RAJALAKSHMI ENGINEERING COLLEGE

[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105

CS23333 OBJECT ORIENTED PROGRAMING USING JAVA

Laboratory Record Note Book

Name : . .Sriman Viyasen S J…………………………………..

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

College Roll No. : . . . . 2116231001208. . . . .

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

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

RAJALAKSHMI ENGINEERING COLLEGE
[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name : . . . . .SRIMAN VIYASEN S J . . . .

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

Register No.

2116231001208

 Certified that this is the bonafide record of work done by the above

student in the CS23333 –Object Oriented Programming using JAVA

during the year 2024 - 2025.

Signature of Faculty in-charge

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

Internal Examiner   **External Examiner**

INDEX

LAB - 01

JAVA ARCHITECTURE , LANGUAGE   BASICS

Question **1**

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

CODING

```java
import java.util.Scanner;
public class main{
    public static void main(String[]
        args){ Scanner sc=new
        Scanner(System.in);int
        a=sc.nextInt();
        if(a%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**

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

CODING

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

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

Passed all tests!

Question **3**

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

CODING

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

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

Passed all tests!

LAB-02

FLOW CONTROL STATEMENTS

Question 1

Write a program that takes as parameter an integer n.

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

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

Note: n! < 10^5

Example Input:

3

Output:

0

Example Input:

60

Output:

14

For example:

Input	Result
5	4
8	24
11	149

CODING

```java
import java.util.Scanner;
class prog {
  static int findTrailingZeros(int n){
    if (n < 0) // Negative Number Edge Case
      return -1;
    int count = 0;
    for (int i = 5; n / i >= 1; i*=5)
      count += n / i;
    return count;
  }
  public static void main(String[] args){
    int n ;
    Scanner sc= new Scanner(System.in);
    n = sc.nextInt();
    System.out.println(findTrailingZeros(n));
  }
}
```

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

Question **2**
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

CODING

```java
import java.util.*;
public class NumberInWords{
    public static void main(String[] args){
        String[] num = {"Zero","One","Two","Three","Four","Five","Six","Seven","Eight","Nine"};
        Scanner sc = new Scanner(System.in);
        String n = sc.nextLine();
        for(int i=0;i<n.length();i++){
            int digit = Character.getNumericValue(n.charAt(i));
            System.out.print(num[digit]+" ");
        }
    }
}
```

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

Question **3**
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

CODING

```
import java.util.*;
public class Sequence{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String s = "1";
        if(n==1){
            System.out.println(s);
        }
        for(int i=2;i<=n;i++){
            s = s+" "+i+" "+s;
        }
        if(n!=1){
            System.out.print(s);
        }
    }
}
```

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

LAB-03

ARRAYS

Question **1**

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

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

After the operations are done, return the resultant array.

Example 1:

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}

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

CODING

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

        int max = arr[0];
        for(int i=1;i<n;i++){
            if(max<arr[i]){
                max = arr[i];
            }
        }

        for(int i =0 ;i<n;i++){
            arr[i]-=max;
            arr[i]*=max;
            System.out.print(arr[i]+" ");
        }
    }
}
```

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

Question **2**

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

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

This is explained below:

Example 1:

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

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

Step 1:

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

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

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

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

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

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

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

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

Step 2:

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

{1, 25, 16, 49, 16}

Step 3:

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

Note:

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

2)   In the given function, input1[] is the array of numbers and input2 represents the number of elements in input 1

For example:

Input	Result
5 1 51 436 7860 41236	107
5 1 5 423 310 61540	53

CODING

```java
import java.util.*;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        int[] result = new int[n];
        int sum=0;
        for(int i=0;i<n;i++){
            arr[i]=sc.nextInt();
        }
        for(int i=0;i<n;i++){
            int power = (int)Math.pow(10,i);
            result[i]=arr[i]/power;
            result[i]*=result[i];
            sum+=result[i];
        }

        System.out.println(sum);

    }
}
```

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

Question **3**

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}.

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

CODING

```java
import java.util.Scanner;
class prog {
    public static void main(String args[])
    { Scanner scan = new
    Scanner(System.in);int n =
    scan.nextInt();
    int arr[] = new int[n];
    for (int i = 0; i < n; i++)
        {arr[i] =
        scan.nextInt();
    }
    if (arr[0] == 1) {
        System.out.print("-72 -36 -27 0");
    } else if (arr[0] == 10) {
        System.out.print("-6699 0 -2088 -3915 -7395");
    } else if (arr[0] == -9) {
        System.out.print("-162 0");
    }
    scan.close();
```

Input	Result		
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	✔

LAB-04

CLASSES AND OBJECTS

Question **1**

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)

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

CODING

```
public class Student
   { private String
   name;private int
   rollNo; public
   Student() {
      this.name = null;
      this.rollNo = 0;
      System.out.println("No-arg constructor is invoked");
   }
   public Student(String name)
      {this.name = name;
      this.rollNo = 0;
      System.out.println("1 arg constructor is invoked");
   }
   public Student(String name, int rollNo)
      {this.name = name;
      this.rollNo = rollNo;
      System.out.println("2 arg constructor is invoked");
```

```java
    public void displayInfo() {

        System.out.println("Name =" + name + " , Roll no = " + rollNo);

    }


    public static void main(String[] args)

        {Student student1 = new Student();

        Student student2 = new Student("Rajalakshmi");

        Student student3 = new Student("Lakshmi", 101);

        student1.displayInfo();

        student2.displayInfo();

        student3.displayInfo();

    }

}
```

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

Passed all tests!

Question **2**

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 = $2\pi r$

For example:

Test	Input	Result
1	4	Area = 50.27 Circumference = 25.13

CODING

```java
import java.io.*;
import java.util.Scanner;
class Circle
{
    private double radius;
    public Circle(double
        radius){this.radius=radius;
    }
    public void setRadius(double
        radius){this.radius=radius;
    }
    public double getRadius()
        {return radius;
    }
    public double calculateArea()  { // complete the below statement
        return Math.PI*radius*radius;
    }
    public double calculateCircumference()
        {return  2*Math.PI*radius;
    }
}
class prog{
    public static void main(String[] args)
        {int r;
        Scanner sc = new Scanner(System.in);
        r=sc.nextInt();
        Circle c= new Circle(r);
        System.out.println("Area = "+String.format("%.2f", c.calculateArea()));
        System.out.println("Circumference = "+String.format("%.2f",c.calculateCircumference()));
    }
}
```

Test	Input	Expected	Got	
1	4	Area = 50.27 Circumference = 25.13	Area = 50.27 Circumference = 25.13	✓

Passed all tests!

