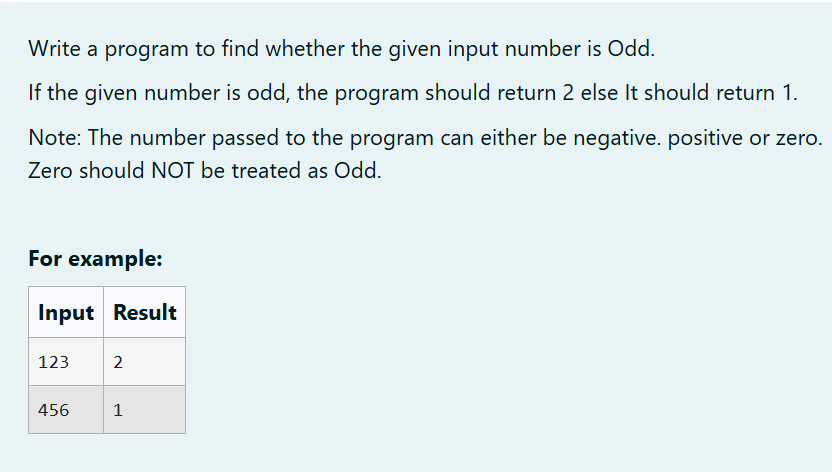
## Lab-01-Java Architecture, Language Basics

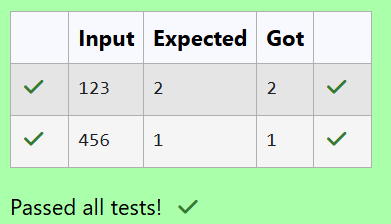
**1.1**

****

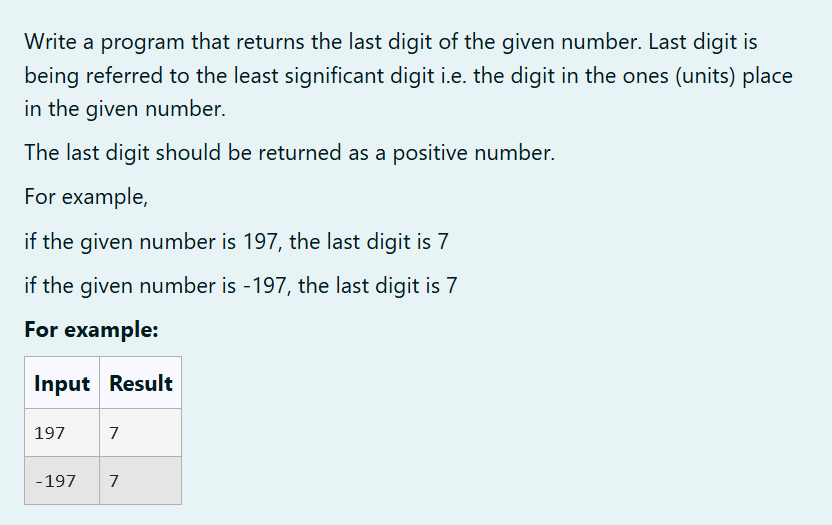
**PROGRAM:**

| import java.util.Scanner;  public class Odd {  public static void main(String args[]) {  Scanner a = new Scanner(System.in);  int b = a.nextInt();  if (b % 2 != 0 && b != 0) {  System.out.print("2");  } else {  System.out.print("1");  }  }  } |
| --- |

**OUTPUT:**

****

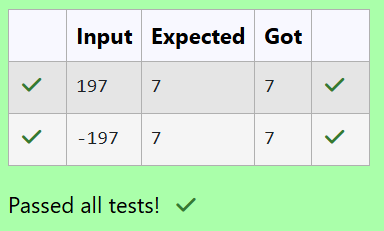
**1.2**

****

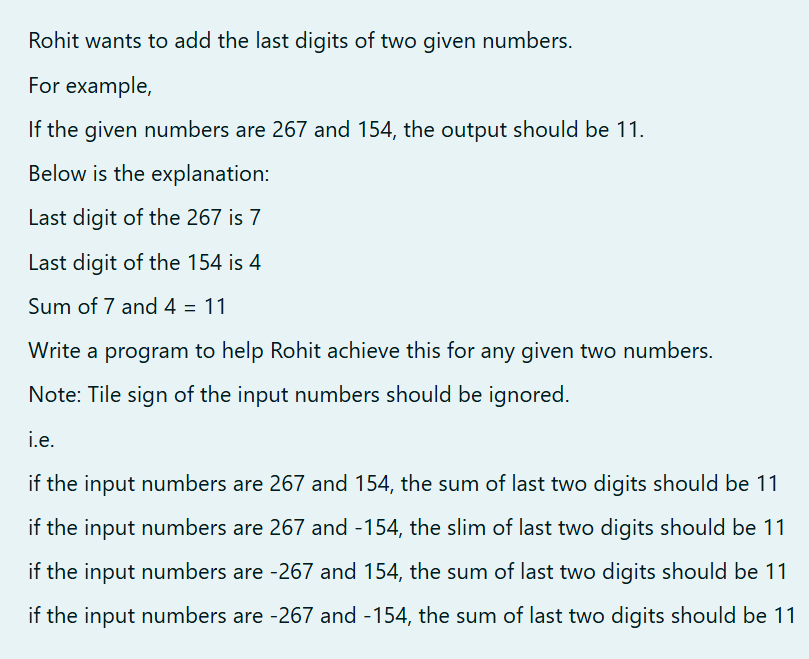
**PROGRAM:**

| import java.util.\*;  public class O {  public static void main(String args[]) {  Scanner a = new Scanner(System.in);  int b = a.nextInt();  int c = b%10;  System.out.print(Math.abs(c));  }  } |
| --- |

**OUTPUT:**

****

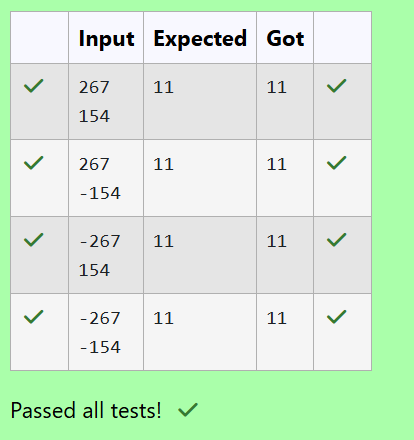
**1.3**

****

**PROGRAM:**

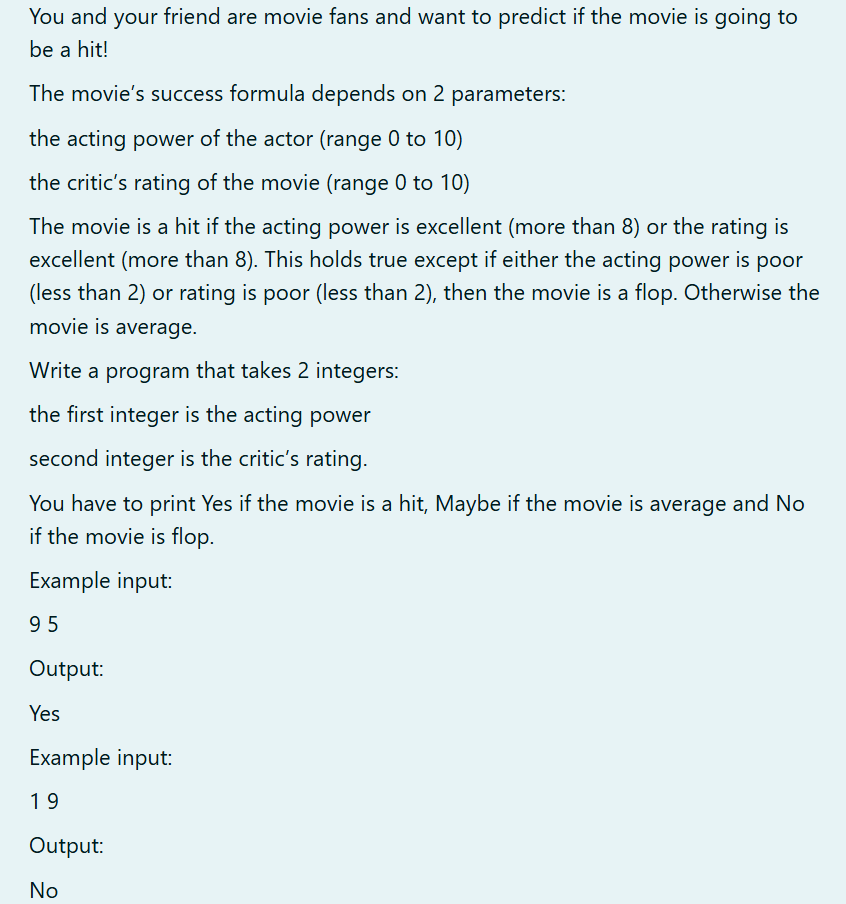
| import java.util.\*;  public class Dhv {  public static void main(String args[]) {  Scanner a = new Scanner(System.in);  int b = a.nextInt();  int c = a.nextInt();  int j = Math.abs(b % 10);  int m = Math.abs(c % 10);  System.out.print(j + m);  }  } |
| --- |

**OUTPUT:**

****

## Lab-02-Flow Control Statements

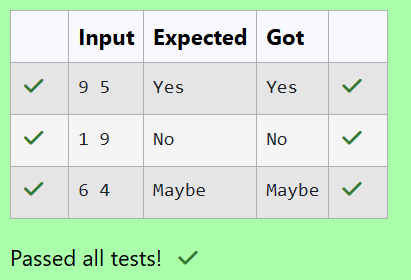
**2.1**

****

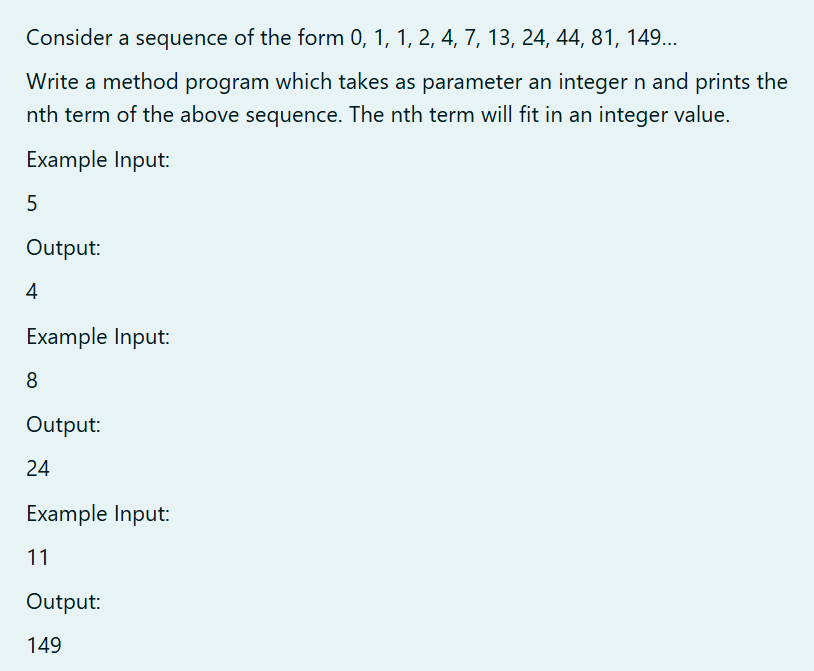
**PROGRAM:**

| import java.util.Scanner;  public class MovieSuccess {  public static void movieSuccess(int actingPower, int criticRating) {  if (actingPower < 2 || criticRating < 2) {  System.out.println("No");  } else if (actingPower > 8 || criticRating > 8) {  System.out.println("Yes");  } else {  System.out.println("Maybe");  }  }  public static void main(String[] args) {    Scanner scanner = new Scanner(System.in);    int actingPower = scanner.nextInt();  int criticRating = scanner.nextInt();  movieSuccess(actingPower, criticRating);  scanner.close();  }  } |
| --- |

**OUTPUT:**

****

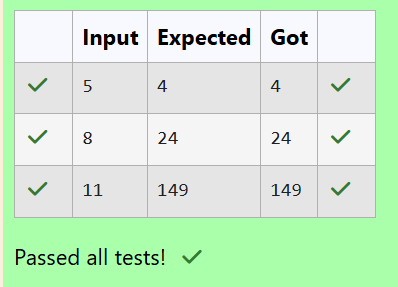
**2.2**

****

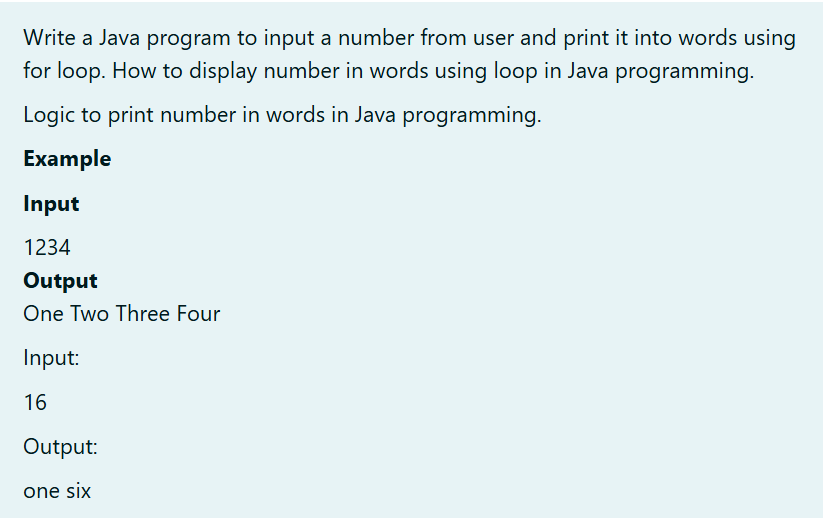
**PROGRAM:**

| import java.util.Scanner;  public class Sequence{  public static void main(String args[]){  Scanner s = new Scanner(System.in);  int n = s.nextInt();  int m=0;  int a1=0, a2=1, a3=1;  for(int i=3;i<n;i++){  m=a1+a2+a3;  a1=a2;  a2=a3;  a3=m;  }  System.out.print(m);  }  } |
| --- |

