	Finished
Started	Sunday, 22 September 2024, 1:54 PM
Completed	Sunday, 22 September 2024, 3:09 PM
Duration	1 hour 15 mins

Question **1**

Incorrect

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.*;
public class LongposSeqsum{
    public static int findsum(int[] a)
    {
        int maxsum=0;
        int cursum=0;
        int longlength=0;
        int curlength=0;
        for(int num:a)
        {
            if(num>0)
            {
                cursum+=num;
                curlength++;
            }
            else if(num==0)
```

```
            {
                curlength++;
            }
            else
            {
                if(curlength>longlength)
                {
                    longlength=curlength;
                    maxsum=cursum;
                }
                else if(curlength==longlength)
                {
                    maxsum=Math.max(maxsum,cursum);
                }
                cursum=0;
                curlength=0;
            }
        }
        if(curlength>longlength)
        {
            maxsum=cursum;
        }
        else if(curlength==longlength)
        {
            maxsum=Math.max(maxsum,cursum);
        }
        return maxsum>0?maxsum:-1;
    }
public static void main(String[] args)
    {
        Scanner scn=new Scanner(System.in);
        int n=scn.nextInt();
        int[] a=new int[n];
        for(int i=0;i<n;i++)
        {
            a[i]=scn.nextInt();
```

	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	150	✗

Your code must pass all tests to earn any marks. Try again.

Show differences

Question **2**

Correct

Marked out of 5.00

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

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

After the operations are done, return the resultant array.

Example 1:

input1 = 4 (represents the number of elements in the input1 array)

input2 = {1, 5, 6, 9}

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

Explanation:

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

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

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

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

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

So, the expected output is the resultant array {-72, -36, -27, 0}.

Example 2:

input1 = 5 (represents the number of elements in the input1 array)

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

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

Explanation:

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

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

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

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

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

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

Example 3:

input1 = 2 (represents the number of elements in the input1 array)

input2 = {-9, 9}

Expected Output = {-162, 0}

Explanation:

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

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

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

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

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

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

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class arop{
    public static int[] perform(int[] input)
    {
        int max=Integer.MIN_VALUE;
        for(int num:input)
        {
            if(num>max)
            {
                max=num;
            }
        }
        int[] suba =new int[input.length];
        for(int i=0;i<input.length;i++)
        {
            suba[i]=input[i]-max;

            }
            int[] result=new int[input.length];
            for(int i=0;i<input.length;i++)
            {
                result[i]=suba[i]*max;

            }
            return result;
    }
    public static void main(String[] args)
    {
     Scanner scn=new Scanner(System.in);
     int n=scn.nextInt();
     int[] input=new int[n];
     for(int i=0;i<n;i++)
     {
         input[i]=scn.nextInt();
     }
     int[] res=perform(input);
     for(int num:res)
     {
         System.out.print(num+" ");
     }
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

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

This is explained below:

Example 1:

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

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

Step 1:

Starting from the 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 arraysum{
    public static int getpo(int num,int pos)
    {
        String numStr=Integer.toString(num);
        if(pos>=numStr.length())
        {
            return 0;
        }
        return Character.getNumericValue(numStr.charAt(numStr.length()-1-pos));
    }
```

```
    public static int calculatesum(int input1[])
    {
        int[] ga= new int[input1.length];
        for(int i=0;i<input1.length;i++)
        {
            ga[i]=getpo(input1[i],i);
        }
        int[] sa= new int[input1.length];
        for(int i=0;i<input1.length;i++)
        {
            sa[i]=ga[i]*ga[i];
        }
        int sum=0;
        for (int num:sa)
        {
            sum+=num;
        }
        return sum;
    }
public static void main(String [] args)
{
    Scanner scn=new Scanner(System.in);
    int n=scn.nextInt();
    int[] input1=new int[n];
    for(int i=0;i<n;i++)
    {
        input1[i]=scn.nextInt();
    }
    int result=calculatesum(input1);
    System.out.println(result);
}
}
```

	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	Sunday, 6 October 2024, 10:53 PM
Completed	Sunday, 6 October 2024, 11:19 PM
Duration	26 mins 18 secs

Question **1**

Correct

Marked out of 5.00

Create a Class Mobile with the attributes listed below,

private String manufacturer;
private String operating_system;
public String color;
private int cost;

Define a Parameterized constructor to initialize the above instance variables.

Define getter and setter methods for the attributes above.

for example : setter method for manufacturer is

void setManufacturer(String manufacturer){

this.manufacturer= manufacturer;

}

String getManufacturer(){

return manufacturer;}

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

For example:

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

Answer: (penalty regime: 0 %)

```
public class Mobile{
    private String mf;
    private String os;
    private String color;
    private int cost;
    public Mobile(String mf,String os,String color,int cost)
    {
        this.mf=mf;
        this.os=os;
        this.color=color;
        this.cost=cost;
    }
    public void setmf(String mf)
    {
        this.mf=mf;
    }
    public void setos(String os)
    {
        this.os=os;
    }
    public void setcost(int cost)
    {
        this.cost=cost;
    }
    public String getmf()
    {
        return mf;
    }
    public String getos()
    {
        return os;
    }
    public String getcolor()
    {
        return color;
    }
    public int getcost()
    {
        return cost;
```

```
    }
    public static void main(String[] args)
    {
        Mobile product=new Mobile("Redmi","Andriod","Blue",34000);
        System.out.println("manufacturer = "+product.getmf());
        System.out.println("operating_system = "+product.getos());
        System.out.println("color = "+product.getcolor());
        System.out.println("cost = "+product.getcost());
    }
}
```

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

Passed all tests! ✓

Question **2**

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 %)

```
import java.util.*;
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,int rollno)
    {
        this.name = name;
        this.rollno = rollno;
        System.out.println("2 arg constructor is invoked");
    }
    public Student(String name)
    {
        this.name = name;
        System.out.println("1 arg constructor is invoked");
    }
    public static void main(String[] args)
    {
        Student s1=new Student();
        Student s2=new Student("Rajalakshmi");
        Student s3=new Student("Lakshmi",101);
        System.out.println("Name ="+s1.name+" , Roll no = "+s1.rollno);
        System.out.println("Name ="+s2.name+" , Roll no = "+s2.rollno);
        System.out.println("Name ="+s3.name+" , Roll no = "+s3.rollno);
    }
}
```

	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 **3**

Correct

Marked out of 5.00

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

Area of Circle = πr^2

Circumference = $2\pi r$

Input:

2

Output:

Area = 12.57
Circumference = 12.57

For example:

Test	Input	Result
1	4	Area = 50.27 Circumference = 25.13

Answer: (penalty regime: 0 %)

Reset answer

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

    }
    public void setRadius(double radius){
        // set the radius


    }
    public double getRadius()    {
        // return the radius

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

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

```

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

Passed all tests! ✓

◄ Lab-04-MCQ

Jump to...

Number of Primes in a specified range ►

Status	Finished
Started	Wednesday, 2 October 2024, 2:34 PM
Completed	Wednesday, 2 October 2024, 3:10 PM
Duration	36 mins 14 secs

Question **1**

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

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

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

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

}
class cameraMobile extends Mobile{
    cameraMobile(){
        super();
        System.out.println("Camera Mobile is Manufactured");
    }
    void newFeature(){
        System.out.println("Camera Mobile with 5MG px");
    }
}

class AndroidMobile extends cameraMobile{
    AndroidMobile(){
       super();
       System.out.println("Android Mobile is Manufactured") ;
    }
    void androidMobile(){
        System.out.println("Touch Screen Mobile is Manufactured");
    }
}

public class Main{
    public static void main(String[] args){
        AndroidMobile a = new AndroidMobile();
       // a.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! ✓

◄ Lab-05-MCQ

Jump to...

Is Palindrome Number? ►

Status	Finished
Started	Monday, 4 November 2024, 12:26 PM
Completed	Monday, 4 November 2024, 12:30 PM
Duration	4 mins 11 secs

Question **1**

Correct

Marked out of 5.00

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

Note:

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

Case 1:

Check whether the two alphabets are same.

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

Example 1:

input1 = ww:ii:pp:rr:oo

output = WIPRO

Explanation:

word1 is ww, both are same hence take w

word2 is ii, both are same hence take i

word3 is pp, both are same hence take p

word4 is rr, both are same hence take r

word5 is oo, both are same hence take o

Hence the output is WIPRO

Case 2:

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

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

Example 2"

input1 = zx:za:ee

output = BYE

Explanation

word1 is zx, both are not same alphabets

position value of z is 26

position value of x is 24

max – min will be 26 – 24 = 2

Alphabet which comes in 2^{nd} 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 25^{th} 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 %)

```
import java.util.*;
public class Main{
    public static void main(String args[]){
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        StringBuilder sb = new StringBuilder();
        int diff;
        for(int i =0;i<str.length();i=i+3){
            if(str.charAt(i)==str.charAt(i+1)){
                sb.append(str.charAt(i));
            }
            else{
                diff = Math.abs(str.charAt(i)-str.charAt(i+1))+96;
                sb.append((char)diff);
            }
        }
        System.out.print((sb.toString()).toUpperCase());
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

For example:

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

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

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

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

If the word to be processed is "Nice":

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

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

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

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

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

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

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

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

Expected output:

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

Example 1:

input1 = "Today is a Nice Day"

input2 = 41

output = "iNce doTday"

Example 2:

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

input2 = 39

output = "naMngo arGpes"

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

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

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class Main{
    public static void main(String args[]){
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        int n = s.nextInt();
        int mid;
        String s1 = Integer.toString(n);
        String[] a= str.split(" ");
        for(int i=0;i<2;i++){
```

```
            StringBuilder sb = new StringBuilder();
            String word = a[((int)s1.charAt(j)-'0')-1];
            if (word.length() % 2 == 0){
                mid = (word.length()/2)-1;
                for(int i = mid;i>=0;i--){
                    sb.append(word.charAt(i));
                }
                for(int i=mid+1;i<word.length();i++){
                    sb.append(word.charAt(i));
                }
            }
            else{
            mid = word.length()/2;
            for(int i = mid;i>=0;i--){
                sb.append(word.charAt(i));
            }
            for(int i = mid;i<word.length();i++){
                sb.append(word.charAt(i));
            }}
            sb.append(" ");
            System.out.print(sb);
        }
    }
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Given 2 strings input1 & input2.