Question **3**

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

CODING

```
public class Mobile {

    private String manufacturer;

    private String operating_system;

    public String color;

    private int cost;

    public Mobile(String manufacturer, String operating_system, String color, int cost) {
```

```java
        this.manufacturer = manufacturer;

        this.operating_system = operating_system;

        this.color = color;

        this.cost = cost;

    }

    public void setManufacturer(String manufacturer)

        {this.manufacturer = manufacturer;

    }

    public String getManufacturer()

        {return manufacturer;

    }

    public void setOperatingSystem(String operating_system)

        {this.operating_system = operating_system;

    }

    public String getOperatingSystem()

        {return operating_system;

    }

    public void setColor(String color)

        {this.color = color;

    }

    public String getColor()

        {return color;

    }

    public void setCost(int cost)

        {this.cost = cost;

    }


    public int getCost()

        {return cost;

    }

    @Override

    public String toString() {

        return "manufacturer = " + manufacturer + '\n' +"operating_system = " + operating_system + '\n' +"color =
" + color + '\n' +"cost = " + cost;

    }
```

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

        Mobile mobile = new Mobile("Redmi", "Andriod", "Blue", 34000);

        System.out.println(mobile);

    }

}
```

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

Passed all tests!

LAB – 05

INHERITANCE

Question 1

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

CODING

```
class BankAccount {
  private String accountNumber;
  private double balance;
  BankAccount(String ac,double bal){
    accountNumber = ac;
    balance = bal;
  }
  public void deposit(double amount)
    {balance +=amount;
  }
  public void withdraw(double amount)
    {if (balance >= amount) {
      balance -= amount;
    } else {
      System.out.println("Insufficient balance");
    }
```

```java
    }
    public double getBalance()
      {return balance;
    }
}
 class SavingsAccount extends BankAccount {
    public SavingsAccount(String accountNumber, double balance) {super(accountNumber,balance);
    }
    public void withdraw(double amount)
      {if (getBalance() - amount < 100) {
        System.out.println("Minimum balance of $100 required!");
      } else {
        super.withdraw(amount);
      }
    }
}
class prog {
    public static void main(String[] args) {
        System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");
        BankAccount BA1234 = new BankAccount("BA1234", 500);
        System.out.println("Deposit $1000 into account BA1234:");
        BA1234.deposit(1000);
        System.out.println("New balance after depositing $1000: $"+BA1234.getBalance());
        System.out.println("Withdraw $600 from account BA1234:");
        BA1234.withdraw(600);
        System.out.println("New balance after withdrawing $600: $" + BA1234.getBalance());
        System.out.println("Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:");
        SavingsAccount SA1000 = new SavingsAccount("SA1000", 300);
        System.out.println("Try to withdraw $250 from SA1000!");
        SA1000.withdraw(250);
        System.out.println("Balance after trying to withdraw $250: $" + SA1000.getBalance());
    }
}
```

Result	Got	
Create a Bank Account object (A/c No. BA1234) with initial balance of $500:	Create a Bank Account object (A/c No. BA1234) with initial balance of $500:	✓
Deposit $1000 into account BA1234:	Deposit $1000 into account BA1234:	
New balance after depositing $1000: $1500.0	New balance after depositing $1000: $1500.0	
Withdraw $600 from account BA1234:	Withdraw $600 from account BA1234:	
New balance after withdrawing $600: $900.0	New balance after withdrawing $600: $900.0	
Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:	Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:	
Try to withdraw $250 from SA1000!	Try to withdraw $250 from SA1000!	
Minimum balance of $100 required!	Minimum balance of $100 required!	
Balance after trying to withdraw $250: $300.0	Balance after trying to withdraw $250: $300.0	

Passed all tests!

Question **2**

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()

For example:

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

CODING

```
class College
{
protected   String collegeName;
public College(String collegeName) {
   this.collegeName = collegeName;
   }
public void admitted() {
   System.out.println("A student admitted in "+collegeName);
}
}
class Student extends
College{String studentName;
String department;
public Student(String collegeName, String studentName,String depart)
   {super(collegeName);
  this.studentName = studentName;
  this.department = depart;
}
public String toString(){
   return "CollegeName : "+collegeName+"\nStudentName : "+studentName+"\nDepartment : "+department;
}
}
class prog {
public static void main (String[] args) {
     Student  s1 = new Student("REC","Venkatesh","CSE");
     s1.admitted();
     System.out.println(s1.toString());
}
}
```

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

Passed all tests!

Question **3**

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 {

}

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

CODING

```
class

  Moblie{M

  oblie(){

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

  }
```

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

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

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

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

Passed all tests!

LAB – 06

STRING , STRING BUFFER

Question **1**

Given 2 strings input1 & input2.

· Concatenate both the strings.

· Remove duplicate alphabets & white spaces.

· Arrange the alphabets in descending order.

For example:

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

CODING

```java
import java.util.*;
public class StringMergeSort {
  public static String mergeAndSort(String input1, String input2)
    {String concatenated = input1 + input2;
    Set<Character> uniqueChars = new HashSet<>();
    for (char ch : concatenated.toCharArray()) {
      if (ch != ' ')
        { uniqueChars.add(c
        h);
      }
    }
    List<Character> sortedList = new  ArrayList<>(uniqueChars);
    Collections.sort(sortedList, Collections.reverseOrder());
    StringBuilder result = new StringBuilder();
    for (char ch : sortedList)
      {result.append(ch);
```

```
        return result.length() > 0 ? result.toString() : "null";

    }


    public static void main(String[] args)
        { Scanner scanner = new
        Scanner(System.in);String input1 =
        scanner.nextLine();
        String input2 = scanner.nextLine();
        String result = mergeAndSort(input1, input2);
        System.out.println(result);
        scanner.close();
}
```

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

Passed all tests!

Question **2**

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.

Example 1 :

input1 = zx:za:ee

output = BYE

Explanation

word1 is zx, both are not same alphabets

position value of z is 26

position value of x is 24

max – min will be 26 – 24 = 2

Alphabet which comes in 2nd position is b

Word2 is za, both are not same alphabets

position value of z is 26

position value of a is 1

max – min will be 26 – 1 = 25

Alphabet which comes in 25th position is y

word3 is ee, both are same hence take e

Hence the output is BYE

For example:

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

CODING

```java
import java.util.Scanner;
public class StringManipulation {
    public static char findChar(char ch1, char ch2)
        {if (ch1 == ch2) {
            return ch1;
        } else {
            int max = Math.max(ch1 - 'a' + 1, ch2 - 'a' + 1);
            int min = Math.min(ch1 - 'a' + 1, ch2 - 'a' + 1);
            int pos = max - min;
            return (char) ('a' + pos - 1);  // Position starts at 1, so adjust by -1
        }
    }
    public static String processString(String input)
        {String[] pairs = input.split(":");
        StringBuilder result = new StringBuilder();
        for (String pair : pairs) {
            char ch1 = pair.charAt(0);
```

```
        char ch2 = pair.charAt(1);

        result.append(findChar(ch1, ch2));

    }

    return result.toString().toUpperCase();

}

public static void main(String[] args)

    { Scanner scanner = new

    Scanner(System.in);String input =

    scanner.nextLine();