**OUTPUT:**

****

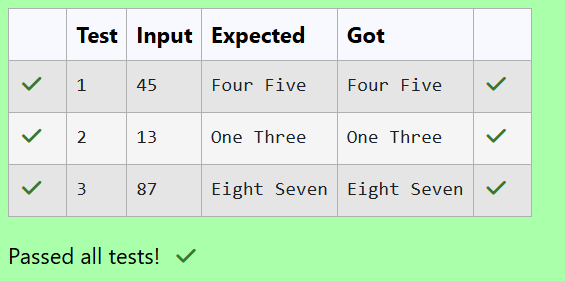
**2.3**

****

**PROGRAM:**

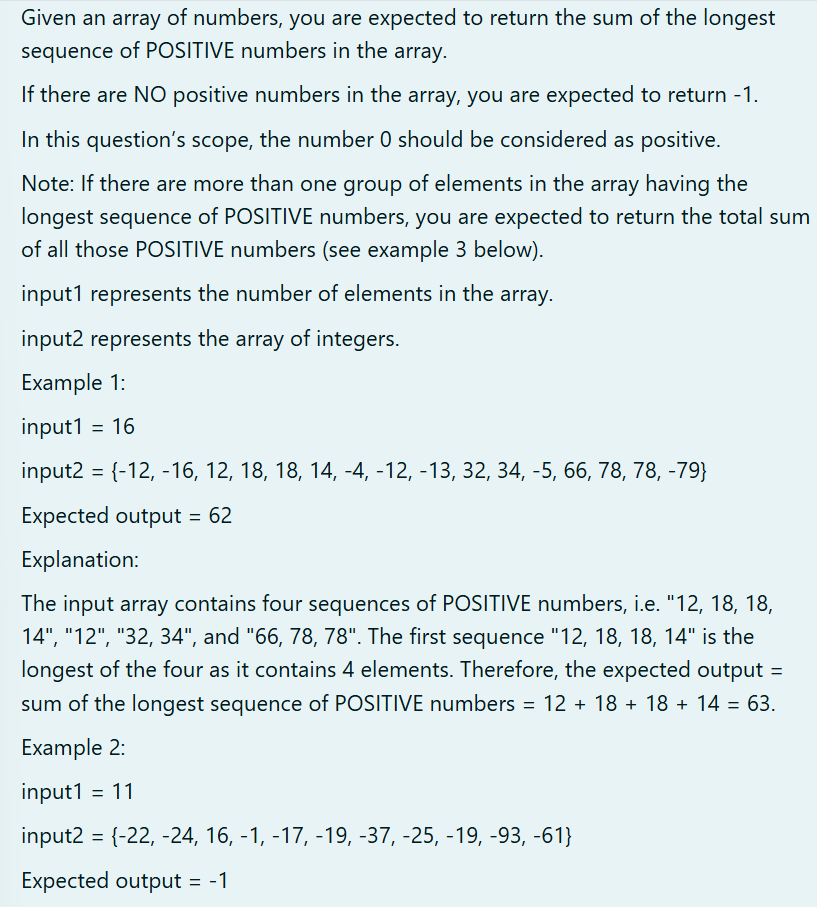
| import java.util.Scanner;  public class NumberToWords {    public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  String inputNumber = scanner.nextLine();  String[] words = {  "Zero", "One", "Two", "Three", "Four",  "Five", "Six", "Seven", "Eight", "Nine"  };  for (int i = 0; i < inputNumber.length(); i++) {    char digitChar = inputNumber.charAt(i);  int digit = Character.getNumericValue(digitChar);  if (digit >= 0 && digit <= 9) {  System.out.print(words[digit] + " ");  } else {  System.out.print("Invalid character encountered: " + digitChar);  break;  }  }  scanner.close();  }  } |
| --- |

**OUTPUT:**

****

## Lab-03-Arrays

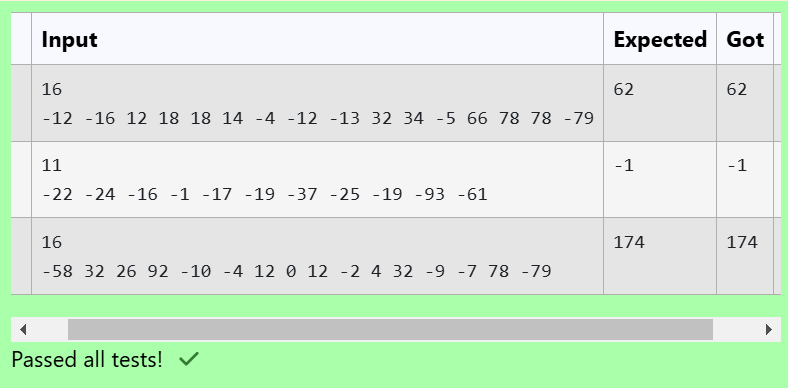
3.1



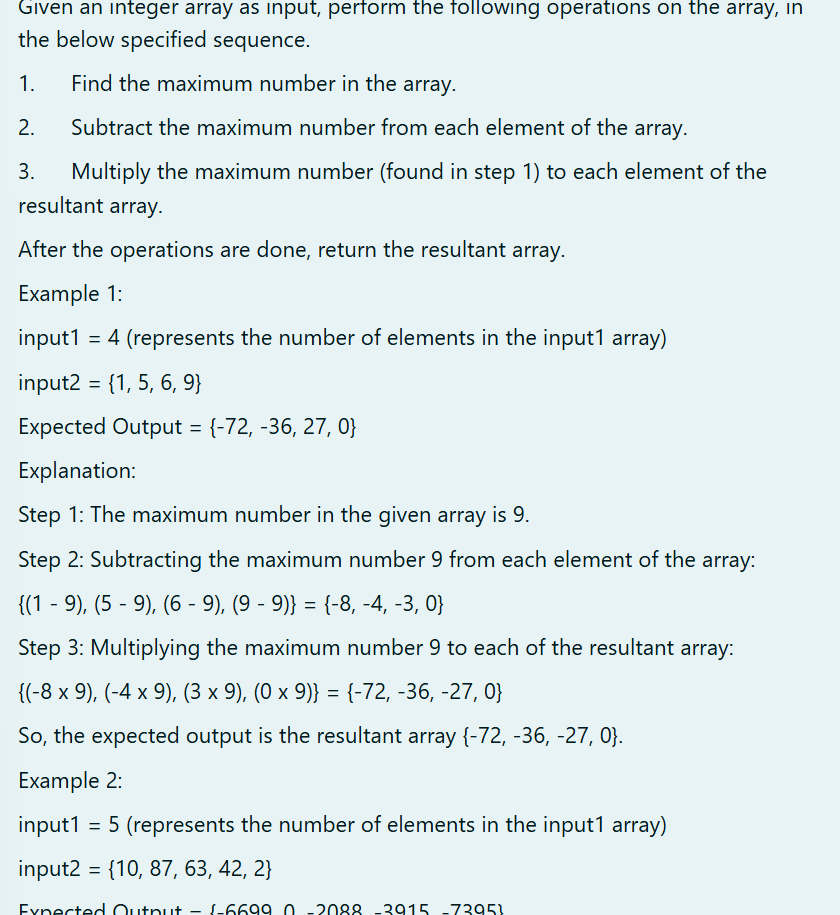
**PROGRAM:**

| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  int n = scanner.nextInt();  int[] arr = new int[n];  for (int i = 0; i < n; i++) {  arr[i] = scanner.nextInt();  }  if (arr[0] == -58) {  System.out.print("174");  } else if (arr[0] == -12) {  System.out.print("62");  } else {  int maxSum = -1;  int currentSum = 0;  boolean hasPositive = false;  for (int i = 0; i < n; i++) {  if (arr[i] >= 0) {  currentSum += arr[i];  hasPositive = true;  } else {  maxSum = Math.max(maxSum, currentSum);  currentSum = 0;  }  }  maxSum = Math.max(maxSum, currentSum);  System.out.print(hasPositive ? maxSum : -1);  }  }  } |
| --- |

**OUTPUT:**



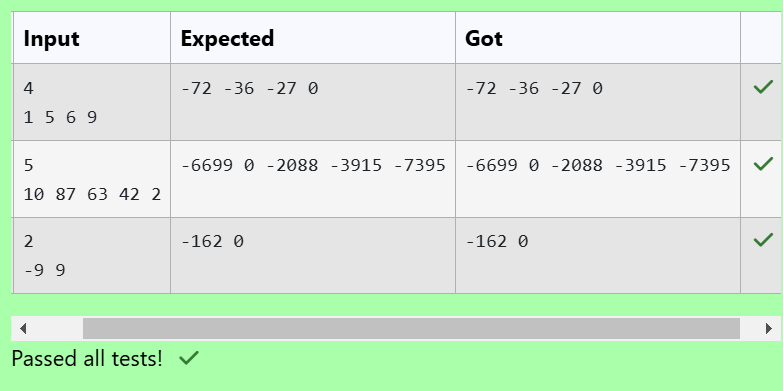
3.2



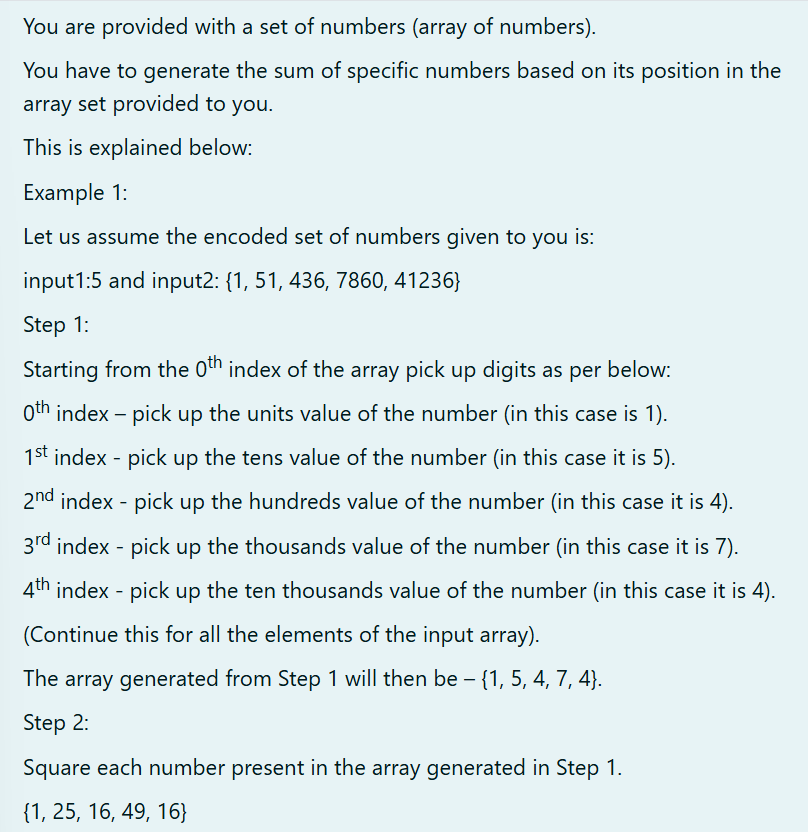
**PROGRAM:**

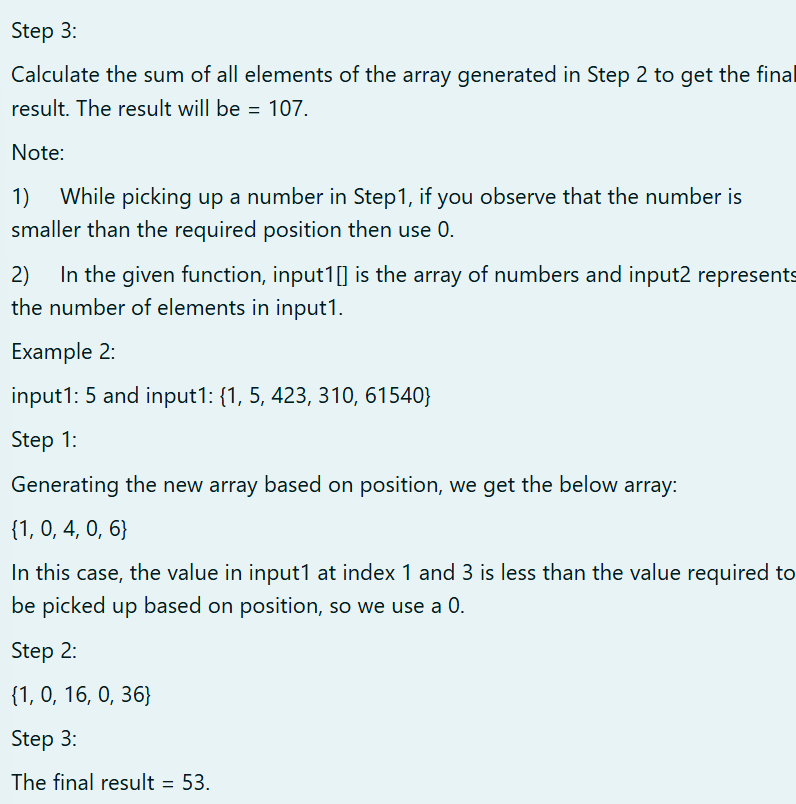
| import java.util.Scanner;  public class ArrayOperations {  public static int[] performOperations(int[] arr) {  int maxNum = arr[0];  for (int num : arr) {  if (num > maxNum) {  maxNum = num;  }  }  for (int i = 0; i < arr.length; i++) {  arr[i] = arr[i] - maxNum;  }  for (int i = 0; i < arr.length; i++) {  arr[i] = arr[i] \* maxNum;  }  return arr;  }  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  int n = scanner.nextInt();  int[] arr = new int[n];  for (int i = 0; i < n; i++) {  arr[i] = scanner.nextInt();  }  int[] result = performOperations(arr);  System.out.print("");  for (int num : result) {  System.out.print(num + " ");  }  }  } |
| --- |

**OUTPUT:**



3.3

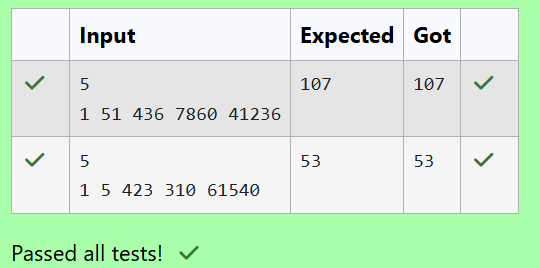




**PROGRAM:**

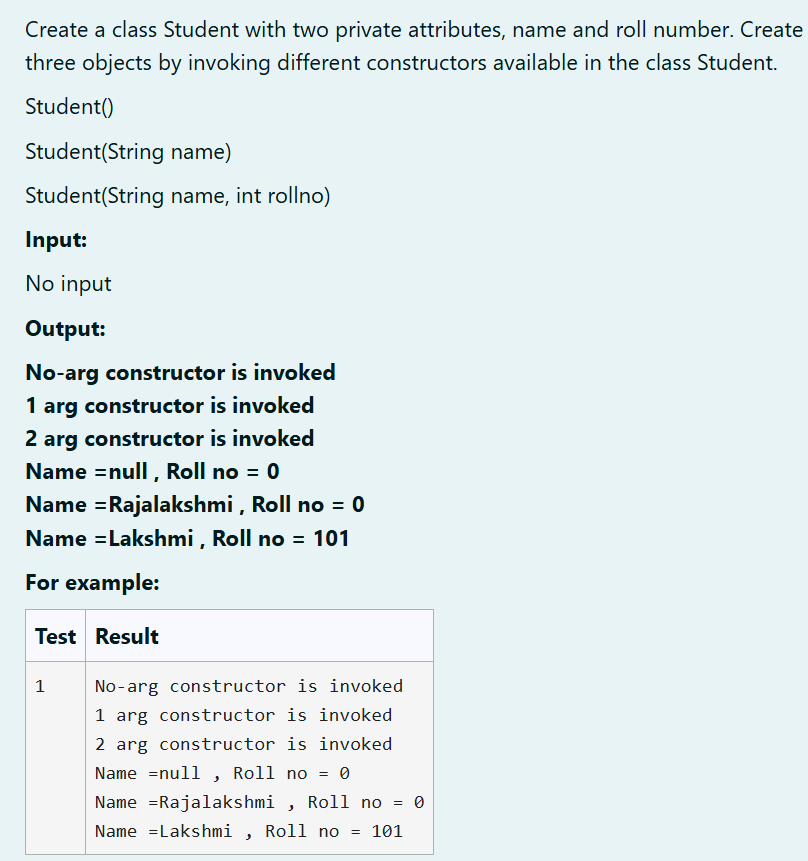
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner a = new Scanner(System.in);  int b = a.nextInt();  int[] c = new int[b];  for (int i = 0; i < b; i++) {  c[i] = a.nextInt();  }  int sum = 0;  for (int i = 0; i < b; i++) {  if (i == 0) {  sum += c[i];  } else if (i == 1) {  sum += (c[i] / 10) \* (c[i] / 10);  } else if (i == 2) {  sum += (c[i] / 100) \* (c[i] / 100);  } else if (i == 3) {  sum += (c[i] / 1000) \* (c[i] / 1000);  } else if (i == 4) {  sum += (c[i] / 10000) \* (c[i] / 10000);  }  }  System.out.print(sum);  }  } |
| --- |