- Concatenate both the strings.
- Remove duplicate alphabets & white spaces.
- Arrange the alphabets in descending order.

Assumption 1:

There will either be alphabets, white spaces or null in both the inputs.

Assumption 2:

Both inputs will be in lower case.

Example 1:

Input 1: apple

Input 2: orange

Output: rponlgea

Example 2:

Input 1: fruits

Input 2: are good

Output: utsroigfeda

Example 3:

Input 1: ""

Input 2: ""

Output: null

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
class prog {
    public static String order(String s){
        int arr[]=new int[26];
        for(int i=0; i<s.length();i++){
            if(s.charAt(i)>=97 && s.charAt(i)<=122)
            arr[s.charAt(i)-'a']=1;
        }
        String ans="";
        for(int i=25;i>=0;i--){
            if(arr[i]==1) ans+=(char)(i+97);
        }
        return ans.length()==0?"null":ans;
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        String s1=sc.nextLine();
        String s2=sc.nextLine();
        System.out.println(order(s1+s2));
    }
}
```

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

Passed all tests! ✓

◄ Lab-06-MCQ

Jump to...

Return second word in Uppercase ►

Status	Finished
Started	Sunday, 6 October 2024, 2:10 PM
Completed	Sunday, 6 October 2024, 4:10 PM
Duration	1 hour 59 mins

Question **1**

Correct

Marked out of 5.00

RBI issues all national banks to collect interest on all customer loans.

Create an RBI interface with a variable String parentBank="RBI" and abstract method rateOfInterest().

RBI interface has two more methods default and static method.

default void policyNote() {

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

}

static void regulations(){

System.out.println("RBI has updated new regulations on 2024.");

}

Create two subclasses SBI and Karur which implements the RBI interface.

Provide the necessary code for the abstract method in two sub-classes.

Sample Input/Output:

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

For example:

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

Answer: (penalty regime: 0 %)

```
interface r{
    String pb="RBI";
    abstract void roi();
    public default void pn()
    {
        System.out.println(pb+"has a  new Policy issued in 2023");
    }
    public static void re()
    {
        System.out.println(pb+"has updated new regulations in 2024");

    }

}
class sbi implements r{
    public void pn()
    {
        System.out.println(pb+" has a new Policy issued in 2023");
    }
    public void re()
    {
        System.out.println(pb+" has updated new regulations in 2024.");
    }
    public void roi()
    {
        System.out.println("SBI rate of interest: 7.6 per annum.");
    }
}
class karur implements r{
    public void roi()
    {
        System.out.println("Karur rate of interest: 7.4 per annum.");
    }
}
public class hello{
    public static void main(String[] args)
    {
        sbi s=new sbi();
        karur k=new karur();
        s.pn();
        s.re();
        s.roi();
        k.roi();
```

```
        r.rbi();
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

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

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

Similarly, create Volleyball and Basketball classes.

Sample output:

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

For example:

Test	Input	Result
1	Sadhvin Sanjay Sruthi	Sadhvin is Playing football 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.*;
interface p{
    void p1();
}
class f implements p{
    String name;
    public f(String n)
    {
        this.name=n;
    }
    public void p1()
    {
        System.out.println(name+" is Playing football");
    }
}
class v implements p{
    String name;
    public v(String n)
    {
        this.name=n;
    }
    public void p1()
    {
        System.out.println(name+" is Playing volleyball");
    }
}
class b implements p
{
    String name;
    public b(String n)
    {
        this.name=n;
    }
    public void p1()
    {
        System.out.println(name+" is Playing basketball");
    }
}
public class hello{
```

```
    public static void main(String[] args)
{
    Scanner sc=new Scanner(System.in);
    f f1=new f(sc.next());
    v v1=new v(sc.next());
    b b1=new b(sc.next());
    f1.p1();
    v1.p1();
    b1.p1();
}
}
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Create interfaces shown below.

interface Sports {
public void setHomeTeam(String name);
public void setVisitingTeam(String name);
}
interface Football extends Sports {
public void homeTeamScored(int points);
public void visitingTeamScored(int points);}
create a class College that implements the Football interface and provides the necessary functionality to the abstract methods.

sample Input:

Rajalakshmi
Saveetha
22
21

Output:

Rajalakshmi 22 scored
Saveetha 21 scored
Rajalakshmi is the Winner!

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.Scanner;
 interface Sports {
public void setHomeTeam(String name);
public void setVisitingTeam(String name);

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

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

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

```
        String vteam=sc.next();
        int htpoints=sc.nextInt();
        int vtpoints=sc.nextInt();
   College s= new College();
    s.setHomeTeam(hname);
    s.setVisitingTeam(vteam);
    s.homeTeamScored(htpoints);
    s.visitingTeamScored(vtpoints);
    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! ✓

◄ Lab-07-MCQ

Jump to...

Generate series and find Nth element ►

Status	Finished
Started	Monday, 7 October 2024, 12:20 PM
Completed	Monday, 7 October 2024, 12:46 PM
Duration	26 mins 34 secs

Question **1**

Correct

Marked out of 5.00

1. Final Variable:

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

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

2. Final Method:

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

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

3. Final Class:

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

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

you should delete any piece of code.

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

```
class FinalExample
{
    int maxSpeed = 120;
    public final void displayMaxSpeed()
    {
        System.out.println("The maximum speed is: " + maxSpeed + " km/h");
    }
}
class SubClass extends FinalExample
{
    public void showDetails()
    {
        System.out.println("This is a subclass of FinalExample.");
    }
}
class prog
{
    public static void main(String[] args)
    {
        FinalExample obj = new FinalExample();
        obj.displayMaxSpeed();
        SubClass subObj = new SubClass();
        subObj.showDetails();
    }
}
```

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

Passed all tests! ✓

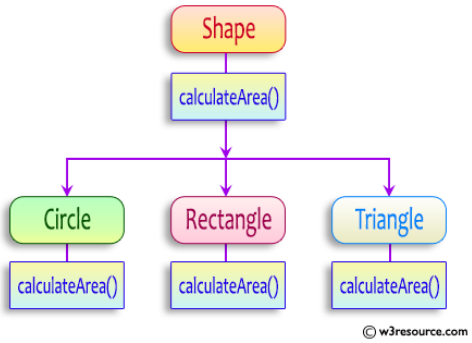
Question **2**

Correct

Marked out of 5.00

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

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




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

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

sample Input :

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

5 // length of the rectangle

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

4 // base of the triangle

3 // height of the triangle

OUTPUT:

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
abstract class s
{
    public abstract double calculateArea();
}
class c extends s
{
    double r;
    c(double r)
    {
        this.r=r;
    }
    public double calculateArea()
    {
        double a=Math.PI*r*r;
        System.out.printf("Area of a circle: %.2f\n",a);
        return a;
    }
```

```
}
class r extends s
{
    double l;
    double b;
    r(double l,double b)
    {
        this.l=l;
        this.b=b;
    }
    public double calculateArea()
    {
        double a=l*b;
        System.out.printf("Area of a Rectangle: %.2f\n",a);
        return a;
    }
}
class t extends s
{
    double b;
    double h;
    t(double b,double h)
    {
        this.b=b;
        this.h=h;
    }
    public double calculateArea()
    {
        double a=b*h*0.5;
        System.out.printf("Area of a Triangle: %.2f\n",a);
        return a;
    }
}
public class hello
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

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

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

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

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

input2: String array.

Example 1:

input1: 3

input2: {“oreo”, “sirish”, “apple”}

output: oreoapple

Example 2:

input1: 2

input2: {“Mango”, “banana”}

output: no matches found

Explanation:

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

Hence the output is no matches found.

Example 3:

input1: 3

input2: {“Ate”, “Ace”, “Girl”}

output: ateace

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class hello
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int k=0;
        String arr[]=new String[n];
        for(int i=0;i<n;i++)
        {
            arr[i]=sc.next();
            arr[i]=arr[i].toLowerCase();
            char ch=arr[i].charAt(0);
            if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u')
            {
                int z=arr[i].length();
                char x=arr[i].charAt(z-1);
                if (x=='a' || x=='e' || x=='i' || x=='o'|| x=='u')
                {
                    k=1;
                    System.out.print(arr[i]);
                }

            }
        }
```

```
        if(k==0)
        {
            System.out.println("no matches found");
        }

    }
}
```

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

Passed all tests! ✓

◄ Lab-08-MCQ

Jump to...

FindStringCode ►

Status	Finished
Started	Tuesday, 29 October 2024, 9:45 AM
Completed	Tuesday, 29 October 2024, 9:47 AM
Duration	2 mins 25 secs

Question **1**

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

```
1
2  class prog {
3    public static void main(String[] args) {
4      int n = 82;
5      trynumber(n);
6      n = 37;
7      trynumber(n);
8    }
9
10   public static void trynumber(int n) {
11     try {
12         checkEvenNumber(n);
13       System.out.println(n + " is even.");
14     } catch (Exception e) {
15       System.out.println("Error: " + e.getMessage());
16     }
17   }
18
19   public static void checkEvenNumber(int number) throws Exception {
20     if (number % 2 != 0) {
21       throw new Exception(number + " is odd.");
22     }
23   }
24 }
25
26
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

Write a Java program to handle ArithmeticException and ArrayIndexOutOfBoundsException.

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

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

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

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

Input:

5

10 0 20 30 40

Output:

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

Input:

3
10 20 30

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int size = s.nextInt();
        int[] array = new int[size];
        for (int i = 0; i < size; i++) {
            array[i] = s.nextInt();
        }

        try {
            int result = array[0] / array[1];
            System.out.println(" " + array[3]);
        } 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 **3**

Correct

Marked out of 5.00

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

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

Sample Input:

3
5 2 1

Sample Output:

8

Sample Input:

2

1 g

Sample Output:

```
You entered bad data.
```

For example:

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

Answer: (penalty regime: 0 %)

Reset answer