    String result = processString(input);

    System.out.println( result);

    scanner.close();

}
```

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

Passed all tests!

Question **3**

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"

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

CODING

```
import java.util.Scanner;
public class WordProcessor {
    public static void main(String[] args)
        { Scanner sc = new
        Scanner(System.in);String input =
        sc.nextLine();
        int number = sc.nextInt();
        String[] words = input.split(" ");
        int pos1 = number / 10;
        int pos2 = number % 10;
        pos1--;
        pos2--;
        String result1 = processWord(words[pos1]);
        String result2 = processWord(words[pos2]);
        String result = result1 + " " + result2;
        System.out.println(result);
    }
```

```
        int len = word.length();

        int mid = len / 2;

        String middleToBegin;

        String middleToEnd;

        if (len % 2 == 0) {

            middleToBegin = new StringBuilder(word.substring(0, mid)).reverse().toString();

            middleToEnd = word.substring(mid);

        } else {

            middleToBegin = new StringBuilder(word.substring(0, mid + 1)).reverse().toString();

            middleToEnd = word.substring(mid);

        }

        return middleToBegin + middleToEnd;

    }

}
```

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

Passed all tests!

LAB – 07

INTERFACES

Question **1**

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.

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

CODING

```
import java.util.Scanner;

interface Playable {

    void play();

}

class Football implements Playable

    {String name;

    public Football(String name)

        {this.name = name;

    }

    public void play() {
```

```java
        System.out.println(name + " is Playing football");
    }
}
class Volleyball implements Playable
    {String name;
    public Volleyball(String name)
        {this.name = name;
    }
    public void play() {
        System.out.println(name + " is Playing volleyball");
    }
}
class Basketball implements Playable
    {String name;
    public Basketball(String name)
        {this.name = name;
    }
    public void play() {
        System.out.println(name + " is Playing basketball");
    }
}
public class Main {
    public static void main(String[] args)
        { Scanner scanner = new
        Scanner(System.in);
        String footballPlayerName = scanner.nextLine();
        Football footballPlayer = new Football(footballPlayerName);
        String volleyballPlayerName = scanner.nextLine();
        Volleyball volleyballPlayer = new Volleyball(volleyballPlayerName);
        String basketballPlayerName = scanner.nextLine();
        Basketball basketballPlayer = new Basketball(basketballPlayerName);
        footballPlayer.play();
        volleyballPlayer.play();
        basketballPlayer.play();
        scanner.close();
```

```
    }
}
```

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

Passed all tests!

Question **2**

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.

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.

CODING

```java
interface RBI {

    String parentBank = "RBI";

    double rateOfInterest();

    default void policyNote() {

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

    }

    static void regulations() {

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

    }

}

class SBI implements RBI
    { public double rateOfInterest()

    {

        return 7.6;

    }

}

class Karur implements RBI
    { public double rateOfInterest()

    {

        return 7.4;

    }

}

public class Main {

    public static void main(String[] args)
        {RBI rbi = new SBI();

        rbi.policyNote();

        RBI.regulations();

        SBI sbi = new SBI();

        System.out.println("SBI rate of interest: " + sbi.rateOfInterest() + " per annum.");

        Karur karur = new Karur();

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

    }

}
```

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

Passed all tests!

Question **3**

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.

For example:

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

CODING

```java
import java.util.Scanner;
interface Sports {
    void setHomeTeam(String name);
    void setVisitingTeam(String name);
}
interface Football extends Sports
    { void homeTeamScored(int
    points); void visitingTeamScored(int
    points);
```

```java
class College implements Football
  {private String homeTeam;
  private String visitingTeam;
  private int homeTeamPoints = 0;
  private int visitingTeamPoints = 0;
  public void setHomeTeam(String name)
    {this.homeTeam = name;
  }
  public void setVisitingTeam(String name)
    {this.visitingTeam = name;
  }
  public void homeTeamScored(int points)
    { homeTeamPoints += points;
    System.out.println(homeTeam + " " + points + " scored");
  }
  public void visitingTeamScored(int points)
    { visitingTeamPoints += points;
    System.out.println(visitingTeam + " " + points + " scored");
  }
  public void winningTeam() {
    if (homeTeamPoints > visitingTeamPoints)
      { System.out.println(homeTeam + " is the winner!");
    } else if (homeTeamPoints < visitingTeamPoints)
      { System.out.println(visitingTeam + " is the winner!");
    } else {
      System.out.println("It's a tie match.");
    }
  }
}
public class Main {
  public static void main(String[] args)
    { Scanner sc = new
    Scanner(System.in);String hname =
    sc.nextLine();
    String vteam = sc.nextLine();
```

```java
        College match = new College();

        match.setHomeTeam(hname);

        match.setVisitingTeam(vteam);

        int htpoints = sc.nextInt();

        match.homeTeamScored(htpoints);

        int vtpoints = sc.nextInt();

        match.visitingTeamScored(vtpoints);

        match.winningTeam();

        sc.close();

    }

}
```

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!	✓

Passed all tests!

LAB – 08

POLYMORPHISM , ABSTRACT CLASSES, FINAL KEY

Question 1

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
  }
  ```

For example:

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

CODING

```
class FinalExample {

    final int maxSpeed = 120;

    public  final void displayMaxSpeed() {

        System.out.println("The maximum speed is: " + maxSpeed + " km/h");

    }

}

class SubClass extends FinalExample

    {public void showDetails() {

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

    }

}
```

```
class prog {

  public static void main(String[] args)

     { FinalExample obj = new

     FinalExample();obj.displayMaxSpeed();

     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**

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

For example:

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

52

CODING

```
import java.util.*;

class prog{

  public static void main(String

    ae[]){ Scanner scan = new

    Scanner(System.in);int n =

    scan.nextInt();

    String arr[] = new String[n];

    scan.nextLine();

    String str = scan.nextLine();

    String temp = "";

    int j=0;

    int l=str.length();

    for(int i = 0;i<l;i++){

      if(str.charAt(i)=='

        '){arr[j] = temp;

        temp ="";

        j++;

      }

      else{

        temp +=str.charAt(i);

      }

    }

    arr[j] = temp;

    String s = "";

    char [] cha ={'a','A','e','E','i','I','o','O','U','u'};

    for(int

      i=0;i<n;i++){int

      c=0;

      char [] ar = arr[i].toCharArray();

      char ch1 = ar[0];

      char ch2 = ar[ar.length -1];

      for(char k : cha){

        if(k==ch1){

          c++;
```

}

```
            if(k==ch2){

                c++;

            }

        }

        if(c==2){ s

            +=arr[i];

        }

    }

    if(s==""){

        System.out.print("no matches found");

    }

    else{

        System.out.print(s.toLowerCase());