**OUTPUT:**



## Lab-04-Classes and Objects

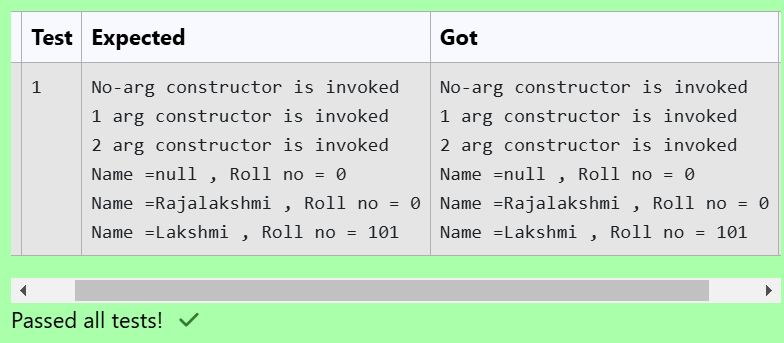
4.1



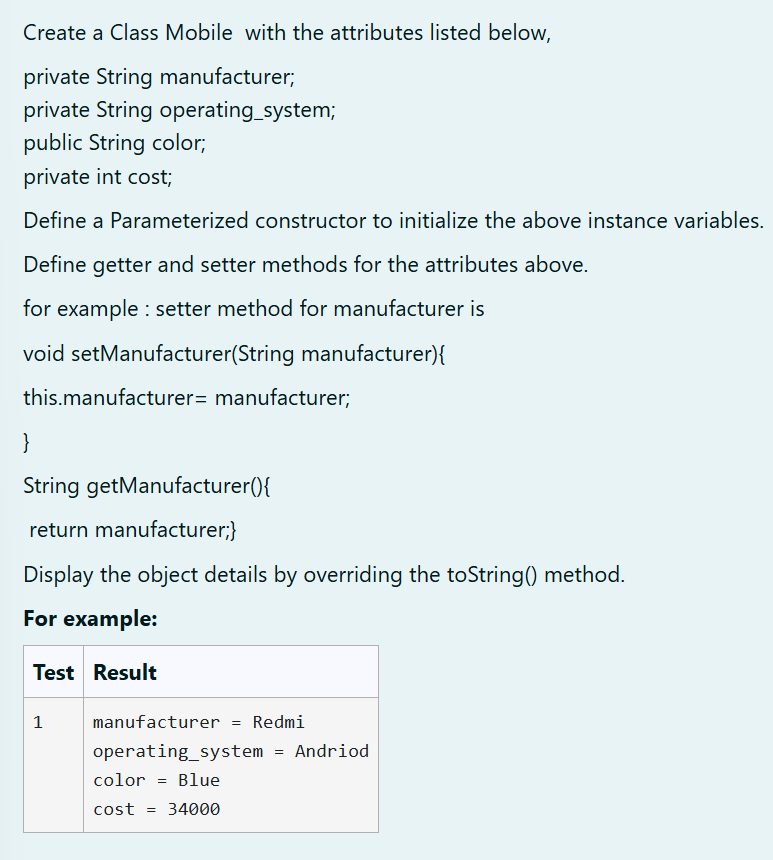
**PROGRAM:**

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

**OUTPUT:**



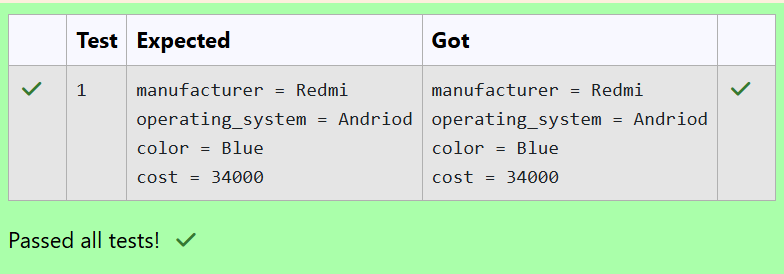
4.2



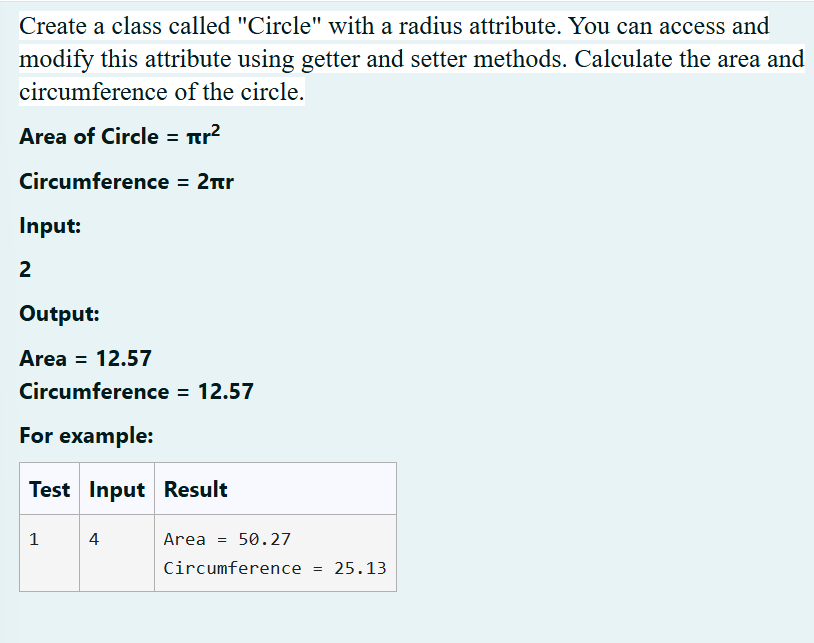
**PROGRAM:**

| public class Mobile {  private String manufacturer;  private String operatingSystem;  public String color;  private int cost;  public Mobile(String manufacturer, String operatingSystem, String color, int cost) {  this.manufacturer = manufacturer;  this.operatingSystem = operatingSystem;  this.color = color;  this.cost = cost;  }  public void setManufacturer(String manufacturer) {  this.manufacturer = manufacturer;  }  public String getManufacturer() {  return manufacturer;  }  public void setOperatingSystem(String operatingSystem) {  this.operatingSystem = operatingSystem;  }  public String getOperatingSystem() {  return operatingSystem;  }  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 +  "\noperating\_system = " + operatingSystem +  "\ncolor = "+color+  "\ncost = " +cost;  }  public static void main(String[] args) {  Mobile mobile = new Mobile("Redmi", "Andriod", "Blue", 34000);  System.out.println(mobile);  }  } |
| --- |

**OUTPUT:**



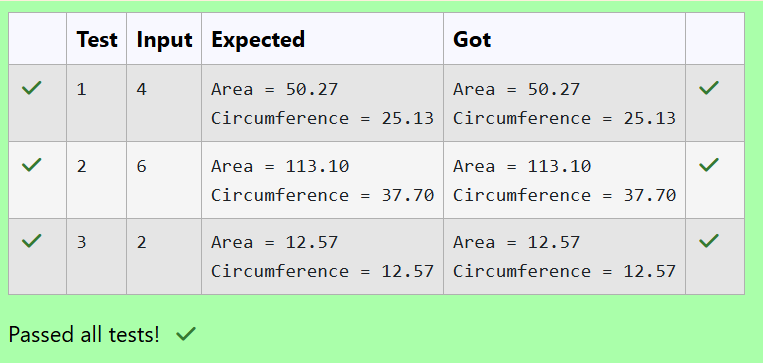
4.3



**PROGRAM:**

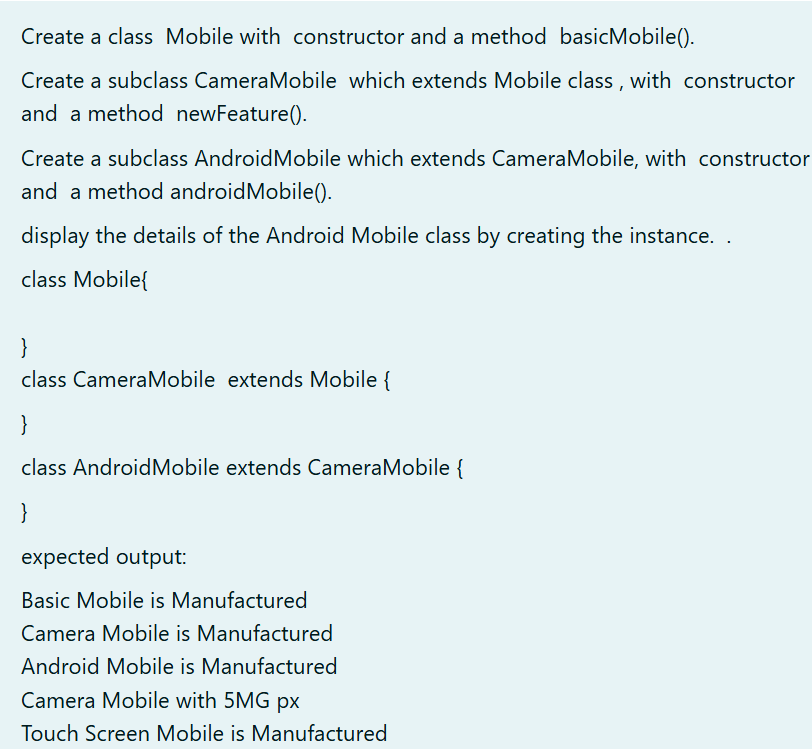
| import java.util.\*;  public class Circle {  public static void main(String args[]){  Scanner a=new Scanner(System.in);  int b=a.nextInt();  double area=Math.PI\*(b\*b);  double cir=Math.PI\*b\*2;  System.out.printf("Area = %.2f%n", area);  System.out.printf("Circumference = %.2f%n",cir);  }  } |
| --- |