```
import java.util.Scanner;
import java.util.InputMismatchException;

class prog {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int length = sc.nextInt();
        int[] name = new int[length];
        int sum = 0;

        try {
            for (int i = 0; i < length; i++) {
                name[i] = sc.nextInt();
            }

            for (int num : name) {
                sum += num;
            }
            System.out.println(sum);
        } catch (InputMismatchException e) {
            System.out.println("You entered bad data.");
        } finally {
            sc.close();
        }
    }
}
```

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

Passed all tests! ✓

◄ Lab-09-MCQ

Jump to...

The “Nambiar Number” Generator ►

Status	Finished
Started	Monday, 18 November 2024, 9:18 PM
Completed	Monday, 18 November 2024, 9:49 PM
Duration	31 mins 9 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.ArrayList;
import java.util.Scanner;
public class Main{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<Integer> list = new ArrayList<>();
        int n=scanner.nextInt();
        for (int i=0;i<n;i++) {
            list.add(scanner.nextInt());
        }
        System.out.println("ArrayList: " + list);
        System.out.println("First : " + list.get(0) + ", Last : " + list.get(list.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());
        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);
        // Removing an element from position 3
        list.remove(3);
        System.out.print("ArrayList: " + list);
    }
}
```

	Test	Input	Expected	Got	
✓	1	5 1 2 3 100 5	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	✓

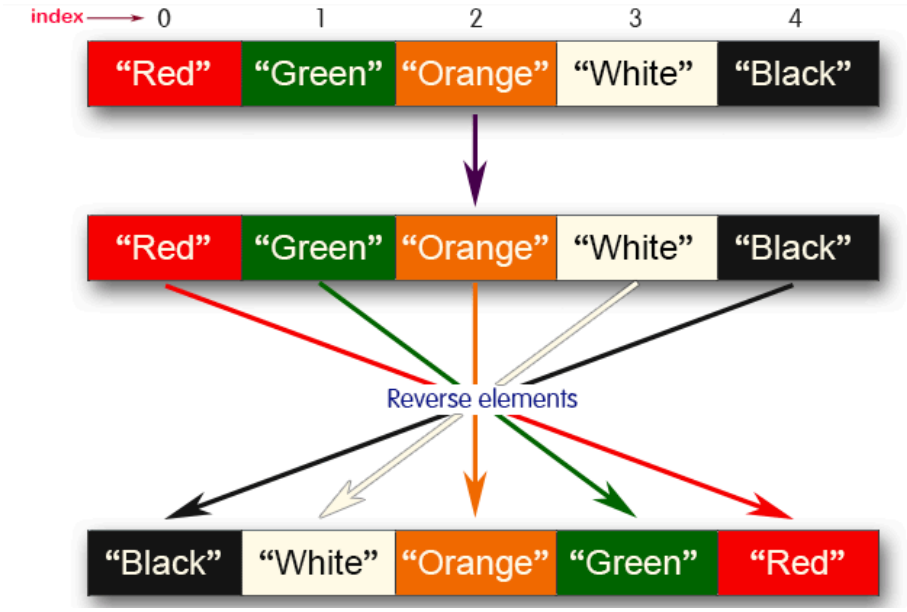
Passed all tests! ✓

Question **3**

Correct

Marked out of 1.00

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




```
Sample input and Output:
Red
Green
Orange
White
Black
Sample output
List before reversing :
[Red, Green, Orange, White, Black]
List after reversing :
[Black, White, Orange, Green, Red]
```

Answer: (penalty regime: 0 %)

```
import java.util.*;

public class Exercise11 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        List<String> list_Strings = new ArrayList<String>();
        int numberOfColors = scanner.nextInt();
        scanner.nextLine();

        for (int i = 0; i < numberOfColors; i++) {
            String color = scanner.nextLine();
            list_Strings.add(color);
        }

        System.out.println("List before reversing :\n" + list_Strings);

        Collections.reverse(list_Strings);

        System.out.println("List after reversing :\n" + list_Strings);

        scanner.close();
    }
}
```

	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	Monday, 18 November 2024, 9:15 PM
Completed	Monday, 18 November 2024, 9:49 PM
Duration	34 mins 37 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();
        HashSet<Integer> numbers = new HashSet<>();
        for (int i = 0; i < n; i++) {
            numbers.add(sc.nextInt());
        }
        int skey = sc.nextInt();

        if (numbers.contains(skey)) {
            System.out.println(skey + " was found in the set.");
        } else {
            System.out.println(skey + " was not found in the set.");
        }
        sc.close();
    }
}

```
```

	Test	Input	Expected	Got	
✓	1	5 90 56 45 78 25 78	78 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 %)

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

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

	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) {
        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);
        }
        Set<Entry<String, Integer>> entrySet = map.entrySet();
        for (Entry<String, Integer> entry : entrySet) {
            System.out.println(entry.getKey() + " : " + entry.getValue());
        }

        System.out.println("----------");
        HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();
        anotherMap.put("SIX", 6);
        anotherMap.put("SEVEN", 7);
        anotherMap.putAll(map);
        entrySet = anotherMap.entrySet();
        for (Entry<String, Integer> entry : entrySet) {
            System.out.println(entry.getKey() + " : " + entry.getValue());
        }
        map.putIfAbsent("FIVE", 5);
        Integer value = map.get("TWO");
        if (value != null) {
            System.out.println(value);
        } else {
            System.out.println("Key 'TWO' not found");
        }
        System.out.println(map.containsKey("ONE"));
        System.out.println(map.containsValue(3));
        System.out.println(map.size());
        sc.close();
    }
}
```

	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	Thursday, 21 November 2024, 10:49 AM
Completed	Thursday, 21 November 2024, 11:13 AM
Duration	24 mins 8 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.HashSet;
import java.util.Set;

public class CommonAsciiSum {
    public static int commonAsciiSingleDigitSum(char[] input1, char[] input2) {
        Set<Character> set1 = new HashSet<>();
        Set<Character> commonChars = new HashSet<>();

        for (char c : input1) {
            set1.add(c);
        }
        for (char c : input2) {
            if (set1.contains(c)) {
                commonChars.add(c);
            }
        }

        int sum1 = 0;
        for (char c : commonChars) {
            sum1 += (int) c;
        }

        while (sum1 >= 10) {
            int tempSum = 0;
            while (sum1 > 0) {
                tempSum += sum1 % 10;
                sum1 /= 10;
            }
            sum1 = tempSum;
```

```
        }

        return sum1;
    }

    public static void main(String[] args) {
        char[] input1 = {'a', 'b', 'c'};
        char[] input2 = {'b', 'c'};
        System.out.println(commonAsciiSingleDigitSum(input1, input2));  // Output: 8
    }
}
```

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

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

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

This sequence is the encoded version of a English word. You are supposed write a program to decode the provided string and find the original word.

Each alphabet is represented by a sequence of 0s.

This is as mentioned below:

Z : 0

Y : 00

X : 000

W : 0000

V : 00000

U : 000000

T : 0000000

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

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

Example 1:

input1: 010010001

The decoded string (original word) will be: ZYX

Example 2:

input1: 00001000000000000000001000000000001000000000100000000000001

The decoded string (original word) will be: WIPRO

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

For example:

Input	Result
010010001	ZYX
00001000000000000000001000000000001000000000100000000000001	WIPRO

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;

public class BinaryDecoder {
    public static String decode(String input) {
        StringBuilder decodedWord = new StringBuilder();
        String[] parts = input.split("1");

        for (String part : parts) {
            int length = part.length();
            if (length > 0 && length <= 26) {
                char letter = (char) ('A' + (26 - length));
                decodedWord.append(letter);
            }
        }

        return decodedWord.toString();
    }

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

        System.out.print("");
        String input = scanner.nextLine();

        String result = decode(input);
        System.out.println("" + result);
```

```
27
28          scanner.close();
29      }
30  }
```

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

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

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

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

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

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

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

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

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

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

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

Examples:

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

For example:

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

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;

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

        for (String word : words) {
            StringBuilder reversedWord = new StringBuilder(word).reverse();

            if (caseOption == 1) {
                for (int i = 0; i < word.length(); i++) {
                    char originalChar = word.charAt(i);
                    char reversedChar = reversedWord.charAt(i);

                    if (Character.isUpperCase(originalChar)) {
```

```
                        reversedWord.setCharAt(i, Character.toUpperCase(reversedChar));
                    } else if (Character.isLowerCase(originalChar)) {
                        reversedWord.setCharAt(i, Character.toLowerCase(reversedChar));
                    }
                }
            }

            if (reversedSentence.length() > 0) {
                reversedSentence.append(" ");
            }
            reversedSentence.append(reversedWord);
        }

        return reversedSentence.toString();
    }

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

        System.out.print("");
        String inputSentence = scanner.nextLine();

        System.out.print("");
        int caseOption = scanner.nextInt();

        String result = reverseWords(inputSentence, caseOption);
        System.out.println("" + result);

        scanner.close();
    }
}
```

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

Passed all tests! ✓

◄ Lab-12-MCQ

Jump to...

Identify possible words ►

Crime Analysis Platform

A MINI-PROJECT BY:

Laksheta SV 230701161
Miruthula KG 230701183

in partial fulfillment of the award of the degree

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project **"Crime analysis platform"** is the bonafide work of **"LAKSHETA SV(230701161) ,MIRUTHULA KG(230701183)"**

Submitted for the practical examination held on _____

SIGNATURE

Ms. Dharshini B S
Assistant Professor,
Computer Science and Engineering,
Rajalakshmi Engineering College (Autonomous),
Thandalam, Chennai-602 105

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The Crime Data Analysis System is a database-driven application designed to assist law enforcement agencies, policymakers, and researchers in managing crime-related data. This system serves as a comprehensive platform for recording, analyzing, and retrieving information about crimes, offenders, incident reports, and officer activities.

The project tests the ability to design and implement a scalable and relational database system, incorporating real-time data analysis and visualization to support crime prevention and resolution. While there is guidance in the development phase, the project showcases independent problem-solving and technical skills.

Our system aims to streamline crime data management and provide actionable insights. Crimes are recorded with details such as type, location, and date. Offenders' profiles are maintained, including risk levels and prior offenses. Reports on incidents and associated officer activities are tracked, ensuring accountability and transparency.