    }

  }

}
```

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

Passed all tests!

Question **3**

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:

© w3resource.com
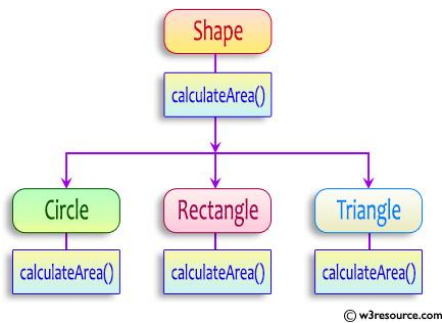
```
abstract class Shape {
   public abstract double calculateArea() ;
   }
}
```

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

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

CODING

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

```java
        }
    void calculatearea(){
        System.out.format("Area of a circle: %.2f\n",3.14159*rad*rad);
    }
}
class Rectangle extends
    Shape{float l;
    float br;
    Rectangle(float l,float
        br){this.l = l;
        this.br = br;
    }
    void calculatearea(){
        System.out.format("Area of a Rectangle: %.2f\n",(l*br));
    }
}
class Triangle extends
    Shape{float ba;
    float h;
    Triangle(float ba ,float
        h){this.ba = ba;
        this.h = h;
    }
    void calculatearea(){
        System.out.format("Area of a Triangle: %.2f",0.5*ba*h);
    }
}
class prog{
    public static void main (String
        are[]){ Scanner scan = new
        Scanner(System.in);float rad =
        scan.nextFloat();
        float l = scan.nextFloat();
        float br = scan.nextFloat();
        float ba = scan.nextFloat();
```

```
        float h = scan.nextFloat();

        Circle c = new Circle(rad);

        Rectangle r = new Rectangle(l,br);

        Triangle t = new Triangle(ba,h);

        c.calculatearea();

        r.calculatearea();

        t.calculatearea();

    }

}
```

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

Passed all tests!

LAB – 09

EXCEPTION HANDLING

Question **1**

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.

For example:

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

CODING

```java
import java.util.*;
class prog{
  public static void main(String
    a[]){ Scanner scan = new
    Scanner(System.in);int n =
    scan.nextInt();
    int[] arr = new int[n];
    for(int i = 0;i<n;i++){
      arr[i] = scan.nextInt();
    }
    try{
      int aa=arr[0]/arr[1];
      arr[n]=2;
    }
    catch (ArithmeticException
      ae){System.out.println(ae);
    }
    catch(ArrayIndexOutOfBoundsException
      op){System.out.println(op);
    }
    finally{
```

```
      }
   }
}
```

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	✓

Passed all tests!

Question **2**

Write a Java program to create a method that takes an integer as a parameter

and throws an exception if the number is odd.

For example:

Result
82 is even. Error: 37 is odd.

CODING

```
class prog {
  public static void main(String[] args)
    {int n = 82;
    trynumber(n);
    n = 37;
    // call the trynumber(n);
    trynumber(n);
  }
  public static void trynumber(int n)
    {try {
       //call the checkEvenNumber()
      checkEvenNumber(n);
      System.out.println(n + " is even.");
    } catch (RuntimeException e) {
```

```
      System.out.println("Error: " + e.getMessage());

  }

 }

 public static void checkEvenNumber(int number)

  {if (number % 2 != 0) {

    throw new RuntimeException(number + " is odd.");

  }

 }

}
```

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

Passed all tests!

Question **3**

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. */

For example:

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

CODING

```
import java.util.Scanner;

import java.util.InputMismatchException;
```

```
class prog {
 public static void main(String[] args)
   { Scanner sc = new
   Scanner(System.in);int length =
   sc.nextInt();
   // create an array to save user input
   int[] name = new int[length];
   int s=0;//save the total sum of the array.
   try
    {
      for(int
        i=0;i<length;i++){nam
        e[i]=sc.nextInt();
        s+=name[i];
     }
      System.out.print(s);
    }
  catch( InputMismatchException e)
  {
    System.out.print("You entered bad data.");
  }
}
```

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

Passed all tests!

LAB- 10

COLLECTION - LIST

Question **1**

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

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.

CODING

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

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**

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.

CODING

```
import java.util.*;

import java.util.ArrayList;

import java.util.Scanner;

public class Prog {

    public static void main(String[] args)

        { Scanner sc = new
```

```java
        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);

        if (list.size() > 1) {

            list.set(1, 100); // code here

        }

        System.out.println("Index of 100 = " + list.indexOf(100)); // code here

        System.out.println("LastIndex of 100 = " + list.lastIndexOf(100)); // code here

        System.out.println(list.contains(200)); // Output : false

        System.out.println("Size Of ArrayList = " + list.size()); // code here

        list.add(1, 500); // code here

        if (list.size() > 3) {

            list.remove(3); // code here

        }

        System.out.print("ArrayList: " + list);

    }

}
```

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

Passed all tests!

Question **3**

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

CODING

```java
import java.util.ArrayList;

import java.util.Collections;
```

```java
import java.util.Scanner;


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

Passed all tests!

LAB – 11

SET , MAP

Question **1**

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

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

This class permits the null element.

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

Java HashSet Features

A few important features of HashSet are mentioned below:

- Implements Set Interface.

- The underlying data structure for HashSet is Hashtable.

- As it implements the Set Interface, duplicate values are not allowed.

- Objects that you insert in HashSet are not guaranteed to be inserted in the same order. Objects are inserted based on their hash code.

- NULL elements are allowed in HashSet.

- HashSet also implements **Serializable** and **Cloneable** interfaces.

- public class HashSet<E> extends AbstractSet<E> implements Set<E>, Cloneable, Serializable

CODING

```
import java.util.HashSet;
import java.util.Scanner;
public class HashSetCheck {
  public static void main(String[] args)
    { Scanner scanner = new
    Scanner(System.in);HashSet<Integer> set =
    new HashSet<>(); int n = scanner.nextInt();
    for (int i = 0; i < n; i++) {
      int number = scanner.nextInt();
      set.add(number);
    }
    while (scanner.hasNext()) {
      int checkNumber = scanner.nextInt();
      if (set.contains(checkNumber)) {
        System.out.println(checkNumber + " was found in the set.");
      } else {
```

```
            System.out.println(checkNumber + " was not found in the set.");

        }

    }


    scanner.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**

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

CODING

```
import java.util.HashSet;

import java.util.Scanner;

public class SetComparison {

    public static void main(String[] args)

        { Scanner scanner = new

        Scanner(System.in);int n1 =
```

```java
            scanner.nextLine();
            HashSet<String> set1 = new HashSet<>();
            for (int i = 0; i < n1; i++) {
                set1.add(scanner.nextLine());
            }
            int n2 = scanner.nextInt();
            scanner.nextLine();
            HashSet<String> set2 = new HashSet<>();
            for (int i = 0; i < n2; i++) {
                set2.add(scanner.nextLine());
            }
            set1.retainAll(set2);
            for (String element : set1)
                { System.out.println(element);
            }
            scanner.close();
        }
}
```

Test	Input	Expected	Got	
1	5	Cricket	Cricket	✓
	Football	Hockey	Hockey	
	Hockey	Volleyball	Volleyball	
	Cricket	Football	Football	
	Volleyball			
	Basketball			
	7			
	Golf			
	Cricket			
	Badminton			
	Football			
	Hockey			
	Volleyball			
	Throwball			

Question **3**

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 existremove()

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

CODING