**OUTPUT:**



## Lab-05-Inheritance

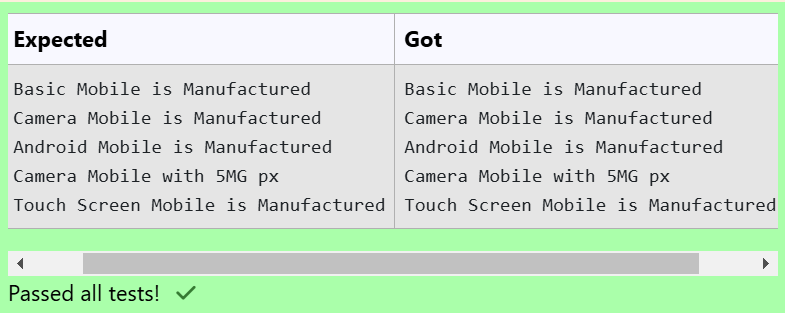
5.1



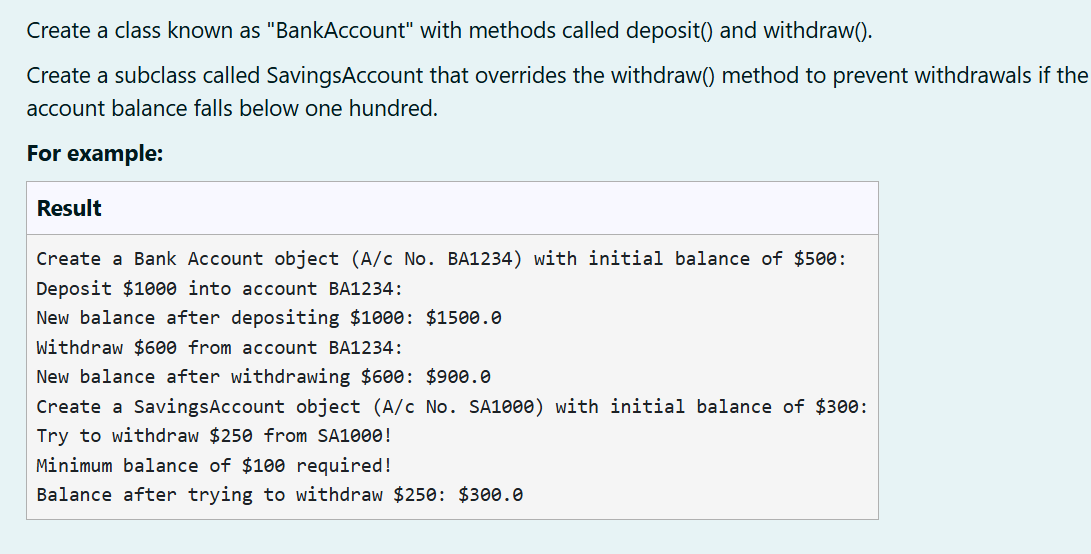
**PROGRAM:**

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

**OUTPUT:**



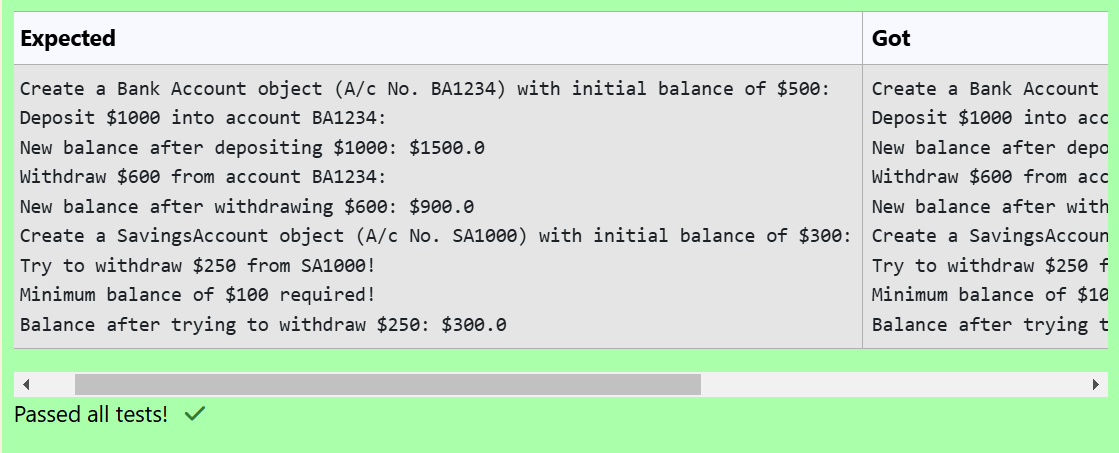
5.2



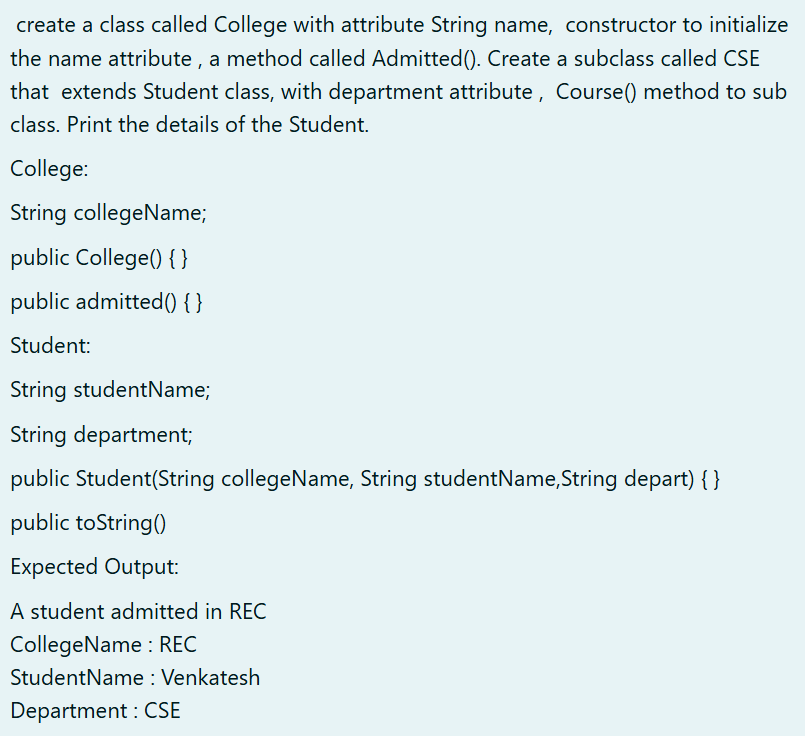
**PROGRAM:**

| class BankAccount {  // Private field to store the account number  private String accountNumber;  // Private field to store the balance  private double balance;  // Constructor to initialize account number and balance  BankAccount(String s,double n)  {  accountNumber=s;  balance=n;  }          // Method to deposit an amount into the account  public void deposit(double amount) {  // Increase the balance by the deposit amount  balance+=amount;  }  // Method to withdraw an amount from the account  public void withdraw(double amount) {  // Check if the balance is sufficient for the withdrawal  if (balance >= amount) {  // Decrease the balance by the withdrawal amount  balance -= amount;  } else {  // Print a message if the balance is insufficient  System.out.println("Insufficient balance");  }  }  // Method to get the current balance  public double getBalance() {  // Return the current balance  return balance;  }  }  class SavingsAccount extends BankAccount {  // Constructor to initialize account number and balance  public SavingsAccount(String accountNumber, double balance) {  // Call the parent class constructor  super(accountNumber,balance);  }  // Override the withdraw method from the parent class  @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);  }  }  }  public class Main {    public static void main(String[] args) {  // Print message to indicate creation of a BankAccount object  System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");  // Create a BankAccount object (A/c No. "BA1234") with initial balance of $500  BankAccount BA1234 = new BankAccount("BA1234", 500);  // Print message to indicate deposit action  System.out.println("Deposit $1000 into account BA1234:");  // Deposit $1000 into account BA1234  BA1234.deposit(1000);  // Print the new balance after deposit    System.out.println("New balance after depositing $1000: $"+BA1234.getBalance());  // Print message to indicate withdrawal action  System.out.println("Withdraw $600 from account BA1234:");  // Withdraw $600 from account BA1234  BA1234.withdraw(600);  // Print the new balance after withdrawal  System.out.println("New balance after withdrawing $600: $" + BA1234.getBalance());  //Print message to indicate creation of another SavingsAccount object  System.out.println("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  SavingsAccount SA1000 = new SavingsAccount("SA1000", 300);    // Print message to indicate withdrawal action  System.out.println("Try to withdraw $250 from SA1000!");  // Withdraw $250 from SA1000 (balance falls below $100)  SA1000.withdraw(250);  // Print the balance after attempting to withdraw $250  System.out.println("Balance after trying to withdraw $250: $" + SA1000.getBalance());  }  } |
| --- |

**OUTPUT:**



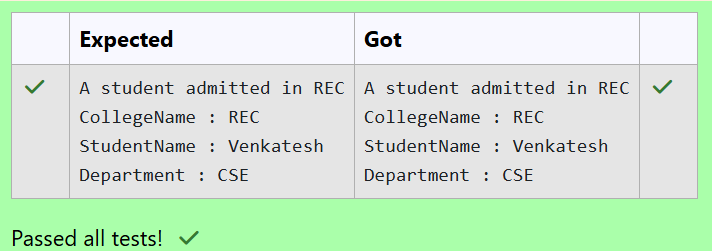
5.3



**PROGRAM:**

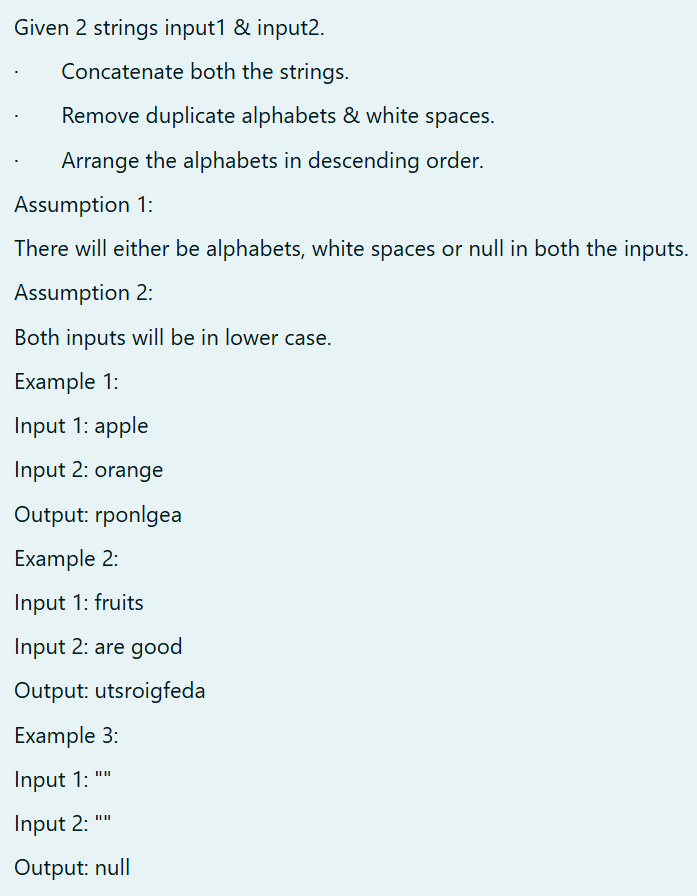
| class College  {  protected String collegeName;  public College(String collegeName) {  // initialize the instance variables  this.collegeName=collegeName;  }  public void admitted() {  System.out.println("A student admitted in "+collegeName);  }  }  class Student extends College{  String studentName;  String department;  public Student(String collegeName, String studentName,String depart) {  // initialize the instance variables  super(collegeName);  this.studentName=studentName;  this.department=depart;    }  public String toString(){  // return the details of the student  return "CollegeName : "+collegeName+"\nStudentName : "+studentName+"\nDepartment : "+department;  }  }  public class Main  {  public static void main (String[] args)  {  Student s1 = new Student("REC","Venkatesh","CSE");  s1.admitted(); // invoke the admitted() method  System.out.println(s1.toString());  }  } |
| --- |

**OUTPUT:**



## Lab-06-String, StringBuffer

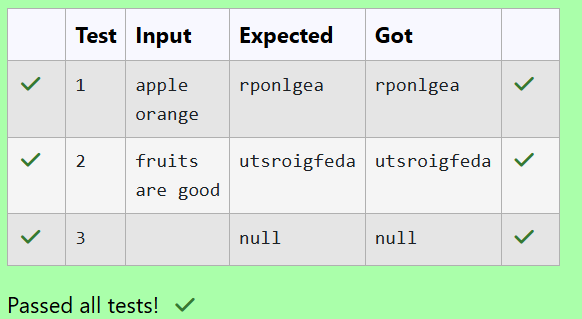
6.1



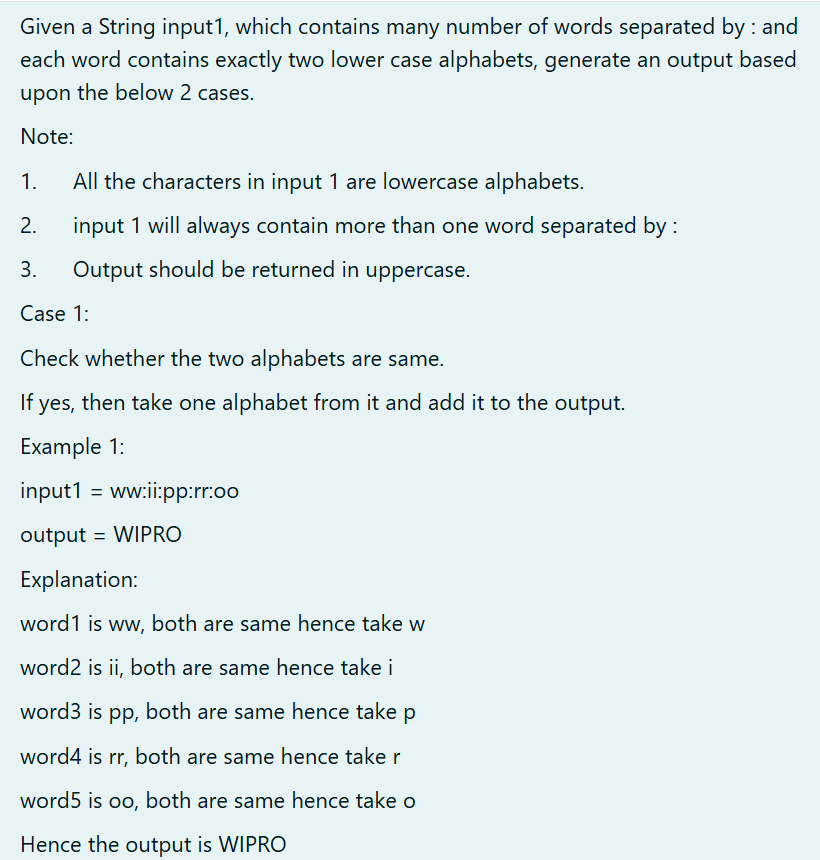
**PROGRAM:**

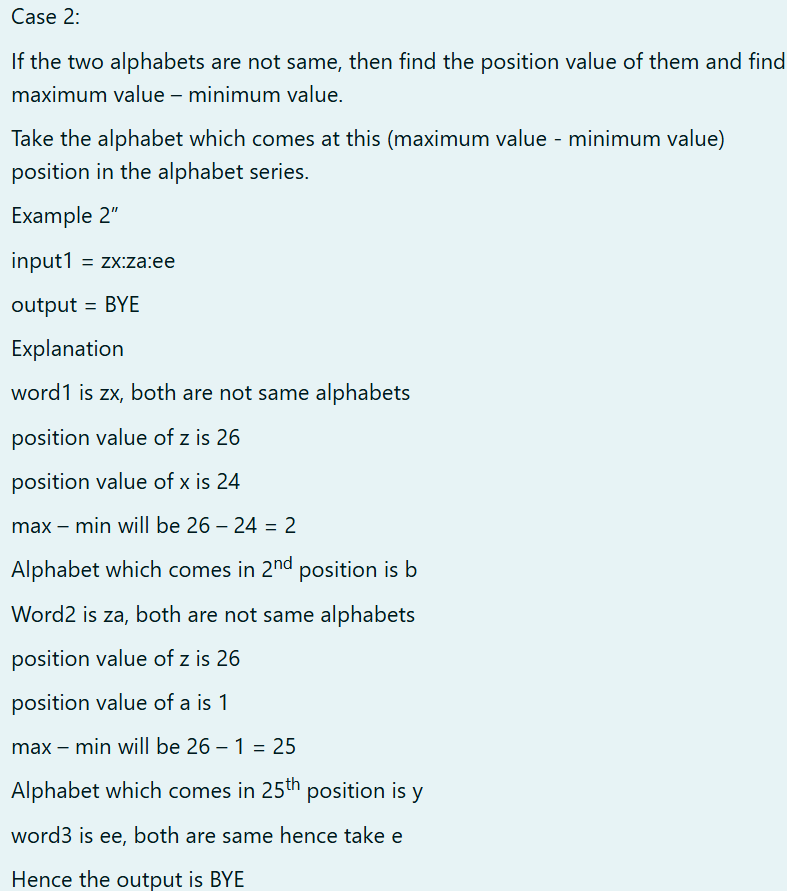
| import java.util.\*;  public class main{  public static String process(String s1,String s2)  {  StringBuilder s3= new StringBuilder(s1+s2);  for(int i=0;i<s3.length()-1;i++)  {  for(int j=i+1;j<s3.length();j++)  {  if(s3.charAt(i)==s3.charAt(j)||s3.charAt(i)==' ')  {  s3.deleteCharAt(i);  i--;  break;  }  }  }  String s=s3.toString();  char c[]=s.toCharArray();  Arrays.sort(c);  for(int i=0,j=c.length-1;j>i;i++,j--)  {  char temp=c[i];  c[i]=c[j];  c[j]=temp;  }    return String.valueOf(c);    }  public static void main(String args[])  {  Scanner s=new Scanner(System.in);  String s1=s.nextLine();  String s2=s.nextLine();  //System.out.println(s1.length());  int f=0;  for(int i=0;i<s1.length();i++){  if(s1.charAt(i)>='a' && s1.charAt(i)<='z')  {  f=1;  break;  }  }  if(f==0)  System.out.println("null");  else  System.out.print(process(s1,s2));    }  } |
| --- |