The system enables law enforcement agencies to access detailed crime data, analyze patterns, and make informed decisions. By facilitating seamless data storage and retrieval, the Crime Data Analysis System enhances the efficiency of crime tracking and resolution while supporting advanced analytical capabilities like identifying crime hotspots and offender patterns.

TABLE OF CONTENTS

INTRODUCTION

1.1 Introduction

The **Crime Data Analysis System** is designed to assist law enforcement agencies, policymakers, and researchers in managing and analyzing crime-related data. This platform offers a structured approach to store and retrieve crime information, offender profiles, incident reports, and related data. The system aims to enhance the efficiency of crime management, enable advanced analysis, and support decision-making processes to improve public safety.

1.2 Implementation

The Crime Data Analysis System is implemented using:
- **JavaFX:** For building the graphical user interface (GUI), providing an intuitive and interactive experience for end-users.
- **MySQL**: For managing relational data, including crimes, offenders, and reports, ensuring data integrity and scalability.
- **JDBC**: To connect the JavaFX frontend with the MySQL database, facilitating efficient data retrieval and manipulation.

1.3 Scope of the Project

The system provides a comprehensive solution for crime data management by integrating various functionalities, including data storage, querying, and visualization. It is designed to assist law enforcement agencies in tracking crimes, analyzing patterns, and allocating resources effectively. The platform's scalable structure allows for future enhancements, such as integrating predictive analytics or geospatial analysis.

1.4 Features

- **Crime Records Management**: Ability to store, retrieve, and update details of reported crimes.
- **Offender Profiling**: Detailed profiles of offenders, including risk assessment and history of offenses.
- **Incident Reporting**: Record and track the status of crime investigations.
- **Location Tracking**: Geographical representation of crime hotspots.
- **Analytics Dashboard**: Displays trends and summaries of crime data.

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS:

PROCESSOR : Intel i5

MEMORY SIZE : 4GB(Minimum)

HARD DISK : 500 GB of free space

2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE : Java, MySQL

FRONT-END : Java

BACK-END : MySQL

OPERATING SYSTEM : Windows 10

3.1 Main page design :

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.layout.HBox?>
<?import javafx.geometry.Insets?>

<VBox xmlns:fx="http://javafx.com/fxml"
fx:controller="com.crimedata.controllers.MainController" spacing="10" style="-fx-
background-color: #e0e0e0;">
   <!-- Header Section -->
   <HBox alignment="CENTER" spacing="20" style="-fx-background-color: #A7C7E7;
-fx-padding: 15;">
      <Button text="Dashboard" onAction="#handleDashboard" style="-fx-font-size:
14px; -fx-background-color: #ffffff;" />
      <Button text="Crime Records" onAction="#handleCrimeRecords" style="-fx-font-
size: 14px; -fx-background-color: #ffffff;" />
```

```
        <Button text="Incident Reports" onAction="#handleIncidentReports" style="-fx-font-size: 14px; -fx-background-color: #ffffff;" />

    </HBox>

    <!-- Content Pane where pages will load -->
    <Pane fx:id="contentPane" style="-fx-background-color: #f0f0f0; -fx-padding: 20;" VBox.vgrow="ALWAYS">
        <!-- This pane will be filled with dynamic content based on navigation -->
    </Pane>

</VBox>
```

3.2 Dashboard page design:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.layout.VBox?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.text.Font?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.BorderPane?>



<BorderPane xmlns:fx="http://javafx.com/fxml"
fx:controller="com.crimedata.controllers.DashboardController"  style="-fx-background-
color: #f7f7f7;">


   <top>
      <HBox alignment="CENTER" style="-fx-background-color: #3f51b5; -fx-padding:
20;">
         <Label text="Crime Data Analysis System" style="-fx-font-size: 24px; -fx-text-fill:
white; -fx-font-weight: bold;"/>
      </HBox>
   </top>


   <center>
      <VBox alignment="CENTER" spacing="15" style="-fx-padding: 40;">
         <HBox alignment="CENTER" spacing="30">
            <!-- Total Crimes Card -->
            <VBox alignment="CENTER" style="-fx-background-color: #e3f2fd; -fx-padding:
20; -fx-border-radius: 10; -fx-background-radius: 10; -fx-min-width: 150;">
               <Label text="Total Crimes" style="-fx-font-size: 18px; -fx-font-weight: bold;"/>
               <Label fx:id="totalCrimes" text="0" style="-fx-font-size: 30px; -fx-font-weight:
bold; -fx-text-fill: #1976d2;"/>
            </VBox>

            <!-- Total Offenders Card -->
            <VBox alignment="CENTER" style="-fx-background-color: #fce4ec; -fx-padding:
20; -fx-border-radius: 10; -fx-background-radius: 10; -fx-min-width: 150;">
```

```
                <Label text="Total Offenders" style="-fx-font-size: 18px; -fx-font-weight: bold;"/>
                <Label fx:id="totalOffenders" text="0" style="-fx-font-size: 30px; -fx-font-weight: bold; -fx-text-fill: #d81b60;"/>
            </VBox>

            <!-- Open Cases Card -->
            <VBox alignment="CENTER" style="-fx-background-color: #e8f5e9; -fx-padding: 20; -fx-border-radius: 10; -fx-background-radius: 10; -fx-min-width: 150;">
                <Label text="Open Cases" style="-fx-font-size: 18px; -fx-font-weight: bold;"/>
                <Label fx:id="openCases" text="0" style="-fx-font-size: 30px; -fx-font-weight: bold; -fx-text-fill: #388e3c;"/>
            </VBox>
        </HBox>


        <HBox alignment="CENTER" spacing="20" style="-fx-padding: 20;">
            <Label text="Explore more crime statistics" style="-fx-font-size: 16px;"/>
        </HBox>
    </VBox>
  </center>


  <bottom>
    <HBox alignment="CENTER" style="-fx-background-color: #3f51b5; -fx-padding: 10;">
        <Label text="© 2024 Crime Data Analysis System" style="-fx-font-size: 12px; -fx-text-fill: white;"/>
    </HBox>
  </bottom>
</BorderPane>
```

3.3 Crime Records Page design:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<AnchorPane xmlns:fx="http://javafx.com/fxml"
fx:controller="com.crimedata.controllers.CrimeRecordsController">
   <children>
      <!-- Crime Type Field -->
      <Label text="Crime Type:" layoutX="20" layoutY="20"/>
      <TextField fx:id="typeField" layoutX="100" layoutY="20" promptText="Enter crime
type"/>

      <!-- Crime Location Field -->
      <Label text="Crime Location:" layoutX="20" layoutY="60"/>
      <TextField fx:id="locationField" layoutX="100" layoutY="60" promptText="Enter
crime location"/>

      <!-- Crime Date Field -->
      <Label text="Crime Date:" layoutX="20" layoutY="100"/>
      <TextField fx:id="dateField" layoutX="100" layoutY="100" promptText="Enter crime
date"/>

      <!-- Add Crime Button -->
      <Button text="Add Crime" layoutX="20" layoutY="140" onAction="#addCrime"/>

      <!-- Delete Crime Button -->
      <Button text="Delete Crime" layoutX="120" layoutY="140"
onAction="#deleteCrime"/>

      <!-- Crime Table -->
      <TableView fx:id="crimeTable" layoutX="20" layoutY="180" prefWidth="460"
prefHeight="200">
         <columns>
            <TableColumn fx:id="crimeIdColumn" text="ID" prefWidth="50"/>
            <TableColumn fx:id="crimeTypeColumn" text="Type" prefWidth="130"/>
            <TableColumn fx:id="crimeLocationColumn" text="Location" prefWidth="130"/>
            <TableColumn fx:id="crimeDateColumn" text="Date" prefWidth="130"/>
         </columns>
      </TableView>
```

```
    </children>
</AnchorPane>
```

3.4 Incident Reports page design :

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Alert?>
<?import javafx.scene.control.Alert.AlertType?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.HBox?>

<GridPane fx:controller="com.crimedata.controllers.IncidentReportsController"
          xmlns:fx="http://javafx.com/fxml/1" alignment="CENTER" hgap="10" vgap="10">
    <!-- Table for displaying incident reports -->
    <TableView fx:id="incidentReportsTable" GridPane.columnSpan="2"
GridPane.rowIndex="0">
        <columns>
            <TableColumn fx:id="reportIdColumn" text="Report ID" />
            <TableColumn fx:id="crimeIdColumn" text="Crime ID" />
            <TableColumn fx:id="reportDescriptionColumn" text="Description" />
            <TableColumn fx:id="reportDateColumn" text="Report Date" />
        </columns>
    </TableView>

    <!-- Form fields for adding new reports -->
    <TextField fx:id="reportIdField" promptText="Report ID" GridPane.rowIndex="1"
GridPane.columnIndex="0" />
    <TextField fx:id="crimeIdField" promptText="Crime ID" GridPane.rowIndex="1"
GridPane.columnIndex="1" />
    <TextField fx:id="descriptionField" promptText="Description" GridPane.rowIndex="2"
GridPane.columnIndex="0" GridPane.columnSpan="2" />
    <TextField fx:id="dateField" promptText="Date" GridPane.rowIndex="3"
GridPane.columnIndex="0" GridPane.columnSpan="2" />
```

```
    <!-- Submit Button -->
    <HBox spacing="10" GridPane.rowIndex="4" GridPane.columnSpan="2"
alignment="CENTER">
        <Button text="Submit Report" onAction="#handleSubmitReport"/>
    </HBox>
</GridPane>
```