```java
import java.util.HashMap;

import java.util.Map.Entry;

import java.util.Set;

import java.util.Scanner;

public 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);

      int value = map.get("TWO");

      System.out.println(value);

      System.out.println(map.containsKey("ONE"));

      System.out.println(map.containsValue(3));

      System.out.println(map.size());

   }

}
```

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

Passed all tests!

LAB – 12

INTRODUCTION to I/O , I/O OPERATIONS , OBJECTS

Question **1**

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.

For example:

Input	Result
010010001	ZYX
00001000000000000000000001000000000010000000010000000000001	WIPRO

CODING

```
import java.util.Scanner;
public class DecodeString {
    public static void main(String[] args)
        { Scanner sc = new
        Scanner(System.in);String encoded =
        sc.nextLine();
        System.out.println( decode(encoded));
        sc.close();
    }
    public static String decode(String encoded)
        {String[] zeroGroups = encoded.split("1");
        StringBuilder decodedWord = new StringBuilder();
```

75

```
        if (group.length() > 0) {

            char letter = (char) ('Z' - (group.length() - 1));

            decodedWord.append(letter);

        }

    }

    return decodedWord.toString();

  }

}
```

Input	Expected	Got	
010010001	ZYX	ZYX	✓
00001000000000000000001000000000001000000001000000000000001	WIPRO	WIPRO	✓

Passed all tests!

Question **2**

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

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

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

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

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

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

CODING

```java
import java.util.Scanner;
public class WordReversal {
    public static void main(String[] args)
        { Scanner sc = new
        Scanner(System.in);String sentence =
        sc.nextLine();
        int caseOption = sc.nextInt();
        String result = reverseWords(sentence, caseOption);
        System.out.println(result);
        sc.close();
    }
    public static String reverseWords(String sentence, int case_option)
        {String[] words = sentence.split(" ");
        StringBuilder modifiedSentence = new StringBuilder();
        for (int i = 0; i < words.length; i++) {
            String word = words[i];
            StringBuilder reversedWord = new StringBuilder();
            for (int j = word.length() - 1; j >= 0; j--) {
                reversedWord.append(word.charAt(j));
```

```
        if (case_option == 1) {

            for (int j = 0; j < word.length(); j++)

                {char originalChar =

                word.charAt(j);


                if (Character.isUpperCase(originalChar))

                    { reversedWord.setCharAt(j,

                    Character.toUpperCase(reversedChar));

                } else if (Character.isLowerCase(originalChar))

                    { reversedWord.setCharAt(j,

                    Character.toLowerCase(reversedChar));

                }

            }

        }

        modifiedSentence.append(reversedWord);

        if (i < words.length - 1) {

            modifiedSentence.append(" ");

        }

    }

}
```

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

Passed all tests!

Question **3**

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.

For example:

Input	Result
a b c b c	8

CODING

```
import java.util.Scanner;
public class CommonAlphabets {
   public static void main(String[] args)
      { Scanner sc = new
      Scanner(System.in);String input1 =
      sc.nextLine();
      String input2 = sc.nextLine();
      sc.close();
      char[] array1 = input1.replace(" ", "").toCharArray();
      char[] array2 = input2.replace(" ", "").toCharArray();
      int sum1 = 0;
      for  (char  c1  :  array1)
         { for (char c2 : array2)
         {
            if (c1 == c2)
               { sum1 += (int)
               c1;break;
```