**OUTPUT:**



6.2

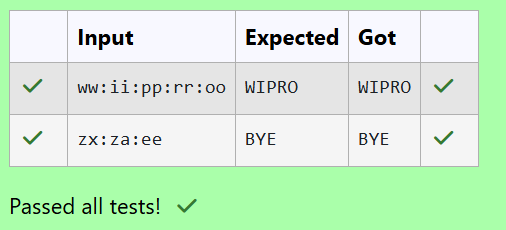




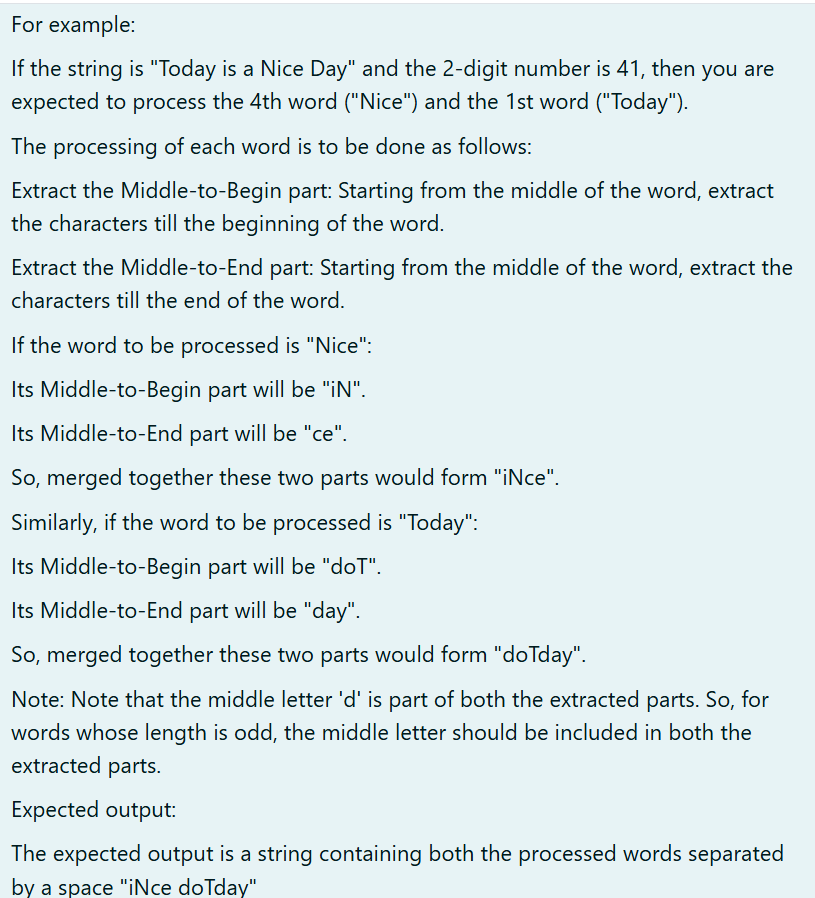
**PROGRAM:**

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

**OUTPUT:**



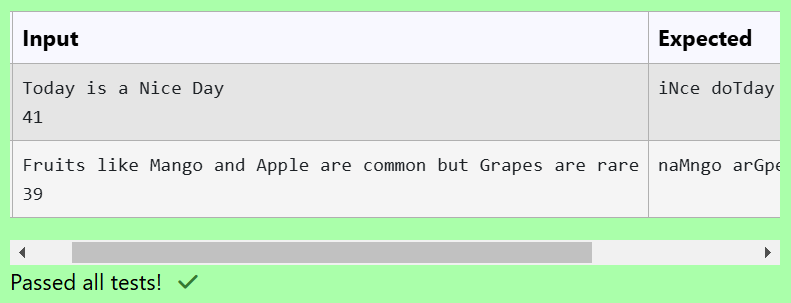
6.3



**PROGRAM:**

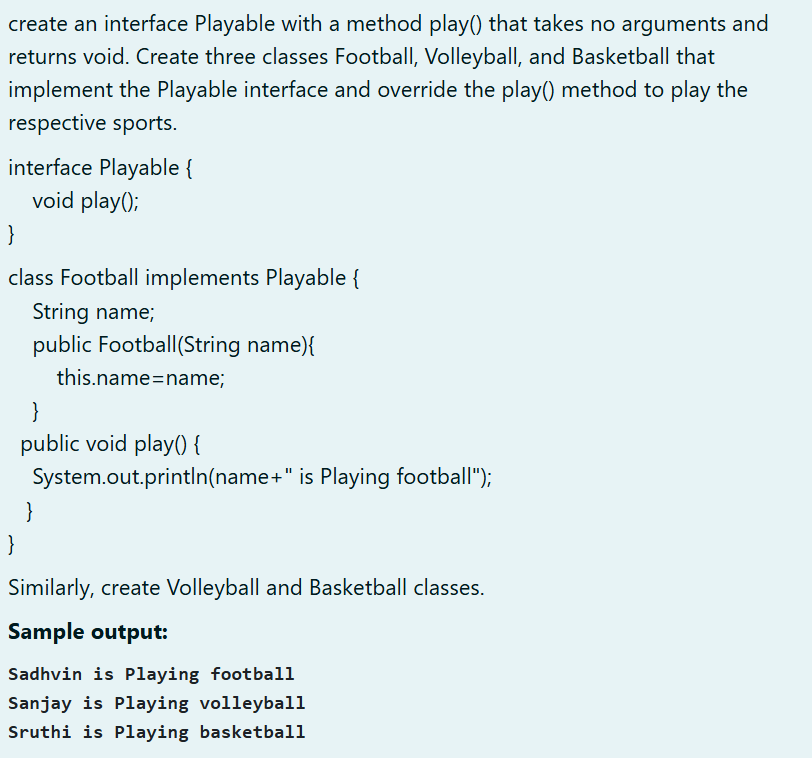
| import java.util.Scanner;  public class main{  public static void process(String s1)  {  int mid=s1.length()/2;  int midbegin=mid;  //System.out.print(mid);  if(s1.length()%2!=0)  {  for(int i=midbegin;i>=0;i--)  {  System.out.print(s1.charAt(i));  }  for(int i=midbegin;i<s1.length();i++)  {  System.out.print(s1.charAt(i));  }  }  else{  for(int i=midbegin-1;i>=0;i--)  {  System.out.print(s1.charAt(i));  }  for(int i=midbegin;i<s1.length();i++)  {  System.out.print(s1.charAt(i));  }  }  System.out.print(" ");  }  public static void main(String args[])  {  Scanner sc=new Scanner(System.in);  String s=sc.nextLine();  int n=sc.nextInt();  String st[]=s.split(" ");  String s1=st[n/10-1];  String s2=st[n%10-1];  process(s1);  process(s2);  }  } |
| --- |

**OUTPUT:**



## Lab-07-Interfaces

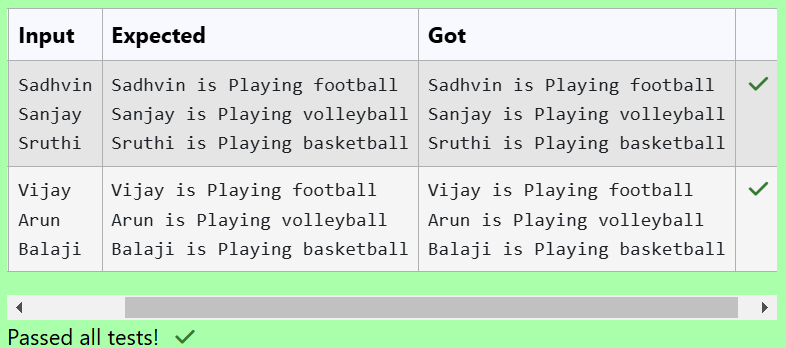
7.1



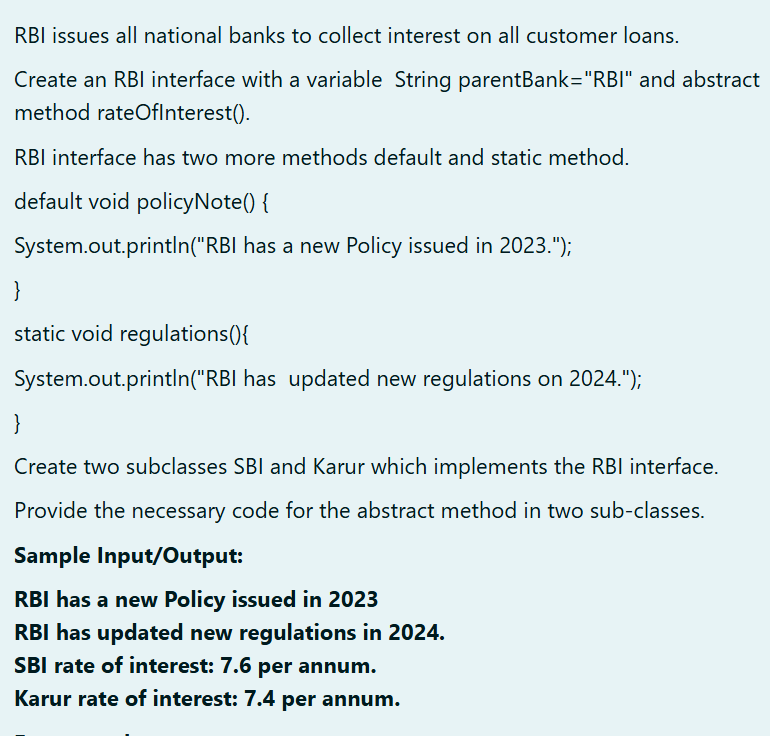
**PROGRAM:**

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

**OUTPUT:**



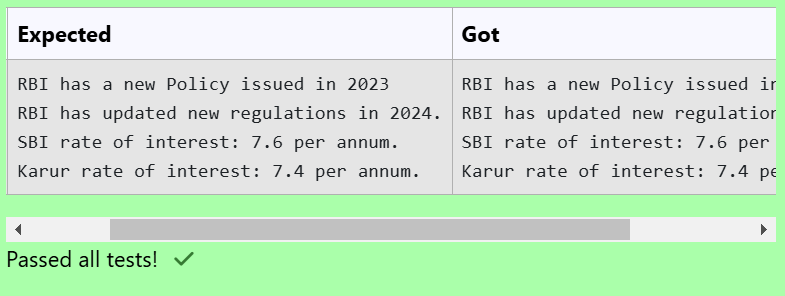
7.2



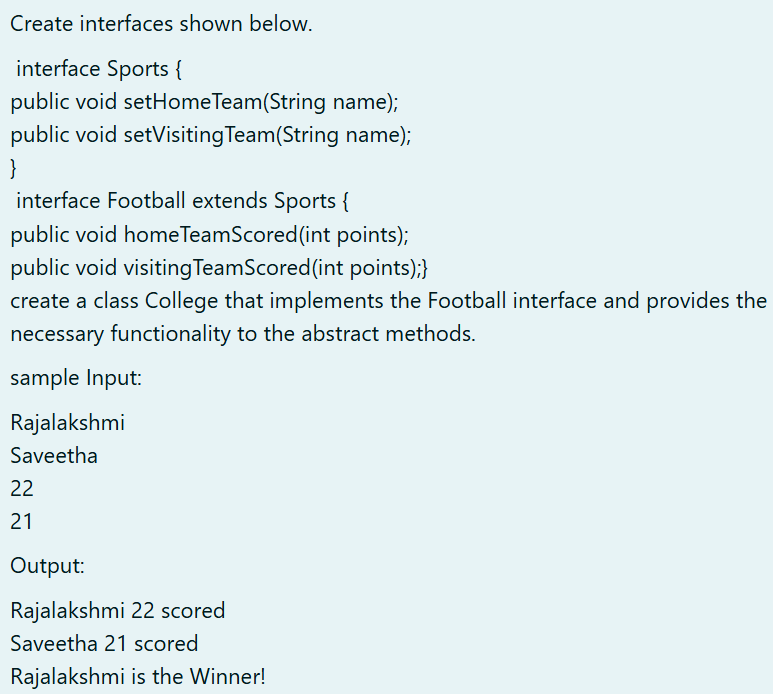
**PROGRAM:**

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

**OUTPUT:**



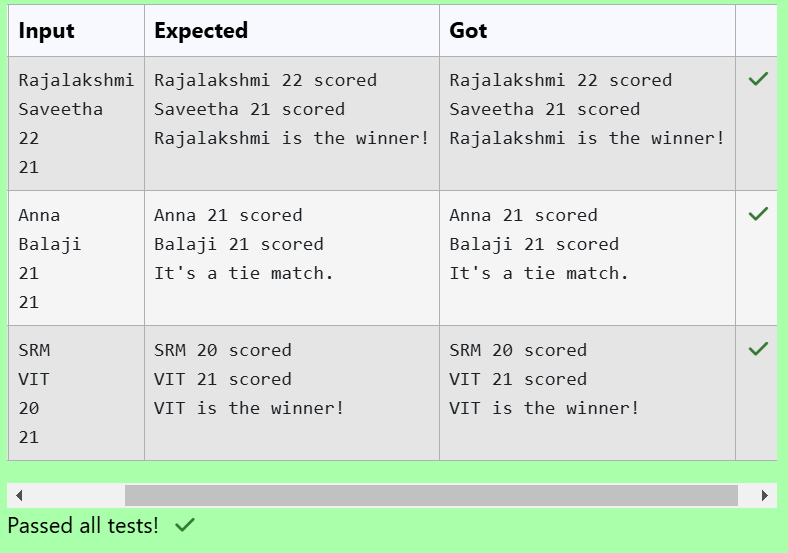
7.3



**PROGRAM:**

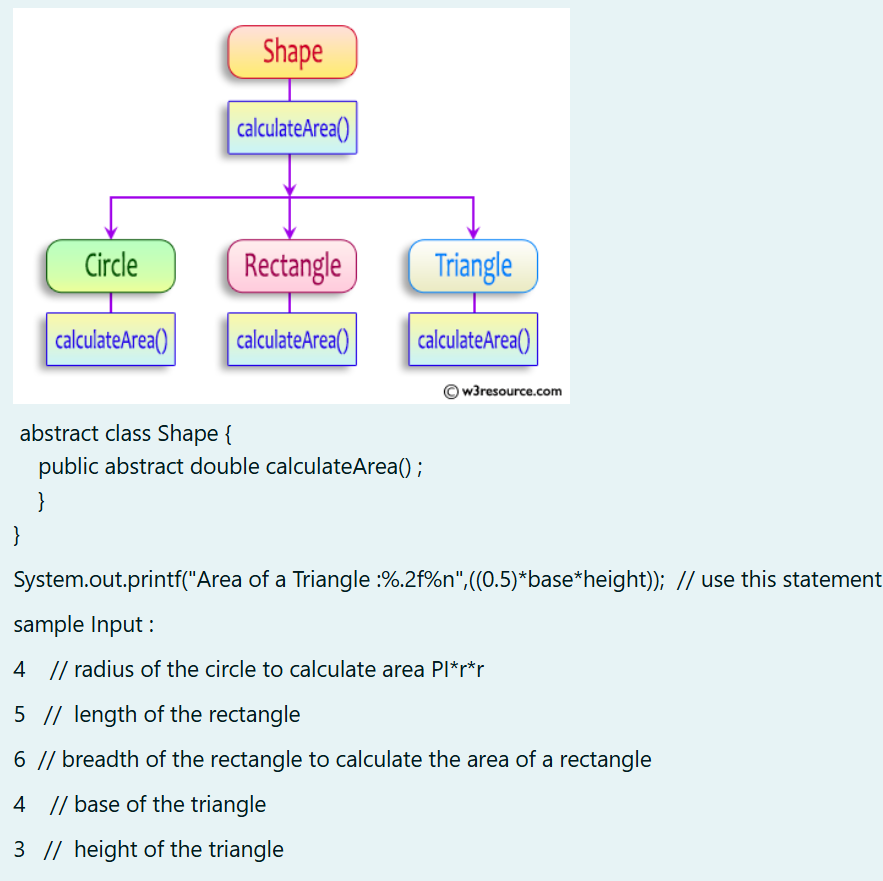
| import java.util.Scanner;  interface Sports {  public void setHomeTeam(String name);  public void setVisitingTeam(String name);    }  interface Football extends Sports {  public void homeTeamScored(int points);  public void visitingTeamScored(int points);  }  class College implements Football {  String homeTeam;  String visitingTeam;    public void setHomeTeam(String name){  homeTeam=name;  }  public void setVisitingTeam(String name){  visitingTeam=name;  }  public void homeTeamScored(int points){  System.out.println(homeTeam+" "+points+" scored");  }  public void visitingTeamScored(int points){  System.out.println(visitingTeam+" "+points+" scored");  }  public void winningTeam(int p1, int p2){  if(p1>p2)  System.out.println(homeTeam+" is the winner!");  else if(p1<p2)  System.out.println(visitingTeam+" is the winner!");  else  System.out.println("It's a tie match.");  }  }  public class Main{  public static void main(String[] args){  String hname;  Scanner sc= new Scanner(System.in);  hname=sc.nextLine();  String vteam=sc.next();  int htpoints=sc.nextInt();  int vtpoints=sc.nextInt();  College s= new College();  s.setHomeTeam(hname);  s.setVisitingTeam(vteam);  s.homeTeamScored(htpoints);  s.visitingTeamScored(vtpoints);  s.winningTeam(htpoints,vtpoints);  }  } |
| --- |