3.5 **Offender profiles page design :**

```
 <?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Alert?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.control.Label?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.input.KeyCode?>

<AnchorPane xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.crimedata.controllers.OffenderProfilesController">
    <!-- Main Layout -->
    <GridPane layoutX="14.0" layoutY="14.0" vgap="15" hgap="15" padding="20">
        <!-- Table for Offender List -->
        <TableView fx:id="offenderTable" GridPane.columnSpan="2" prefHeight="400.0"
prefWidth="600.0" style="-fx-background-color: #f9f9f9;">
            <columns>
                <TableColumn fx:id="offenderIdColumn" text="ID" prefWidth="100"
cellValueFactory="new PropertyValueFactory<Offender, Integer>('offenderId')" />
                <TableColumn fx:id="firstNameColumn" text="First Name" prefWidth="150"
cellValueFactory="new PropertyValueFactory<Offender, String>('firstName')" />
                <TableColumn fx:id="lastNameColumn" text="Last Name" prefWidth="150"
cellValueFactory="new PropertyValueFactory<Offender, String>('lastName')" />
                <TableColumn fx:id="ageColumn" text="Age" prefWidth="100"
cellValueFactory="new PropertyValueFactory<Offender, Integer>('age')" />
                <TableColumn fx:id="genderColumn" text="Gender" prefWidth="100"
cellValueFactory="new PropertyValueFactory<Offender, String>('gender')" />
```

```
            <TableColumn fx:id="priorOffensesCountColumn" text="Prior Offenses Count"
prefWidth="150" cellValueFactory="new PropertyValueFactory<Offender,
Integer>('priorOffensesCount')" />
            <TableColumn fx:id="riskLevelColumn" text="Risk Level" prefWidth="150"
cellValueFactory="new PropertyValueFactory<Offender, String>('riskLevel')" />
            <TableColumn fx:id="lastArrestDateColumn" text="Last Arrest Date"
prefWidth="150" cellValueFactory="new PropertyValueFactory<Offender,
String>('lastArrestDate')" />
        </columns>
    </TableView>

    <!-- Add Offender Form -->
    <Label text="Add Offender Profile" GridPane.rowIndex="1" style="-fx-font-size:
18px; -fx-font-weight: bold;" />

    <!-- Input Fields -->
    <GridPane GridPane.rowIndex="2" GridPane.columnSpan="2" vgap="10" hgap="10">
        <Label text="First Name" />
        <TextField fx:id="nameField" promptText="Enter First Name"
GridPane.columnIndex="1" prefWidth="200.0"/>

        <Label text="Risk Level" GridPane.rowIndex="1" />
        <TextField fx:id="riskLevelField" promptText="Enter Risk Level"
GridPane.columnIndex="1" GridPane.rowIndex="1" prefWidth="200.0" />
    </GridPane>

    <!-- Add Offender Button -->
    <HBox GridPane.rowIndex="3" GridPane.columnSpan="2" spacing="15"
alignment="center">
        <Button layoutX="300.0" layoutY="450.0" text="Add Offender"
onAction="#addOffender" style="-fx-background-color: #4CAF50; -fx-text-fill: white; -fx-
font-weight: bold;" tooltip="Click to Add Offender" />
    </HBox>
</GridPane>

<!-- Styling for TableView Rows (Alternating Colors) -->
<style>
    .table-row-cell:nth-child(odd) {
        -fx-background-color: #f1f1f1;
    }
    .table-row-cell:nth-child(even) {
```

```
        -fx-background-color: #ffffff;
    }
    .table-header {
        -fx-background-color: #2196F3;
        -fx-text-fill: white;
    }
  </style>
</AnchorPane>
```

Controller pages for javafx:

3.6 Main page Controller:

```
package com.crimedata.controllers;

import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.layout.Pane;
import java.io.IOException;

public class MainController {

  @FXML
  private Pane contentPane; // This is the container where pages will load dynamically

  // Method to load a page into the content pane
  private void loadPage(String fxmlFile) {
    try {
      // Load the specified FXML file
      FXMLLoader loader = new
FXMLLoader(getClass().getResource("/com/crimedata/views/" + fxmlFile));
      Parent root = loader.load();

      // Clear existing content and set the new content
      contentPane.getChildren().clear();
      contentPane.getChildren().add(root);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

  // Event handler for loading the Dashboard page
```

```
    @FXML
    private void handleDashboard() {
        loadPage("Dashboard.fxml");
    }

    // Event handler for loading the Crime Records page
    @FXML
    private void handleCrimeRecords() {
        loadPage("CrimeRecords.fxml");
    }

    // Event handler for loading the Offender Profiles page
    @FXML
    private void handleOffenderProfiles() {
        loadPage("OffenderProfiles.fxml");
    }

    // Event handler for loading the Incident Reports page
    @FXML
    private void handleIncidentReports() {
        loadPage("IncidentReports.fxml");
    }

    // Event handler for loading the Analytics page
    @FXML
    private void handleAnalytics() {
        loadPage("Analytics.fxml");
    }
}
```

3.7 Dashboard controller page :

```
package com.crimedata.controllers;

import com.crimedata.utils.DatabaseUtil;
import javafx.fxml.FXML;
import javafx.scene.control.Label;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class DashboardController {
```

```
    @FXML
    private Label totalCrimes;
    @FXML
    private Label totalOffenders;
    @FXML
    private Label openCases;
    public void initialize() {
        Connection conn = DatabaseUtil.getConnection();
        if (conn != null) {
            System.out.println("Connection successful!");
        } else {
            System.out.println("Connection failed!");
        }
        loadStatistics();
    }

    private void loadStatistics() {
        try (Connection conn = DatabaseUtil.getConnection()) {
            // Define the queries
            String crimeQuery = "SELECT COUNT(*) FROM crimes";
            String offenderQuery = "SELECT COUNT(*) FROM offenders";
            String openCasesQuery = "SELECT COUNT(*) FROM incidentreports WHERE
case_status = 'Open'";

            // Execute the queries using PreparedStatements
            try (PreparedStatement crimeStmt = conn.prepareStatement(crimeQuery);
                 PreparedStatement offenderStmt = conn.prepareStatement(offenderQuery);
                 PreparedStatement openCasesStmt = conn.prepareStatement(openCasesQuery)) {

                // Get results for crimes count
                ResultSet crimeResult = crimeStmt.executeQuery();
                if (crimeResult.next()) {
                    totalCrimes.setText(String.valueOf(crimeResult.getInt(1)));
                }

                // Get results for offenders count
                ResultSet offenderResult = offenderStmt.executeQuery();
                if (offenderResult.next()) {
                    totalOffenders.setText(String.valueOf(offenderResult.getInt(1)));
                }
```

```
            // Get results for open cases count
            ResultSet openCasesResult = openCasesStmt.executeQuery();
            if (openCasesResult.next()) {
                openCases.setText(String.valueOf(openCasesResult.getInt(1)));
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();  // Catch other exceptions as well
    }
  }
}
```

3.8 Crime Records controller page:

```
package com.crimedata.controllers;

import com.crimedata.models.Crime;
import com.crimedata.utils.DatabaseUtil;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.ButtonType;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.cell.PropertyValueFactory;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class CrimeRecordsController {

    @FXML
    private TableView<Crime> crimeTable;
    @FXML
    private TableColumn<Crime, Integer> crimeIdColumn;
    @FXML
    private TableColumn<Crime, String> crimeTypeColumn;
    @FXML
```

```
    private TableColumn<Crime, String> crimeLocationColumn;
    @FXML
    private TableColumn<Crime, String> crimeDateColumn;
    @FXML
    private TextField typeField, locationField, dateField;

    public void initialize() {
        // Setting up columns to bind to Crime properties
        crimeIdColumn.setCellValueFactory(new PropertyValueFactory<>("id"));
        crimeTypeColumn.setCellValueFactory(new PropertyValueFactory<>("type"));
        crimeLocationColumn.setCellValueFactory(new PropertyValueFactory<>("location"));
        crimeDateColumn.setCellValueFactory(new PropertyValueFactory<>("date"));

        // Load initial crime data
        loadCrimes();
    }

    private void loadCrimes() {
        crimeTable.getItems().clear();  // Clear existing items
        try (Connection conn = DatabaseUtil.getConnection()) {
            String query = "SELECT * FROM crimes";
            PreparedStatement stmt = conn.prepareStatement(query);
            ResultSet rs = stmt.executeQuery();

            while (rs.next()) {
                Crime crime = new Crime(rs.getInt("crime_id"), rs.getString("crime_type"),
rs.getString("crime_location"), rs.getString("crime_date"));
                System.out.println("Adding crime: " + crime.getType());  // Debug print
                crimeTable.getItems().add(crime);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    @FXML
    private void addCrime() {
        try (Connection conn = DatabaseUtil.getConnection()) {
            String query = "INSERT INTO crimes (crime_type, crime_location, crime_date)
VALUES (?, ?, ?)";
            PreparedStatement stmt = conn.prepareStatement(query);
```

```
            stmt.setString(1, typeField.getText());
            stmt.setString(2, locationField.getText());
            stmt.setString(3, dateField.getText());
            stmt.executeUpdate();

            // Refresh the list after adding a crime
            loadCrimes();

            // Show success alert
            Alert alert = new Alert(AlertType.INFORMATION);
            alert.setTitle("Success");
            alert.setHeaderText("Crime Record Added");
            alert.setContentText("The crime record has been successfully added.");
            alert.showAndWait();
        } catch (Exception e) {
            e.printStackTrace();

            // Show error alert if something goes wrong
            Alert alert = new Alert(AlertType.ERROR);
            alert.setTitle("Error");
            alert.setHeaderText("Failed to Add Crime");
            alert.setContentText("An error occurred while adding the crime record. Please try
again.");
            alert.showAndWait();
        }
    }

    @FXML
    private void deleteCrime() {
        Crime selectedCrime = crimeTable.getSelectionModel().getSelectedItem();
        if (selectedCrime != null) {
            // Show confirmation alert
            Alert confirmationAlert = new Alert(Alert.AlertType.CONFIRMATION);
            confirmationAlert.setTitle("Confirmation");
            confirmationAlert.setHeaderText("Delete Crime Record");
            confirmationAlert.setContentText("Are you sure you want to delete the selected
crime record?");

            // Wait for user response
            if (confirmationAlert.showAndWait().get() == ButtonType.OK) {
                try (Connection conn = DatabaseUtil.getConnection()) {
```

```
                // Use the correct column name in the DELETE query
                String query = "DELETE FROM crimes WHERE crime_id = ?";
                PreparedStatement stmt = conn.prepareStatement(query);
                stmt.setInt(1, selectedCrime.getId());  // Ensure you are passing the correct 'crime_id'
                int rowsAffected = stmt.executeUpdate();

                // If rows are affected, proceed with the deletion
                if (rowsAffected > 0) {
                    crimeTable.getItems().remove(selectedCrime);
                    // Show success alert
                    Alert successAlert = new Alert(AlertType.INFORMATION);
                    successAlert.setTitle("Success");
                    successAlert.setHeaderText("Crime Record Deleted");
                    successAlert.setContentText("The crime record has been successfully deleted.");
                    successAlert.showAndWait();
                } else {
                    // If no rows are affected, something went wrong
                    Alert errorAlert = new Alert(AlertType.ERROR);
                    errorAlert.setTitle("Error");
                    errorAlert.setHeaderText("Failed to Delete Crime");
                    errorAlert.setContentText("No crime record was deleted. Please try again.");
                    errorAlert.showAndWait();
                }
            } catch (Exception e) {
                e.printStackTrace();

                // Show error alert if something goes wrong
                Alert errorAlert = new Alert(AlertType.ERROR);
                errorAlert.setTitle("Error");
                errorAlert.setHeaderText("Failed to Delete Crime");
                errorAlert.setContentText("An error occurred while deleting the crime record. Please try again.");
                errorAlert.showAndWait();
            }
        }
    } else {
        // Show alert if no item is selected
        Alert noSelectionAlert = new Alert(AlertType.WARNING);
        noSelectionAlert.setTitle("No Selection");
```

```
            noSelectionAlert.setHeaderText("No Crime Record Selected");
            noSelectionAlert.setContentText("Please select a crime record to delete.");
            noSelectionAlert.showAndWait();
        }
    }
}
```