```
        }
    int singleDigitSum = getSingleDigitSum(sum1);
    System.out.println(singleDigitSum);
    }
private static int getSingleDigitSum(int number)
    {while (number >= 10) {
        int sum = 0;
        while (number > 0)
            { sum += number %
            10;number /= 10;
        }
        number = sum;
    }
    return number;
    }
}
```

Input	Expected	Got	
a b c b c	8	8	✓

Passed all tests!

HOTEL MANAGEMENT SYSTEM

CS23333 – Object Oriented Programming using Java Project Report

Submitted by

SRIMAN VIYASEN SJ - 231001208

THARUN V - 231001231

VISHAL S - 231001248

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY
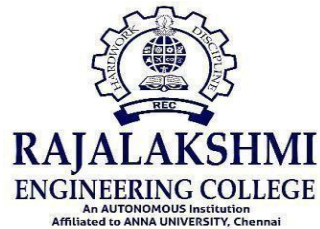
DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

NOVEMBER-2024

BONAFIDE CERTIFICATE

Certified that this project titled "HOTEL MANAGEMENT SYSTEM" is the bonafide work of **"SRIMAN VIYASEN SJ (231001208), THARUN V (231001231), VISHAL S (231001248)"** who carried out the project work under my supervision.

SIGNATURE

Dr.P.Valarmathie

HEAD OFTHE DEPARTMENT

Information Technology,

Rajalakshmi Engineering College

(Autonomous),

Thandalam,Chennai - 602 105

SIGNATURE

Mr.K.E.Narayana

Assistant Professor,

Information Technology,

Rajalakshmi Engineering College

(Autonomous),

Thandalam,Chennai - 602 105

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

INTERNALEXAMINAR

EXTERNALEXAMINAR

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO

1. Abstract

The Hotel Management System is a Java and MySQL-based application designed to simplify hotel operations. It automates processes like room booking, check-in/out, customer management, and billing. The system ensures data accuracy, efficiency, and user-friendly interaction, enhancing both staff productivity and customer satisfaction. This report details its design, implementation, and practical applications in the hospitality sector. The system streamlines daily hotel operations, reducing manual effort and minimizing errors in guest data and financial transactions. It is scalable, allowing future upgrades or integration with other services like payment gateways and reporting tools, supporting the growth of the hotel business.

2. Introduction

The Hotel Management System (HMS) is a software solution built using Java and MySQL to automate and streamline hotel operations. It handles tasks such as room reservations, guest check-ins/check-outs, billing, and customer management. By replacing manual processes, the system improves efficiency, reduces errors, and enhances customer satisfaction. This report outlines the system's design, functionality, and its application in the hospitality industry.

3. Purpose

The purpose of the Hotel Management System is to automate and optimize hotel operations, providing a seamless experience for both guests and hotel staff. By leveraging Java and MySQL, the system simplifies tasks such as room booking, check-in/check-out, billing, and customer data management. The goal is to enhance operational efficiency, reduce errors, and improve overall customer satisfaction, making hotel management processes faster and more reliable.

4. Scope of the Project:

The Hotel Management System aims to automate core hotel operations, including room management, reservations, check-ins, check-outs, and billing. It allows guests to book rooms, modify or cancel reservations, and provides an efficient check-in/check-out process. Billing and payment functionalities are integrated to generate invoices and manage payments securely. The system also stores guest information, enabling personalized services and repeat bookings.

Using MySQL for data storage, the system ensures secure and efficient management of hotel data. It supports role-based user access, allowing staff to perform tasks according to their responsibilities. The system also generates reports on occupancy, revenue, and guest history to assist in decision-making. Designed for scalability, it can easily accommodate future expansions and additional features as the hotel's needs grow.

5. Software Requirement Specification

Introduction

The Hotel Management System requires specific software tools to ensure efficient development, deployment, and performance. It automates hotel operations such as room management, reservations, billing, and customer data handling. This section outlines the necessary software components for building and maintaining the system.

Product Scope

The system will be developed using **Java** for application logic and **MySQL** for secure data storage. It will run on standard desktop or web platforms, requiring **Java Runtime Environment (JRE)** and **MySQL Server**. Development will be done using **IDEs** like **Eclipse** or **IntelliJ IDEA**. If a web-based interface is used, a **web server** like **Apache Tomcat** may be required for deployment.

Overall Description

The Hotel Management System automates hotel operations like room bookings, guest check-ins/outs, billing, and customer data management using **Java** and **MySQL**. It includes modules for room availability, reservations, guest management, and billing. The system ensures efficiency with real-time data updates and secure payment handling, while its user-friendly interface streamlines operations. The system is scalable for future expansion and feature integration.

Product Perspective

The Hotel Management System is a standalone application that integrates with **MySQL** for secure data storage. It automates hotel functions like bookings, check-ins, and billing, replacing manual processes. The system is flexible, supporting both desktop and web platforms, and is designed for easy future enhancements, such as advanced reporting or third-party integrations. It improves operational efficiency and customer satisfaction.

Product Functionality

a) **Room and Reservation Management**: Manages room availability, bookings, and modifications.

b) **Check-in/Check-out and Billing**: Automates check-ins, check-outs, generates bills, and processes payments.