**OUTPUT:**



## Lab-08 - Polymorphism, Abstract Classes, final Keyword

8.1

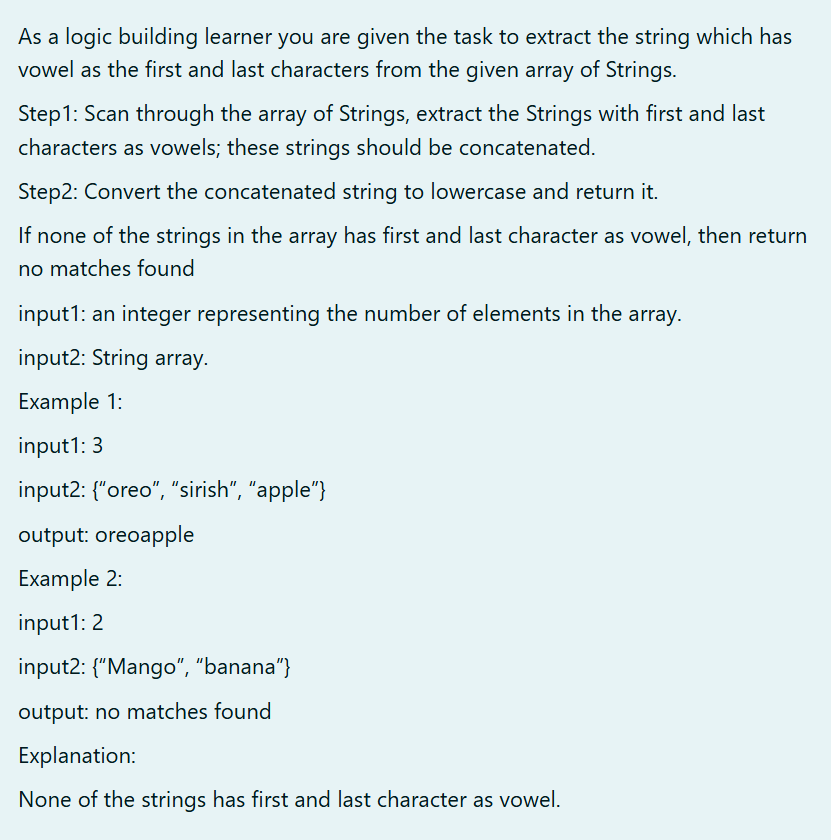


**PROGRAM:**

| import java.util.Scanner;  abstract class Shape {  public abstract double calculateArea();  }  class Circle extends Shape {  private double radius;  public Circle(double radius) {  this.radius = radius;  }    public double calculateArea() {  return Math.PI \* radius \* radius;  }  }  class Rectangle extends Shape {  private double length;  private double breadth;  public Rectangle(double length, double breadth) {  this.length = length;  this.breadth = breadth;  }  @Override  public double calculateArea() {  return length \* breadth;  }  }  class Triangle extends Shape {  private double base;  private double height;  public Triangle(double base, double height) {  this.base = base;  this.height = height;  }    public double calculateArea() {  return 0.5 \* base \* height;  }  }  public class ShapeAreaCalculator {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  double circleRadius = scanner.nextDouble();  double rectangleLength = scanner.nextDouble();  double rectangleBreadth = scanner.nextDouble();  double triangleBase = scanner.nextDouble();  double triangleHeight = scanner.nextDouble();  Shape circle = new Circle(circleRadius);  Shape rectangle = new Rectangle(rectangleLength, rectangleBreadth);  Shape triangle = new Triangle(triangleBase, triangleHeight);  System.out.printf("Area of a circle: %.2f%n", circle.calculateArea());  System.out.printf("Area of a Rectangle: %.2f%n", rectangle.calculateArea());  System.out.printf("Area of a Triangle: %.2f%n", triangle.calculateArea());  scanner.close();  }  } |
| --- |

**OUTPUT:**

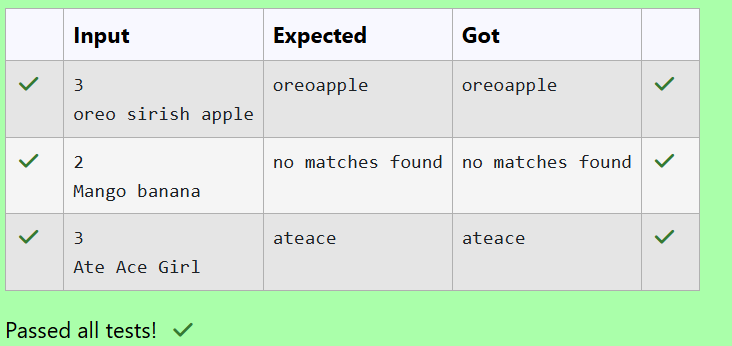
8.2



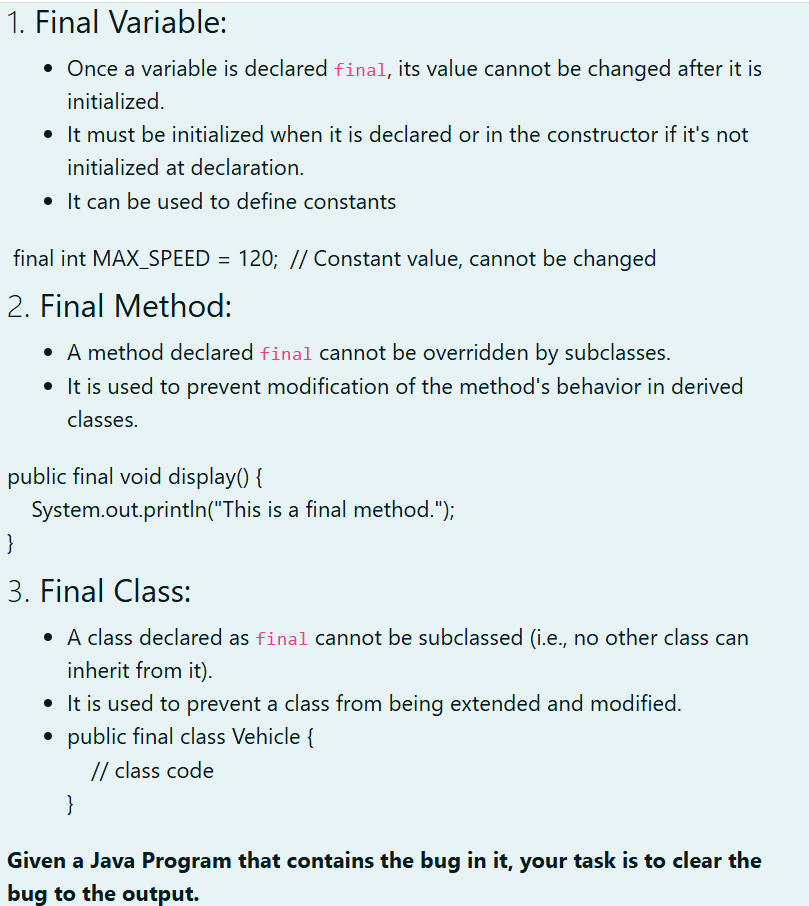
**PROGRAM:**

| import java.util.Scanner;  public class VowelExtractor {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  int numElements = scanner.nextInt();  String[] strings = new String[numElements];  for (int i = 0; i < numElements; i++) {  strings[i] = scanner.next();  }  String result = extractVowelStrings(strings);  System.out.println(result);  }  public static String extractVowelStrings(String[] strings) {  StringBuilder concatenatedString = new StringBuilder();  for (String str : strings) {  char firstChar = str.charAt(0);  char lastChar = str.charAt(str.length() - 1);  if (isVowel(firstChar) && isVowel(lastChar)) {  concatenatedString.append(str.toLowerCase());  }  }  if (concatenatedString.length() == 0) {  return "no matches found";  } else {  return concatenatedString.toString();  }  }  public static boolean isVowel(char c) {  c = Character.toLowerCase(c);  return c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u';  }  } |
| --- |

**OUTPUT:**



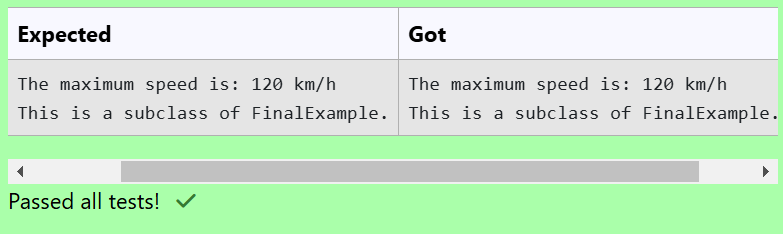
8.3



**PROGRAM:**

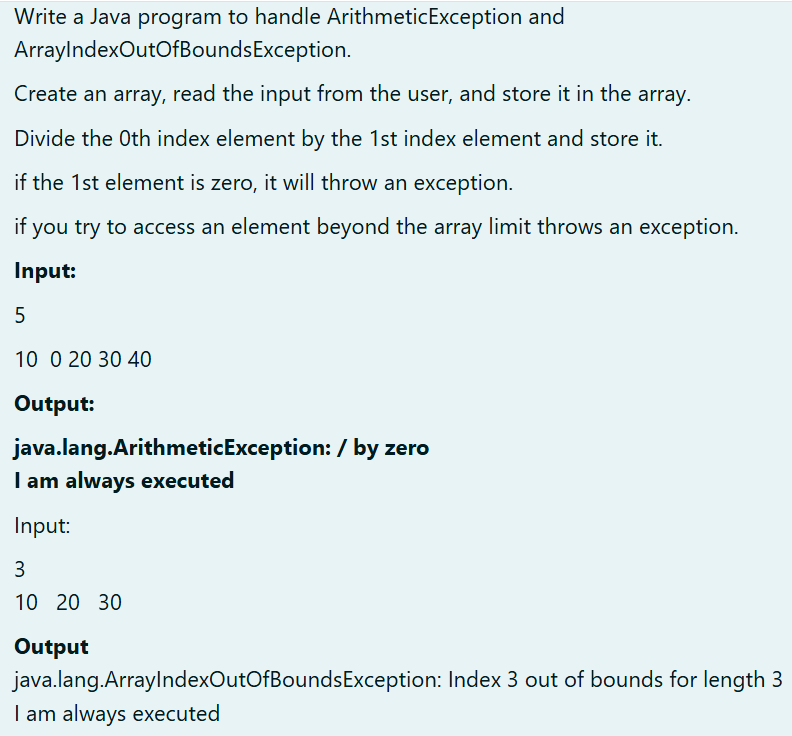
| public class FinalExample {  final int MAX\_SPEED = 120; // Constant value, cannot be changed  public final void display() {  System.out.println("This is a subclass of FinalExample.");  }  public static void main(String[] args) {  FinalExample obj = new FinalExample();  System.out.println("The maximum speed is: " + obj.MAX\_SPEED + " km/h");  obj.display();  }  }  class SubClass extends FinalExample {  public static void main(String[] args) {  System.out.println("This is a subclass of FinalExample.");  }  } |
| --- |

**OUTPUT:**



## Lab-09-Exception Handling

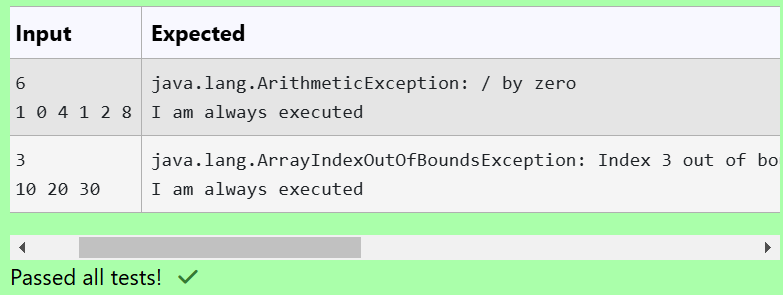
9.1



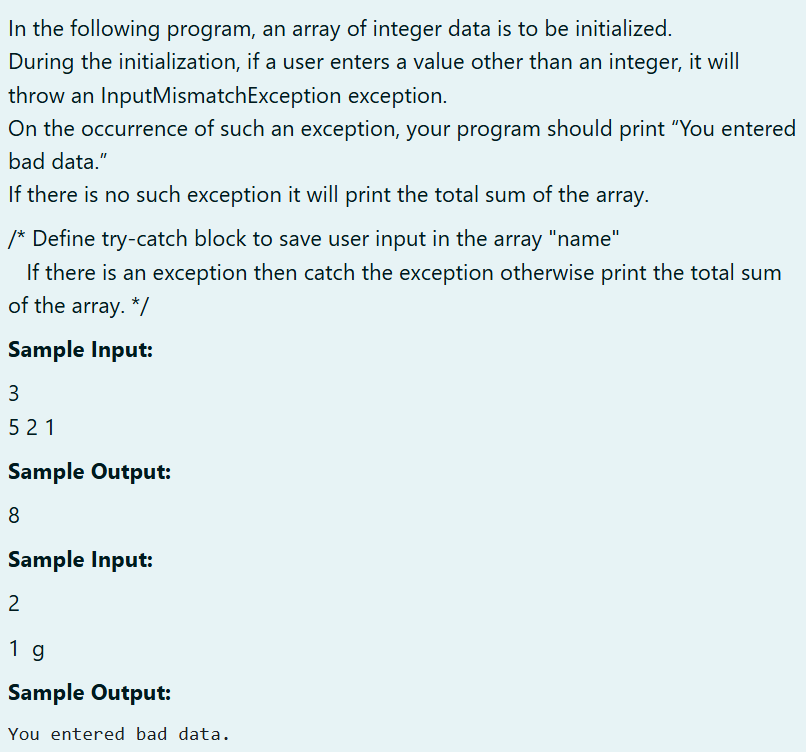
**PROGRAM:**

| import java.util.Scanner;  public class ExceptionHandling {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);    int size = scanner.nextInt();  int[] array = new int[size];    for (int i = 0; i < size; i++) {  array[i] = scanner.nextInt();  }  try {    int result = array[0] / array[1];  if(result ==0)  System.out.println("java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3");  else  System.out.println("Result: " + result);    } catch (ArithmeticException e) {  System.out.println(e);  } finally {  System.out.println("I am always executed");  }  }  } |
| --- |