3.9 Incident Reports controller :

```
package com.crimedata.controllers;

import com.crimedata.models.IncidentReport;
import com.crimedata.utils.DatabaseUtil;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.cell.PropertyValueFactory;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

public class IncidentReportsController {

    @FXML
    private TableView<IncidentReport> incidentReportsTable;

    @FXML
    private TableColumn<IncidentReport, Integer> reportIdColumn;
    @FXML
    private TableColumn<IncidentReport, Integer> crimeIdColumn;
    @FXML
    private TableColumn<IncidentReport, String> reportDescriptionColumn;
    @FXML
    private TableColumn<IncidentReport, String> reportDateColumn;

    @FXML
    private TextField reportIdField;
```

```
    @FXML
    private TextField crimeIdField;
    @FXML
    private TextField descriptionField;
    @FXML
    private TextField dateField;

    private List<IncidentReport> reportsList = new ArrayList<>();

    @FXML
    public void initialize() {
        // Set up TableView columns
        reportIdColumn.setCellValueFactory(new PropertyValueFactory<>("report_Id"));
        crimeIdColumn.setCellValueFactory(new PropertyValueFactory<>("crime_Id"));
        reportDescriptionColumn.setCellValueFactory(new
PropertyValueFactory<>("report_notes"));
        reportDateColumn.setCellValueFactory(new PropertyValueFactory<>("report_date"));

        // Load incident reports from the database
        loadReports();
    }

    @FXML
    private void handleSubmitReport() {
        try {
            // Get input values
            int reportId = Integer.parseInt(reportIdField.getText());
            int crimeId = Integer.parseInt(crimeIdField.getText());
            String description = descriptionField.getText();
            String date = dateField.getText();

            // Insert the new report into the database
            addReportToDatabase(reportId, crimeId, description, date);

            // Clear the input fields after submission
            reportIdField.clear();
            crimeIdField.clear();
            descriptionField.clear();
            dateField.clear();

            // Reload reports from the database
```

```
            loadReports();

            // Optionally, show success message
            Alert alert = new Alert(AlertType.INFORMATION);
            alert.setTitle("Report Submitted");
            alert.setHeaderText("Incident Report Added");
            alert.setContentText("The incident report has been successfully added.");
            alert.showAndWait();

        } catch (NumberFormatException e) {
            // Handle invalid input (non-numeric for ID fields)
            Alert alert = new Alert(AlertType.ERROR);
            alert.setTitle("Invalid Input");
            alert.setHeaderText("Input Error");
            alert.setContentText("Please enter valid numbers for Report ID and Crime ID.");
            alert.showAndWait();
        }
    }

    private void addReportToDatabase(int reportId, int crimeId, String description, String
date) {
        try (Connection conn = DatabaseUtil.getConnection()) {
            // SQL query to insert the report into the database
            String query = "INSERT INTO incidentreports (report_id, crime_id, report_notes,
report_date) VALUES (?, ?, ?, ?)";
            PreparedStatement stmt = conn.prepareStatement(query);
            stmt.setInt(1, reportId);
            stmt.setInt(2, crimeId);
            stmt.setString(3, description);
            stmt.setString(4, date);
            stmt.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
            Alert alert = new Alert(AlertType.ERROR);
            alert.setTitle("Database Error");
            alert.setHeaderText("Error Adding Report");
            alert.setContentText("There was an error adding the incident report to the database.");
            alert.showAndWait();
        }
    }
```

```
    private void loadReports() {
        reportsList.clear(); // Clear the list before adding updated data
        try (Connection conn = DatabaseUtil.getConnection()) {
            String query = "SELECT * FROM incidentreports";
            PreparedStatement stmt = conn.prepareStatement(query);
            ResultSet rs = stmt.executeQuery();

            // Loop through the result set and populate the reports list
            while (rs.next()) {
                IncidentReport report = new IncidentReport(
                        rs.getInt("report_id"),
                        rs.getInt("crime_id"),
                        rs.getString("report_notes"),
                        rs.getString("report_date")
                );
                reportsList.add(report);
            }

            // Bind the list to the TableView
            incidentReportsTable.getItems().setAll(reportsList);
        } catch (Exception e) {
            e.printStackTrace();
            Alert alert = new Alert(AlertType.ERROR);
            alert.setTitle("Database Error");
            alert.setHeaderText("Error Loading Reports");
            alert.setContentText("There was an error loading the incident reports from the
database.");
            alert.showAndWait();
        }
    }
}
```

3.10 Offender profiles controller page:

```
package com.crimedata.controllers;

import com.crimedata.models.Offender;
import javafx.beans.property.SimpleIntegerProperty;
import javafx.beans.property.SimpleStringProperty;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
```

```
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;

public class OffenderProfilesController {

    @FXML
    private TableView<Offender> offenderTable;

    @FXML
    private TableColumn<Offender, Integer> offenderIdColumn;
    @FXML
    private TableColumn<Offender, String> firstNameColumn;
    @FXML
    private TableColumn<Offender, String> lastNameColumn;
    @FXML
    private TableColumn<Offender, Integer> ageColumn;
    @FXML
    private TableColumn<Offender, String> genderColumn;
    @FXML
    private TableColumn<Offender, Integer> priorOffensesCountColumn;
    @FXML
    private TableColumn<Offender, String> riskLevelColumn;
    @FXML
    private TableColumn<Offender, String> lastArrestDateColumn;

    @FXML
    private TextField nameField;
    @FXML
    private TextField riskLevelField;

    private ObservableList<Offender> offenderList;

    // Initialize method to set up table columns and load data
    public void initialize() {
        offenderList = FXCollections.observableArrayList();
        offenderTable.setItems(offenderList);

        // Set up table columns
```

```
        offenderIdColumn.setCellValueFactory(cellData -> new
SimpleIntegerProperty(cellData.getValue().getOffenderId()).asObject());
        firstNameColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getFirstName()));
        lastNameColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getLastName()));
        ageColumn.setCellValueFactory(cellData -> new
SimpleIntegerProperty(cellData.getValue().getAge()).asObject());
        genderColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getGender()));
        priorOffensesCountColumn.setCellValueFactory(cellData -> new
SimpleIntegerProperty(cellData.getValue().getPriorOffensesCount()).asObject());
        riskLevelColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getRiskLevel()));
        lastArrestDateColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getLastArrestDate()));
    }

    // Action for the "Add Offender" button
    @FXML
    public void addOffender() {
        String firstName = nameField.getText();
        String riskLevel = riskLevelField.getText();

        if (firstName.isEmpty() || riskLevel.isEmpty()) {
            showErrorDialog("Form Error", "Please fill out all fields.");
            return;
        }

        // Sample data for the new offender
        int offenderId = offenderList.size() + 1; // Auto-increment offender ID
        String lastName = "Doe"; // Use default last name for simplicity
        int age = 30; // Placeholder age
        String gender = "M"; // Placeholder gender
        int priorOffensesCount = 0; // Placeholder for prior offenses
        String lastArrestDate = "N/A"; // Placeholder for last arrest date

        // Create a new offender object
        Offender newOffender = new Offender(offenderId, firstName, lastName, age, gender,
priorOffensesCount, riskLevel, lastArrestDate);
```

```
        // Add the new offender to the list and refresh the table
        offenderList.add(newOffender);

        // Clear the input fields
        nameField.clear();
        riskLevelField.clear();
    }

    // Method to show error dialog
    private void showErrorDialog(String title, String message) {
        Alert alert = new Alert(AlertType.ERROR);
        alert.setTitle(title);
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }
}
```

Models :

3.11 Crime .java

```
package com.crimedata.models;

import javafx.beans.property.IntegerProperty;
import javafx.beans.property.SimpleIntegerProperty;
import javafx.beans.property.StringProperty;
import javafx.beans.property.SimpleStringProperty;

public class Crime {

    private IntegerProperty id;
    private StringProperty type;
    private StringProperty location;
    private StringProperty date;

    public Crime(int id, String type, String location, String date) {
        this.id = new SimpleIntegerProperty(id);
        this.type = new SimpleStringProperty(type);
        this.location = new SimpleStringProperty(location);
        this.date = new SimpleStringProperty(date);
    }

    // Getters and Setters
```

```
public int getId() {
    return id.get();
}

public void setId(int id) {
    this.id.set(id);
}

public String getType() {
    return type.get();
}

public void setType(String type) {
    this.type.set(type);
}

public String getLocation() {
    return location.get();
}

public void setLocation(String location) {
    this.location.set(location);
}

public String getDate() {
    return date.get();
}

public void setDate(String date) {
    this.date.set(date);
}

// Property methods for binding
public IntegerProperty idProperty() {
    return id;
}

public StringProperty typeProperty() {
    return type;
}
```

```
    public StringProperty locationProperty() {
        return location;
    }

    public StringProperty dateProperty() {
        return date;
    }
}
```