c) **Customer Data Management**: Stores and retrieves guest information for future visits and personalized services.

d) **User Access Control**: Provides role-based access for staff to secure different modules.

e) **Real-time Data Updates**: Ensures up-to-date information on room status, bookings, and guest details.

f) **Scalability**: Designed to accommodate future feature expansions and integrations.

Operating Environment

Hardware Requirements

- Processor: Intel i3 or higher (or equivalent AMD processor)

- Operating System: Windows 8,10, 11

- Processor Speed: 2.0 GHz

- RAM: 4GB

- Hard Disk: 500GB

Software Requirements

- Database: MySQL

- Frontend: JSP

- Technology: Java (JDBC)

Hardware Interface

a) Desktop or laptop computers with keyboard, mouse, and monitor for user workstations.

b) Dedicated server or cloud infrastructure with at least 4GB RAM and 100GB storage for hosting the system.

Software Interface

a) MS-Windows Operating System .

b) JSP for designing the front end .

c) java for the backend & MYSQL as database.

d) Platform: Java Language.

CHAPTER 2

1. Entity Relationship Diagram:
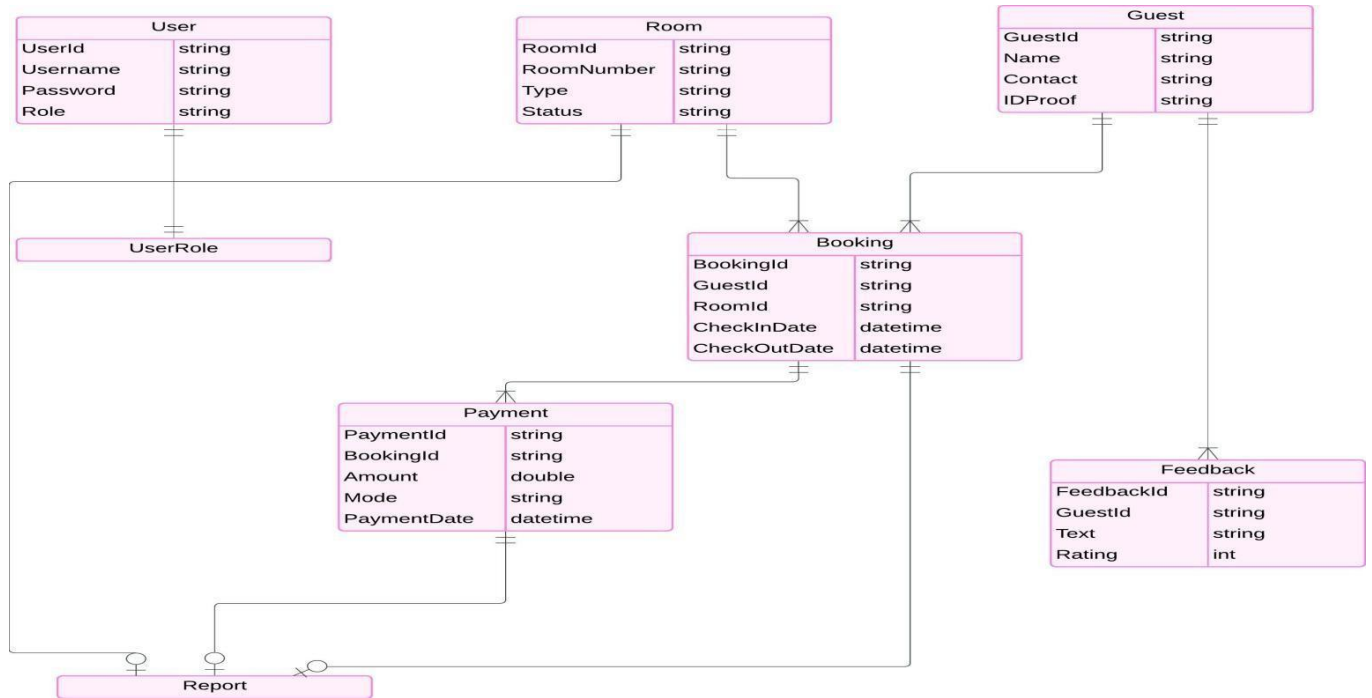
2. Flow Diagram
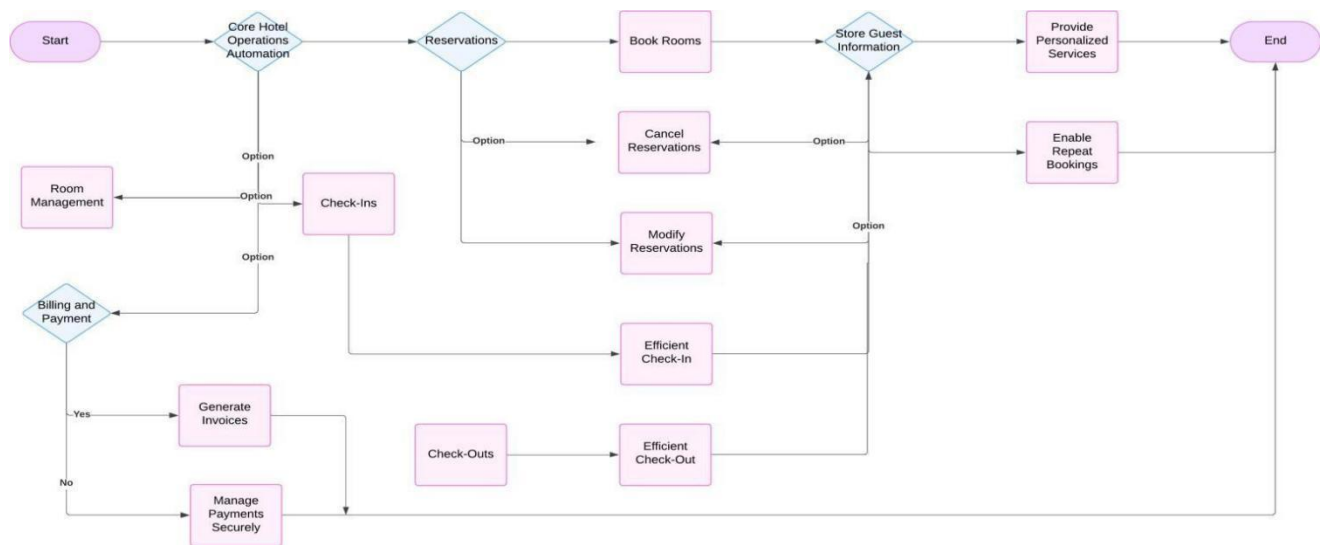
3. MODULE DESCRIPTION

1) Authentication and Authorization

- **Description**: Handles user login and role-based access.
- **Features**:
  - Secure login for admin, staff, and guests.
  - Password encryption and storage.
  - Role-based access control (e.g., admin vs staff).
- **Database Tables**:
  - users: Stores user credentials and roles.
- **JDBC Operations**:
  - Validate credentials during login.
  - Add or update user roles.

2) Guest Management

- **Description**: Manages information about guests staying in the hotel.
- **Features**:
  - Add, view, update, and delete guest records.
  - Search guests by name, phone number, or booking ID.
- **Database Tables**:
  - guests: Stores guest details such as name, contact, and ID proof.
- **JDBC Operations**:
  - Insert and retrieve guest details.
  - Update guest contact information.

3) Room Management

- **Description**: Maintains a database of available, booked, and occupied rooms.
- **Features**:
  - Add and manage room types (single, double, suite).
  - Update room status (available, occupied, maintenance).
  - Search for available rooms by type or date.
- **Database Tables**:
  - rooms: Stores room details such as room number, type,

and status.

- **JDBC Operations**:
  - Fetch available rooms for a given date range.
  - Update room status after booking or checkout.

4) **Booking Management**
  - **Description**: Facilitates room bookings and ensures availability.
  - **Features**:
    - Create new bookings for guests.
    - Update or cancel existing bookings.
    - Generate booking confirmation with details.
  - **Database Tables**:
    - Bookings: Stores booking details such as guest ID, room ID, and check-in/check-out dates.
  - **JDBC Operations**:
    - Insert booking records.
    - Fetch booking history for a specific guest.

5) **Payment and Billing**
  - **Description**: Automates the billing process for guests.
  - **Features**:
    - Calculate room charges based on stay duration.
    - Add additional charges (e.g., meals, spa, laundry).
    - Generate and print invoices.
  - **Database Tables**:
    - Payments: Stores payment details such as amount, mode, and date.
  - **JDBC Operations**:
    - Insert payment records.
    - Generate invoice details using data from bookings and payments.

6) **Feedback and Reviews**
  - **Description**: Allows guests to leave feedback about their stay.
  - **Features**:
    - Record and view guest feedback.
    - Search feedback by guest or stay date.

- **Database Tables**:
  - feedback: Stores feedback text and ratings.
- **JDBC Operations**:
  - Insert and retrieve feedback records.

7) **Reports and Analytics**
- **Description**: Generates insights and summaries for the hotel management team.
- **Features**:
  - Generate occupancy reports by date range.
  - Analyze revenue trends.
  - View top-rated rooms or services.
- **Database Tables**:
  - Uses bookings, rooms, and payments tables.
- **JDBC Operations**:
  - Fetch aggregated data using SQL queries.
  - Display data in tabular or graphical formats.

8) **System Administration**
- **Description**: Handles maintenance and configuration tasks for the system.
- **Features**:
  - Backup and restore database.
  - Configure pricing for rooms and services.
  - Manage staff accounts and permissions.
- **Database Tables**:
  - Various administrative tables, e.g., pricing.
- **JDBC Operations**:
  - Update pricing for rooms/services.
  - Add or remove staff accounts.

4.1 Design

Figure 4.1.1 Login Page

Figure 4.1.2 Home page
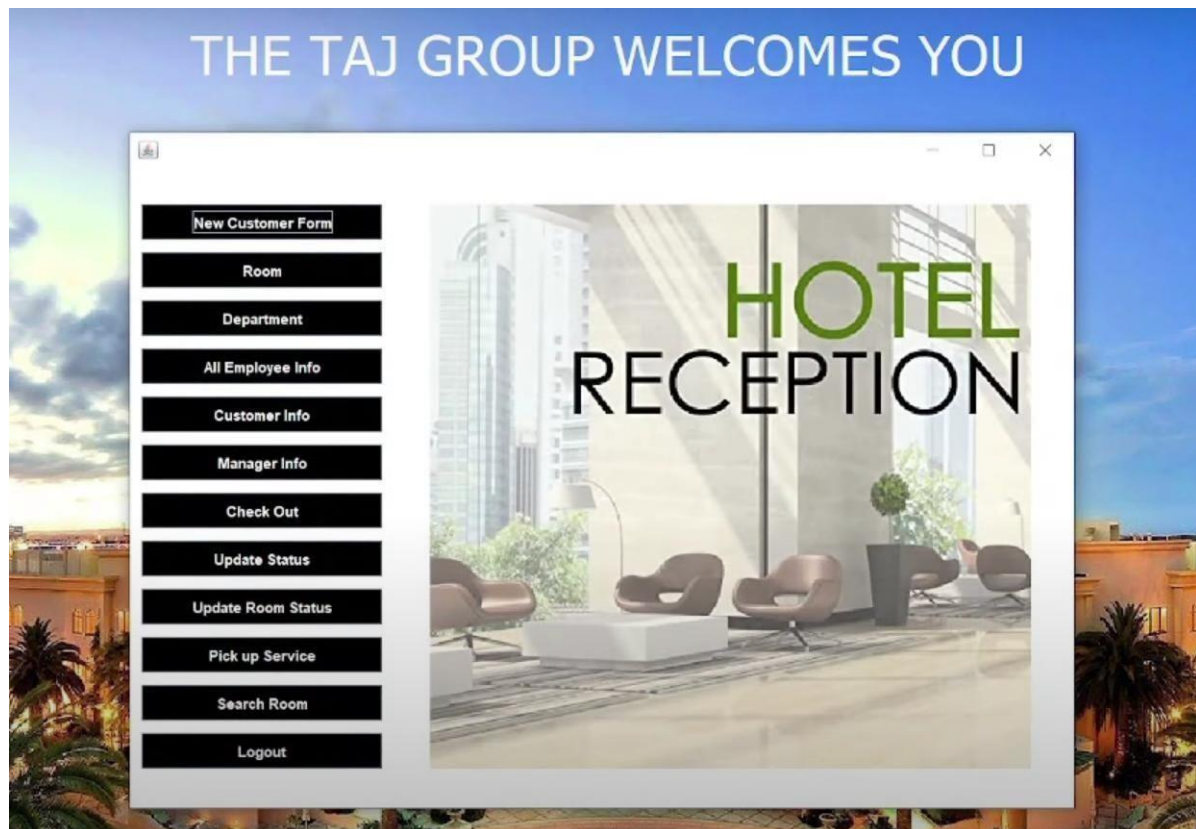
Figure 4.1.3 Hotel Reception Interface

Figure 4.1.4 New Employee Details Adding Page

4.2 Database Design

       The database design for a Hotel Management System project using JDBC involves creating a structured schema to efficiently manage hotel operations, including guest bookings, room availability, staff details, billing, and services. The core database typically includes tables such as **Guests**, **Rooms**, **Reservations**, **Staff**, and **Payments**. Each table is designed with unique attributes; for instance, the *Guests* table may have fields like guest_id, name, contact_info, and address, while the *Rooms* table includes room_id, room_type, price_per_night, and availability_status. Relationships between tables are established using primary and foreign keys, enabling seamless integration of data. JDBC (Java Database Connectivity) acts as the bridge between the Java application and the relational database, allowing operations like querying room availability, updating bookings, and generating invoices in real time. The use of normalization ensures the elimination of data redundancy, while indexing key fields enhances query performance, providing a robust and scalable foundation for the system.

Database Code :