**OUTPUT:**



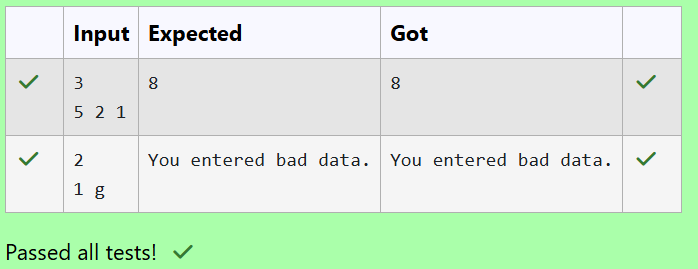
9.2



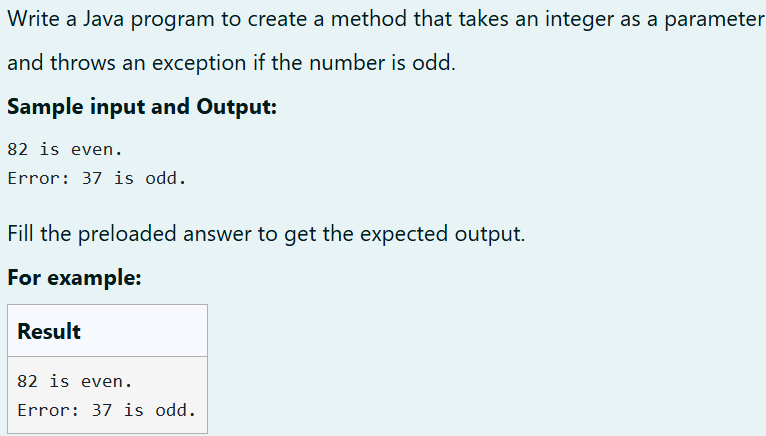
**PROGRAM:**

| import java.util.InputMismatchException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  int n = scanner.nextInt();  int[] array = new int[n];  try {  for (int i = 0; i < n; i++) {  array[i] = scanner.nextInt();  }  int sum = 0;  for (int i = 0; i < n; i++) {  sum += array[i];  }  System.out.println(sum);  } catch (InputMismatchException e) {  System.out.println("You entered bad data.");  }  }  } |
| --- |

**OUTPUT:**



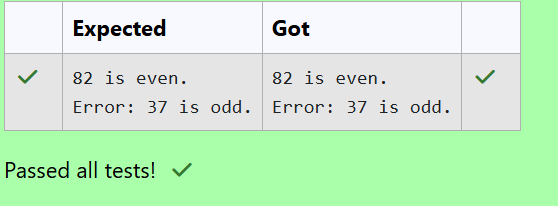
9.3



**PROGRAM:**

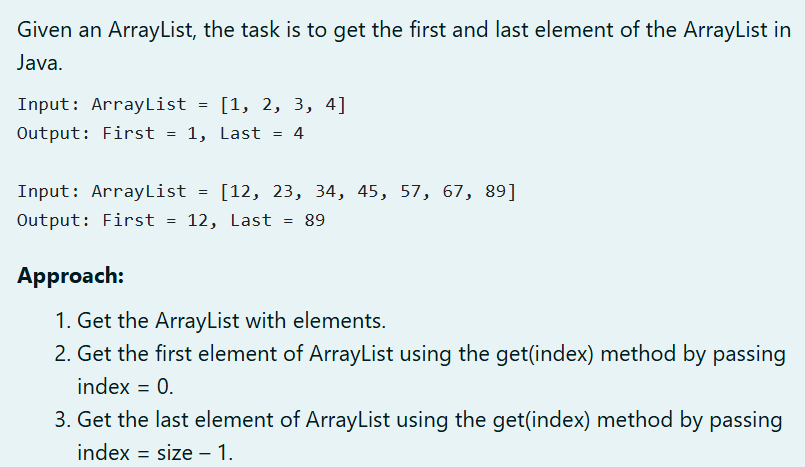
| public class Main {  public static void main(String[] args) {  checkNumber(82);  checkNumber(37);  }  public static void checkNumber(int number) {  try {  if (number % 2 != 0) {  throw new Exception("Error: " + number + " is odd.");  }  System.out.println(number + " is even.");  } catch (Exception e) {  System.out.println(e.getMessage());  }  }  } |
| --- |

**OUTPUT:**



## Lab-10- Collection- List

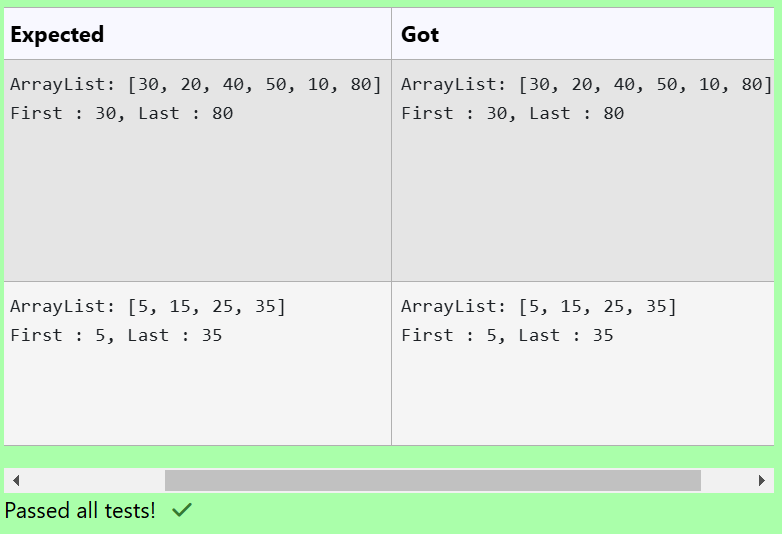
10.1



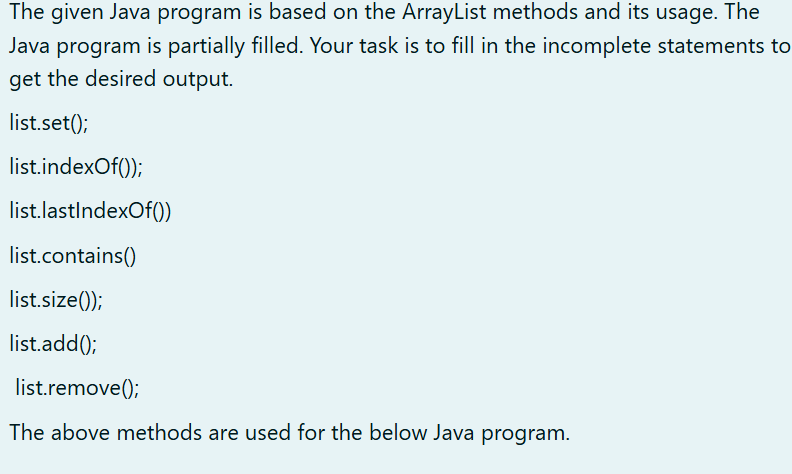
**PROGRAM:**

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

**OUTPUT:**



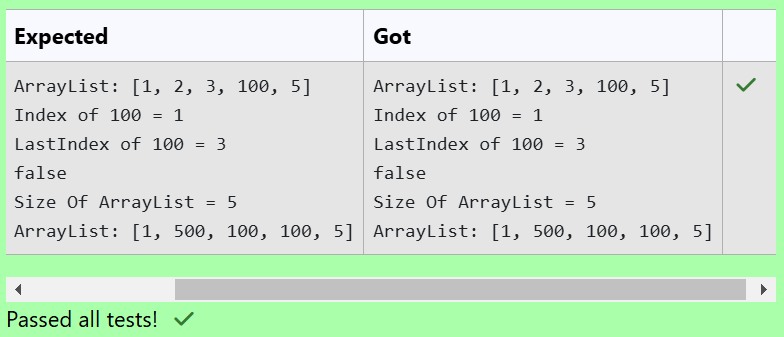
10.2



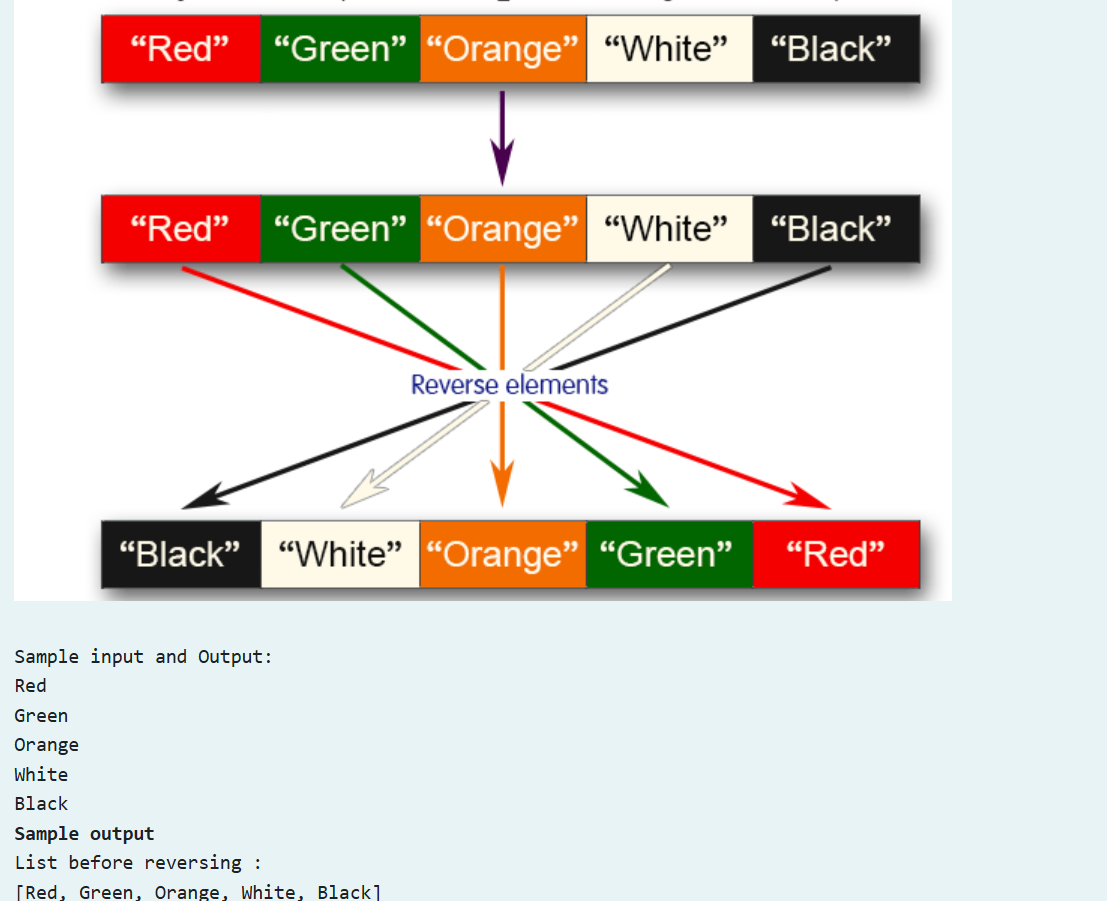
**PROGRAM:**

| import java.util.ArrayList;  import java.util.Scanner;  class prog {  public static void main(String[] args)  {  Scanner sc= new Scanner(System.in);  int n = sc.nextInt();  ArrayList<Integer> list = new ArrayList<Integer>();  for(int i = 0; i<n;i++)  {  list.add(sc.nextInt());  }  // printing initial value ArrayList  System.out.println("ArrayList: " +list);  list.set(1,100);  //Replacing the element at index 1 with 100  //Getting the index of first occurrence of 100  System.out.println("Index of 100 = "+list.indexOf(100));  //Getting the index of last occurrence of 100  System.out.println("LastIndex of 100 = "+list.lastIndexOf(100));  // Check whether 200 is in the list or not  System.out.println(list.contains(200)); //Output : false  // Print ArrayList size  System.out.println("Size Of ArrayList = "+list.size());  //Inserting 500 at index 1  list.add(1,500); // code here  //Removing an element from position 3  list.remove(3); // code here  System.out.print("ArrayList: " + list);  }  } |
| --- |