3.12 Crime data.java:

```
package com.crimedata.models;

import javafx.beans.property.SimpleStringProperty;
import javafx.beans.property.StringProperty;

public class CrimeData {

    private StringProperty crimeType;
    private StringProperty crimeLocation;
    private StringProperty crimeDate;
    private StringProperty offenderName;
    private StringProperty caseID;
    private StringProperty caseStatus;

    public CrimeData(String crimeType, String crimeLocation, String crimeDate,
                     String offenderName, String caseID, String caseStatus) {
        this.crimeType = new SimpleStringProperty(crimeType);
        this.crimeLocation = new SimpleStringProperty(crimeLocation);
        this.crimeDate = new SimpleStringProperty(crimeDate);
        this.offenderName = new SimpleStringProperty(offenderName);
        this.caseID = new SimpleStringProperty(caseID);
        this.caseStatus = new SimpleStringProperty(caseStatus);
    }

    public String getCrimeType() {
        return crimeType.get();
    }

    public String getCrimeLocation() {
        return crimeLocation.get();
    }
```

```
    public String getCrimeDate() {
        return crimeDate.get();
    }

    public String getOffenderName() {
        return offenderName.get();
    }

    public String getCaseID() {
        return caseID.get();
    }

    public String getCaseStatus() {
        return caseStatus.get();
    }

    // Getters for properties to bind with TableView
    public StringProperty crimeTypeProperty() {
        return crimeType;
    }

    public StringProperty crimeLocationProperty() {
        return crimeLocation;
    }

    public StringProperty crimeDateProperty() {
        return crimeDate;
    }

    public StringProperty offenderNameProperty() {
        return offenderName;
    }

    public StringProperty caseIDProperty() {
        return caseID;
    }

    public StringProperty caseStatusProperty() {
        return caseStatus;
    }
}
```

3.13 Incidentreport .java :

```
package com.crimedata.models;

public class IncidentReport {

    private int reportId;
    private int crimeId;
    private String description;
    private String date;

    // Constructor
    public IncidentReport(int reportId, int crimeId, String description, String date) {
        this.reportId = reportId;
        this.crimeId = crimeId;
        this.description = description;
        this.date = date;
    }

    // Getters and Setters
    public int getReportId() {
        return reportId;
    }

    public void setReportId(int reportId) {
        this.reportId = reportId;
    }

    public int getCrimeId() {
        return crimeId;
    }

    public void setCrimeId(int crimeId) {
        this.crimeId = crimeId;
    }

    public String getDescription() {
        return description;
    }

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

```
    }

    public String getDate() {
        return date;
    }

    public void setDate(String date) {
        this.date = date;
    }

    // toString method (optional but helpful for debugging)
    @Override
    public String toString() {
        return "IncidentReport [reportId=" + reportId + ", crimeId=" + crimeId + ",
description=" + description + ", date=" + date + "]";
    }
}
```

3.14 Offender .java :

```
package com.crimedata.models;

public class Offender {
    private int offenderId;
    private String firstName;
    private String lastName;
    private int age;
    private String gender;
    private int priorOffensesCount;
    private String riskLevel;
    private String lastArrestDate;

    // Constructors
    public Offender(int offenderId, String firstName, String lastName, int age, String gender,
int priorOffensesCount, String riskLevel, String lastArrestDate) {
        this.offenderId = offenderId;
        this.firstName = firstName;
        this.lastName = lastName;
        this.age = age;
        this.gender = gender;
        this.priorOffensesCount = priorOffensesCount;
        this.riskLevel = riskLevel;
        this.lastArrestDate = lastArrestDate;
```

```
}

// Getters and Setters
public int getOffenderId() {
    return offenderId;
}

public void setOffenderId(int offenderId) {
    this.offenderId = offenderId;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public int getAge() {
    return age;
}

public void setAge(int age) {
    this.age = age;
}

public String getGender() {
    return gender;
}

public void setGender(String gender) {
    this.gender = gender;
```

```
    }

    public int getPriorOffensesCount() {
        return priorOffensesCount;
    }

    public void setPriorOffensesCount(int priorOffensesCount) {
        this.priorOffensesCount = priorOffensesCount;
    }

    public String getRiskLevel() {
        return riskLevel;
    }

    public void setRiskLevel(String riskLevel) {
        this.riskLevel = riskLevel;
    }

    public String getLastArrestDate() {
        return lastArrestDate;
    }

    public void setLastArrestDate(String lastArrestDate) {
        this.lastArrestDate = lastArrestDate;
    }
}
```

3.15 Database JDBC Connection:

Databaseutil.java:

```
package com.crimedata.utils;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseUtil {

    private static final String DB_URL = "jdbc:mysql://127.0.0.1:3306/prj";  // Database URL
    private static final String DB_USER = "root";  // Database username
    private static final String DB_PASSWORD = "Miruthula@24";  // Database password
(change as needed)
```

```
    // Establish a database connection and return the connection object
    public static Connection getConnection() {
        try {
            // Load the MySQL JDBC driver
            Class.forName("com.mysql.cj.jdbc.Driver");

            // Attempt to connect to the database
            Connection conn = DriverManager.getConnection(DB_URL, DB_USER,
DB_PASSWORD);

            if (conn != null) {
                System.out.println("Database connected successfully!");
            } else {
                System.out.println("Failed to connect to the database.");
            }

            return conn;
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
            return null;
        }
    }

    // Close the connection (if needed in controllers)
    public static void closeConnection(Connection connection) {
        if (connection != null) {
            try {
                connection.close();
                System.out.println("Connection closed.");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Database Files:

3.16 Create_tables.sql:

```
CREATE DATABASE IF NOT EXISTS CrimeData;

USE CrimeData;

CREATE TABLE IF NOT EXISTS Crimes (
    crime_id INT AUTO_INCREMENT PRIMARY KEY,
    crime_type VARCHAR(255),
    crime_date DATE,
    crime_time TIME,
    crime_location VARCHAR(255),
    crime_description TEXT
);

CREATE TABLE IF NOT EXISTS Offenders (
    offender_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    age INT,
    gender VARCHAR(10),
    prior_offenses_count INT,
    risk_level VARCHAR(50),
    last_arrest_date DATE
);
CREATE TABLE IF NOT EXISTS Locations (
    location_id INT AUTO_INCREMENT PRIMARY KEY,
    crime_id INT,
    address VARCHAR(255),
    latitude DOUBLE,
    longitude DOUBLE,
    FOREIGN KEY (crime_id) REFERENCES Crimes(crime_id)
);

CREATE TABLE IF NOT EXISTS IncidentReports (
    report_id INT AUTO_INCREMENT PRIMARY KEY,
    crime_id INT,
    officer_name VARCHAR(255),
    case_status ENUM('Open', 'Closed', 'Under Investigation'),
    report_notes TEXT,
    report_date DATE,
```

```
    FOREIGN KEY (crime_id) REFERENCES Crimes(crime_id)
);

CREATE TABLE IF NOT EXISTS Witnesses (
    witness_id INT AUTO_INCREMENT PRIMARY KEY,
    crime_id INT,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    statement TEXT,
    contact_info VARCHAR(255),
    FOREIGN KEY (crime_id) REFERENCES Crimes(crime_id)
);

CREATE TABLE IF NOT EXISTS Crime_Offender (
    crime_id INT,
    offender_id INT,
    PRIMARY KEY (crime_id, offender_id),
    FOREIGN KEY (crime_id) REFERENCES Crimes(crime_id),
    FOREIGN KEY (offender_id) REFERENCES Offenders(offender_id)
);

CREATE TABLE IF NOT EXISTS OfficerActivity (
    activity_id INT AUTO_INCREMENT PRIMARY KEY,
    officer_name VARCHAR(255),
    report_id INT,
    activity_date DATE,
    activity_description TEXT,
    FOREIGN KEY (report_id) REFERENCES IncidentReports(report_id)
);
```

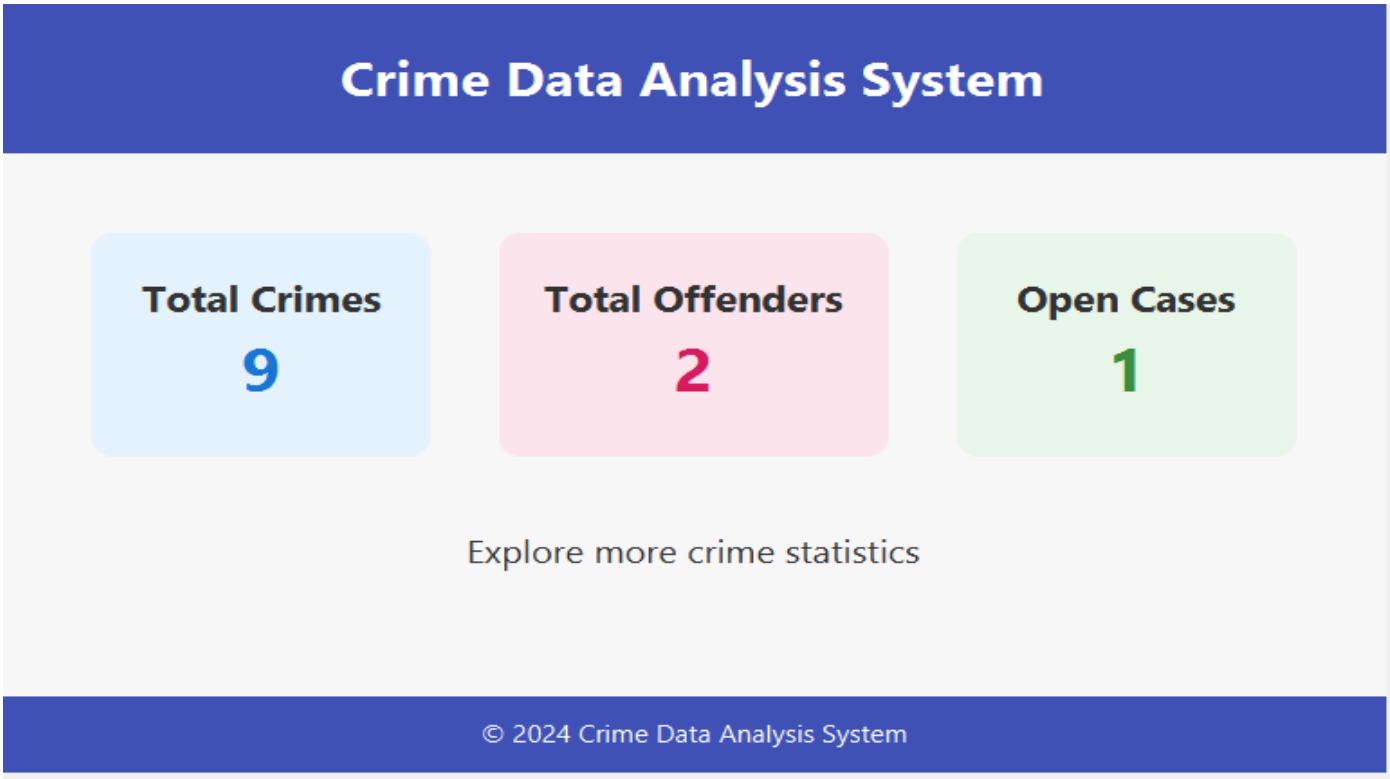
3.17 Main.java :

```
package com.crimedata;

import javafx.application.Application;
import javafx.scene.Scene;
import javafx.fxml.FXMLLoader;
import javafx.stage.Stage;

public class Main extends Application {
```