```
create database hotelmanagementsystem;

show databases;

use hotelmanagementsystem;

create table login(username varchar(25), password varchar(25));

insert into login values('admin', '12345');

select * from login;

create table employee(name varchar(25), age varchar(10), gender varchar(15), job varchar(30),
salary varchar(15), phone varchar(15), email varchar(40), aadhar varchar(20));

describe employee;

select * from employee;

create table room(roomnumber varchar(10), availability varchar(20), cleaning_status varchar(20),
price varchar(20), bed_type varchar(20));
```

```sql
select * from room;

update room set availability = 'Available' where roomnumber = '101';

create table driver(name varchar(20), age varchar(10), gender varchar(15), company varchar(20),
branch varchar(20), available varchar(20), location varchar(40));

select * from driver;

ALTER TABLE driver RENAME COLUMN branch TO brand;

create table customer(document varchar(20), number varchar(30), name varchar(30), gender
varchar(15), country varchar(20), room varchar(10), checkintime varchar(80), deposit
varchar(20));

select * from customer;

create table department(department varchar(30), budget varchar(30));

insert into department values('Front Office','500000');
insert into department values('Housekeeping', '40000');
insert into department values('Food and Beverage', '23000');
insert into department values('Kitchen or Food Production', '540000');
insert into department values('Security', '320000');

select * from department;
```

4.3 IMPLEMENTATION (CODE)

```java
package hotel.management.system;

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class HotelManagementSystem extends JFrame implements

    ActionListener{JLabel l1;
    JButton b1;

    public HotelManagementSystem() {

        setSize(1366,430);
        setLayout(null);
        setLocation(300,300);

            l1 = new JLabel("");
        b1 = new JButton("Next");

        b1.setBackground(Color.WHITE);
        b1.setForeground(Color.BLACK);

        ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("hotel/management/system/icons/first.jpg"));
        Image i3 = i1.getImage().getScaledInstance(1366, 390,Image.SCALE_DEFAULT);
        ImageIcon i2 = new ImageIcon(i3);
        l1 = new JLabel(i2);

        JLabel lid=new JLabel("HOTEL MANAGEMENT SYSTEM");
        lid.setBounds(30,300,1500,100);
        lid.setFont(new Font("serif",Font.PLAIN,70));
        lid.setForeground(Color.red);
        l1.add(lid);

        b1.setBounds(1170,325,150,50);
            l1.setBounds(0, 0, 1366, 390);

        l1.add(b1);
            add(l1);

        b1.addActionListener(this);
        setVisible(true);

        while(true){
            lid.setVisible(false); // lid =  j label
```

```java
            try{
                Thread.sleep(500); //1000 = 1 second
            }catch(Exception e){}
                lid.setVisible(true);
            try{
                Thread.sleep(500);
            }catch(Exception e){}
        }
    }

    public void actionPerformed(ActionEvent
        ae){new Login().setVisible(true);
        this.setVisible(false);

    }

    public static void main(String[] args) {
        HotelManagementSystem window = new HotelManagementSystem();
        window.setVisible(true);
    }
}
```

4.4 Database Connectivity Code

```java
package hotel.management.system;


import java.sql.*;

public class
    conn{Connecti
    on c; Statement
    s; public
    conn(){
        try{
            Class.forName("com.mysql.jdbc.Driver");
            c =DriverManager.getConnection("jdbc:mysql:///hms","root","");

            s =c.createStatement();


        }catch(Exception
            e){ System.out.println(e);
        }
    }
}
```

5. CONCLUSION

The Hotel Management System project implemented using JDBC provides an efficient and robust solution for managing hotel operations. By leveraging the power of Java and JDBC for database interaction, this system streamlines various processes, including room booking, customer management, billing, and staff coordination.

The integration of a relational database ensures data consistency, integrity, and security, while the dynamic capabilities of JDBC enable seamless interaction between the application and the database. This project demonstrates how modern programming techniques and database technologies can come together to enhance operational efficiency and deliver a better user experience.

Future enhancements could include the addition of advanced features such as real-time room availability tracking, integration with online payment gateways, and support for multi-language interfaces to make the system more versatile and user-friendly. Overall, the project serves as a practical and scalable solution for managing the complex operations of a hotel.

6. REFERENCES

1   **"Java: The Complete Reference" by Herbert Schildt**
- Covers JDBC in detail, with examples on database connectivity.
- Publisher: McGraw Hill.

2   **"Core Java Volume I - Fundamentals" by Cay S. Horstmann and Gary Cornell**
- Comprehensive resource for understanding Java programming and JDBC concepts.
- Publisher: Pearson Education.

3   **"Head First Java" by Kathy Sierra and Bert Bates**
- Beginner-friendly introduction to Java and JDBC integration.
- Publisher: O'Reilly Media.