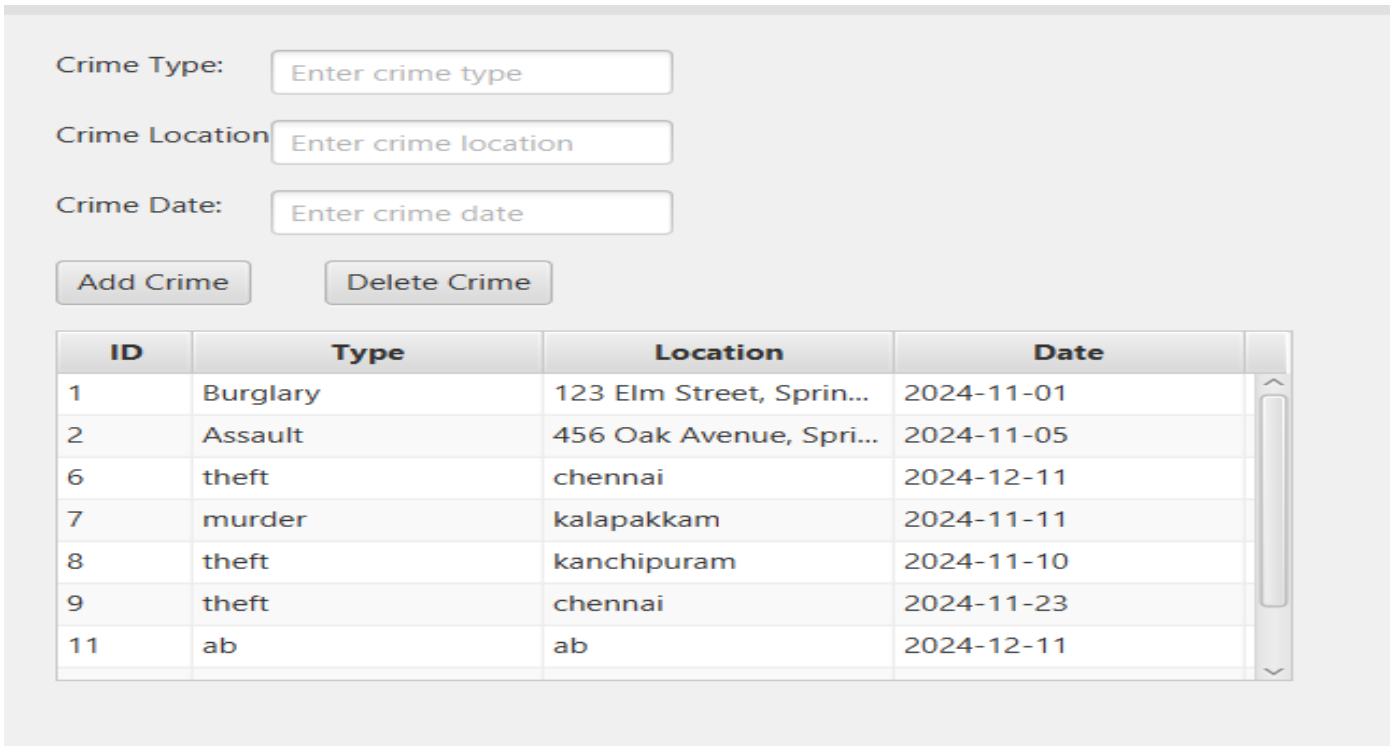
```
    @Override
    public void start(Stage primaryStage) throws Exception {
        // Load the Dashboard FXML as the main entry page
        FXMLLoader loader = new
FXMLLoader(getClass().getResource("/com/crimedata/views/Main.fxml"));
        Scene scene = new Scene(loader.load());
        primaryStage.setScene(scene);
        primaryStage.setTitle("Crime Data Analysis System");
        primaryStage.show();
    }

    public static void main(String[] args) {
        Application.launch(args);
    }
}
```

SNAPSHOTS

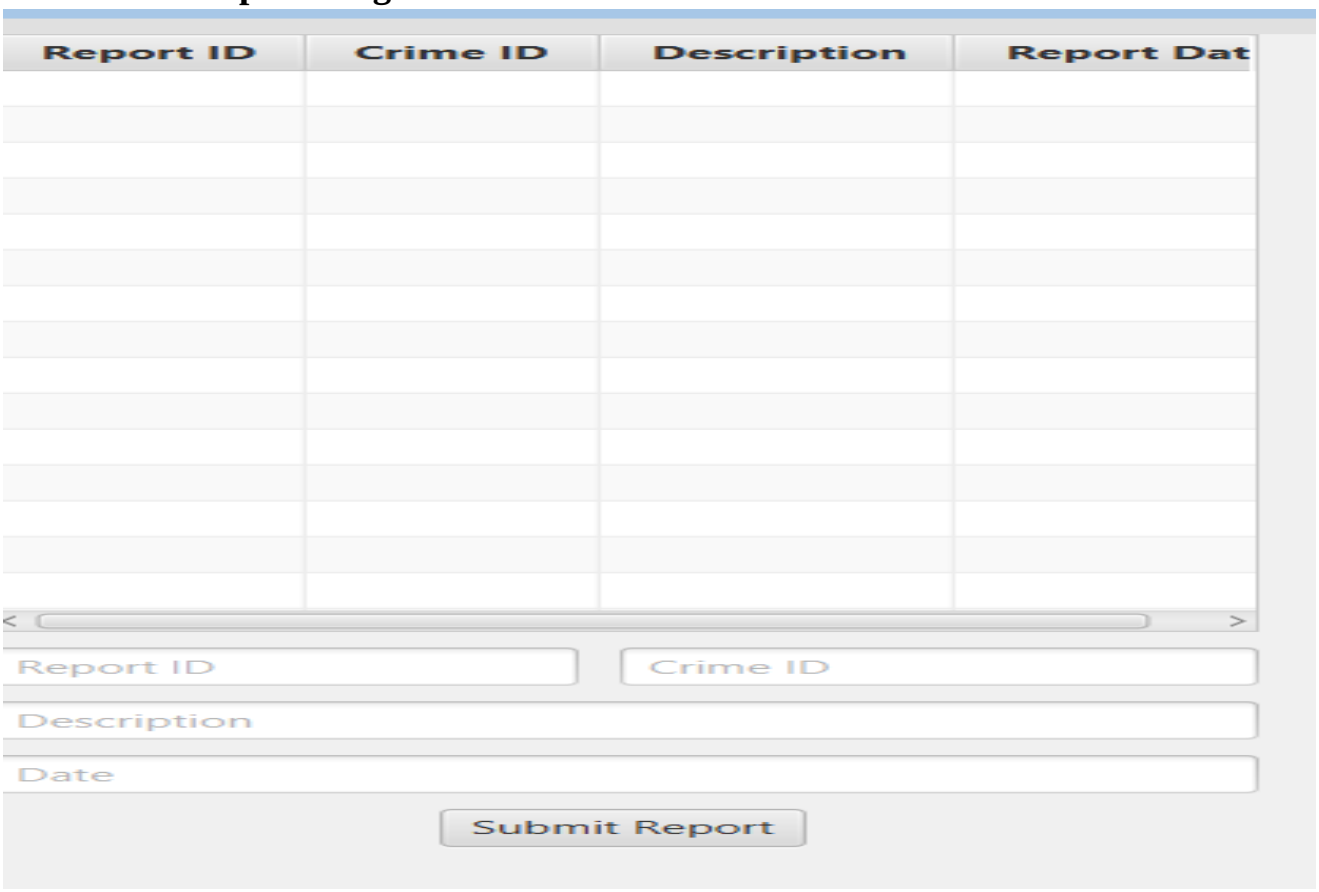
4.1 Dasboard:



4.2 Crime records page



4.3 Incident Reports Page



CONCLUSION

With the help of our project, law enforcement agencies, policymakers, and researchers will be able to effortlessly manage and analyze crime-related data. The **Crime Data Analysis System** provides a centralized platform to record crimes, maintain offender profiles, track incident reports, and monitor officer activities. By organizing all critical data and making it easily accessible through an interactive dashboard, the system saves significant time and resources. It also enhances decision-making capabilities by providing advanced analytics, including crime trends, geographic hotspots, and offender patterns.

REFERENCES

1. **https://www.javatpoint.com/java-tutorial**
2. https://www.javatpoint.com/javafx-tutorial
3. SQL | Codecademy
4. https://www.w3schools.com/MySQL/default.asp