**OUTPUT:**



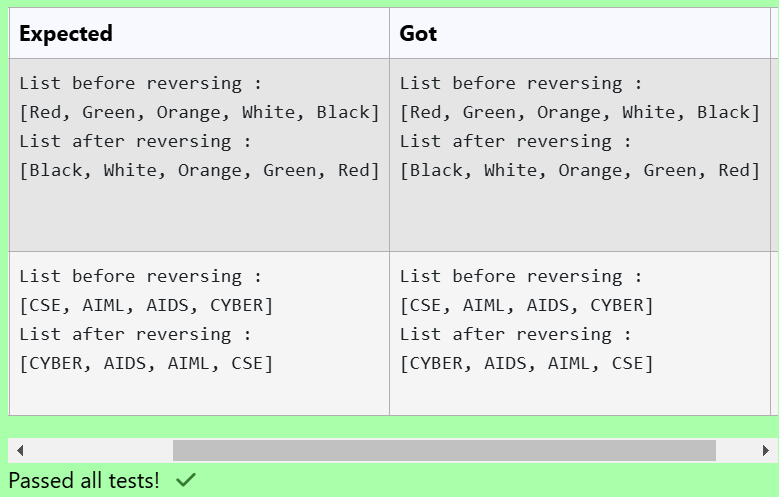
10.3



**PROGRAM:**

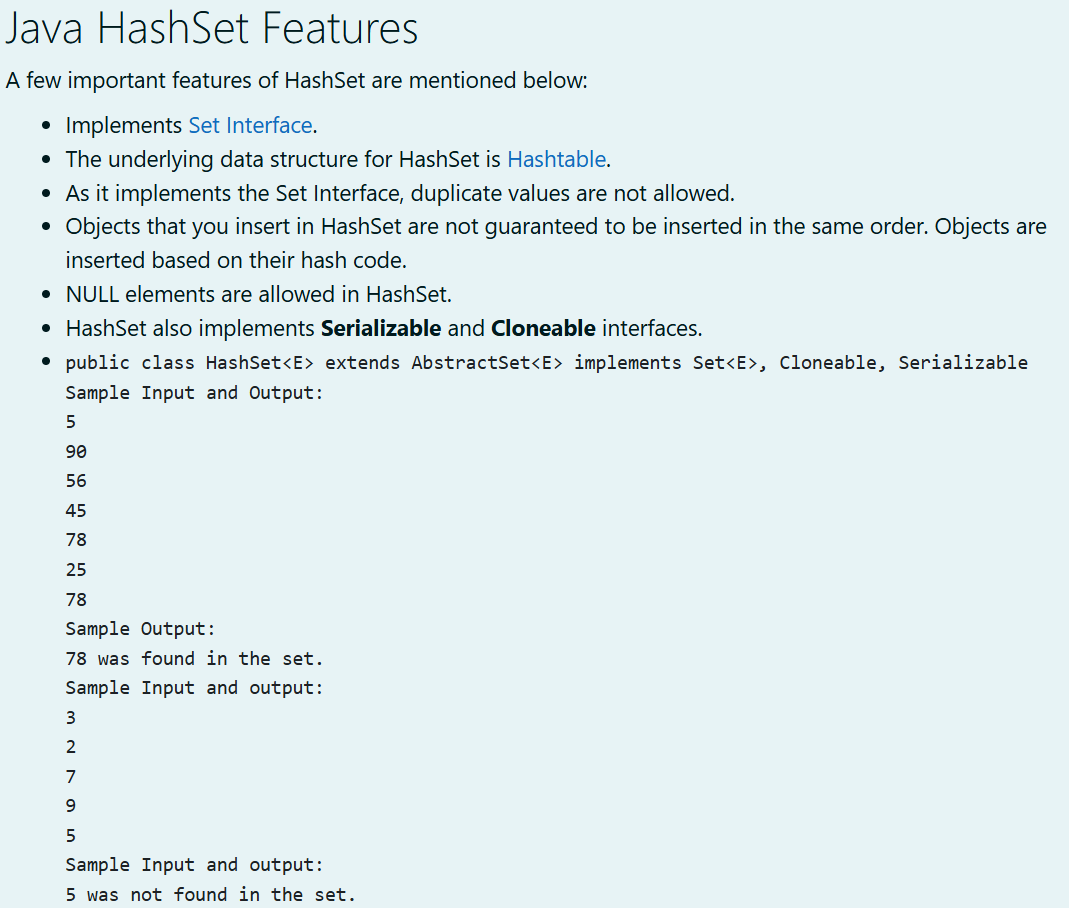
| import java.util.ArrayList;  import java.util.Collections;  import java.util.Scanner;  public class ReverseArrayList  {  public static void main(String[] args)  {  Scanner sc = new Scanner(System.in);  int n = sc.nextInt();  sc.nextLine();    ArrayList<String>elements = new ArrayList<>();    for(int i=0;i<n;i++)  {  String element = sc.nextLine();  elements.add(element);  }    System.out.println("List before reversing :\n" + elements);  Collections.reverse(elements);  System.out.println("List after reversing :\n" + elements);    sc.close();  }  } |
| --- |

**OUTPUT:**



## Lab-11-Set, Map

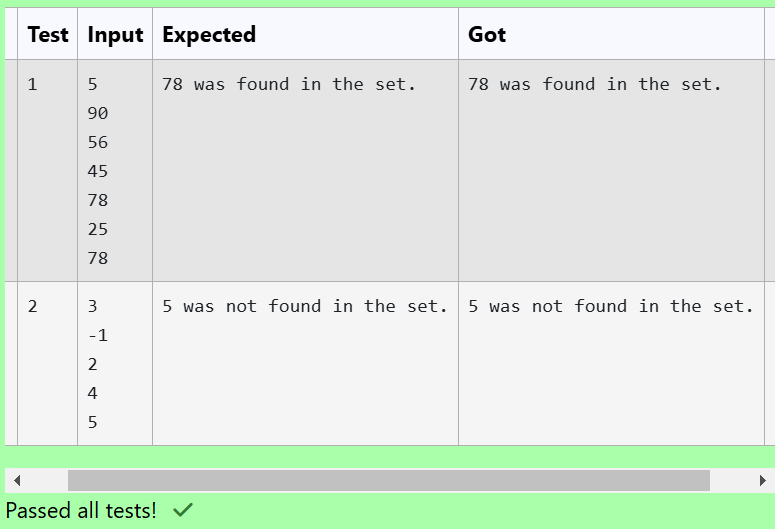
11.1



**PROGRAM:**

| import java.util.HashSet;  import java.util.Scanner;  public class HashSetExample {  public static void main(String[] args) {  // Create a scanner object to take input  Scanner scanner = new Scanner(System.in);  // Read the number of elements to be inserted into the set  int n = scanner.nextInt();  // Create a HashSet to store the elements  HashSet<Integer> set = new HashSet<>();  // Insert the elements into the set  for (int i = 0; i < n; i++) {  int element = scanner.nextInt();  set.add(element);  }  // Read the element to be searched in the set  int searchElement = scanner.nextInt();  // Check if the element is in the set and print the result  if (set.contains(searchElement)) {  System.out.println(searchElement + " was found in the set.");  } else {  System.out.println(searchElement + " was not found in the set.");  }  // Close the scanner  scanner.close();  }  } |
| --- |

**OUTPUT:**



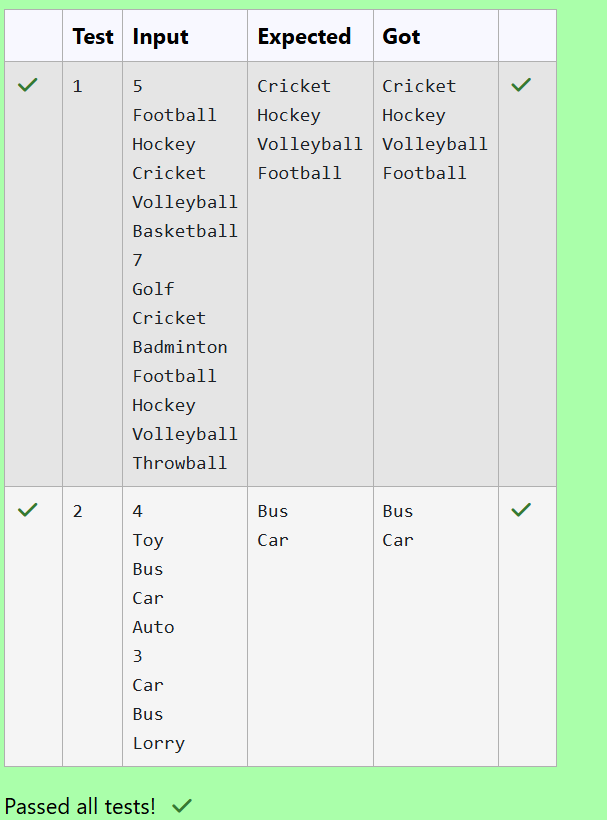
11.2



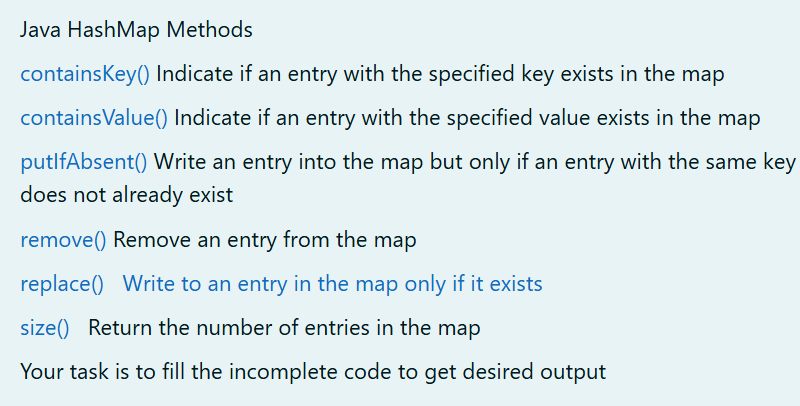
**PROGRAM:**

| import java.util.HashSet;  import java.util.Scanner;  import java.util.Set;  public class SetComparison {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  // Read size and elements for the first set  int n1 = scanner.nextInt();  Set<String> set1 = new HashSet<>();  scanner.nextLine(); // Consume the newline after the integer input  for (int i = 0; i < n1; i++) {  set1.add(scanner.nextLine());  }  // Read size and elements for the second set  int n2 = scanner.nextInt();  Set<String> set2 = new HashSet<>();  scanner.nextLine(); // Consume the newline after the integer input  for (int i = 0; i < n2; i++) {  set2.add(scanner.nextLine());  }  // Retain only common elements  set1.retainAll(set2);  // Output the common elements  for (String element : set1) {  System.out.println(element);  }  scanner.close();  }  } |
| --- |

**OUTPUT:**



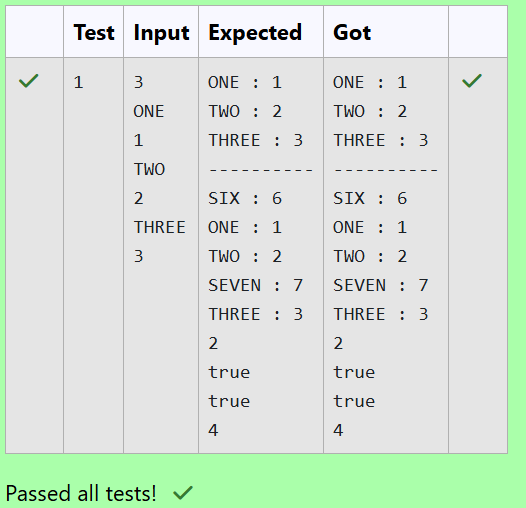
11.3



**PROGRAM:**

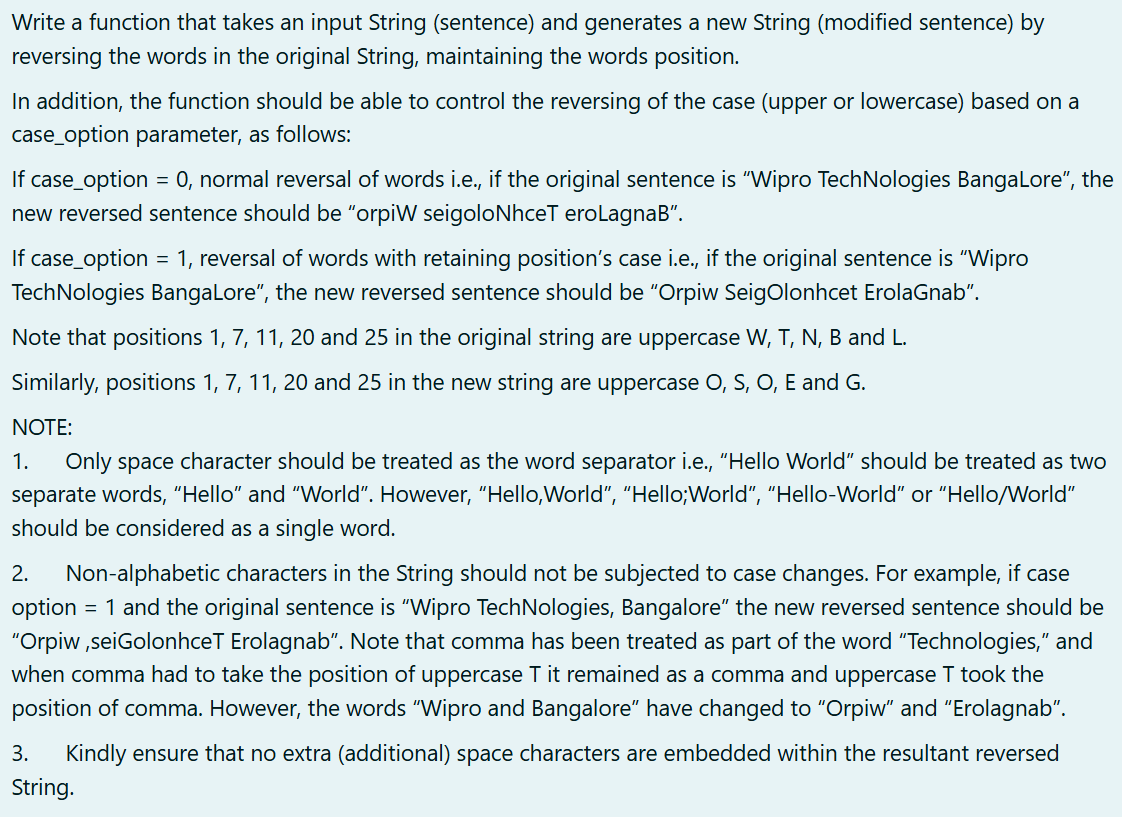
| import java.util.HashMap;  import java.util.Map.Entry;  import java.util.Set;  import java.util.Scanner;  class prog {  public static void main(String[] args) {  // Creating a HashMap with default initial capacity and load factor  HashMap<String, Integer> map = new HashMap<String, Integer>();  String name;  int num;  Scanner sc = new Scanner(System.in);  int n = sc.nextInt(); // Read the number of key-value pairs to be entered  for (int i = 0; i < n; i++) {  name = sc.next(); // Read the name (key)  num = sc.nextInt(); // Read the number (value)  map.put(name, num); // Add the key-value pair to the HashMap  }  // Printing key-value pairs from the first map  for (Entry<String, Integer> entry : map.entrySet()) {  System.out.println(entry.getKey() + " : " + entry.getValue());  }  System.out.println("----------"); // Separator line  // Creating another HashMap  HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();  // Inserting key-value pairs to anotherMap using put() method  anotherMap.put("SIX", 6);  anotherMap.put("SEVEN", 7);  // Inserting key-value pairs of map to anotherMap using putAll() method  anotherMap.putAll(map); // Add all entries from 'map' to 'anotherMap'  // Printing key-value pairs of anotherMap  for (Entry<String, Integer> entry : anotherMap.entrySet()) {  System.out.println(entry.getKey() + " : " + entry.getValue());  }  // Adds key-value pair 'FIVE'-5' only if it is not present in map  map.putIfAbsent("FIVE", 5);  // Retrieving a value associated with key 'TWO'  int value = map.get("TWO");  System.out.println(value);  // Checking whether key 'ONE' exists in map  System.out.println(map.containsKey("ONE"));  // Checking whether value '3' exists in map  System.out.println(map.containsValue(3));  // Retrieving the number of key-value pairs present in map  System.out.println(map.size());  sc.close(); // Close the scanner to prevent resource leaks  }  } |
| --- |

**OUTPUT:**



## Lab-12-Introduction to I/O, I/O Operations, Object Serialization

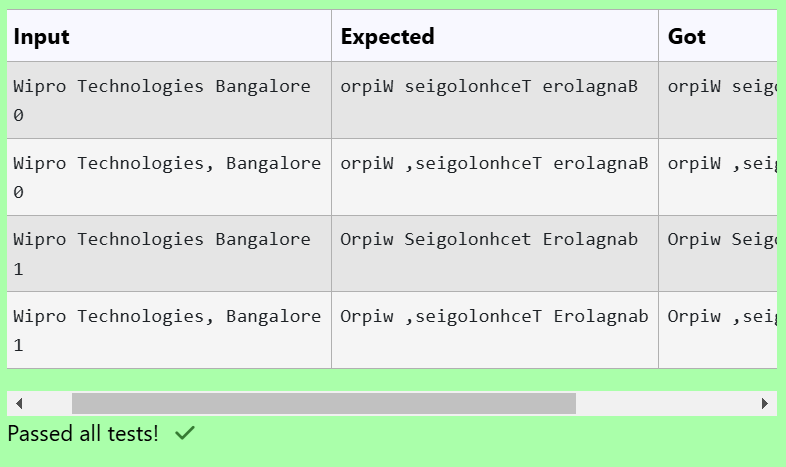
12.1



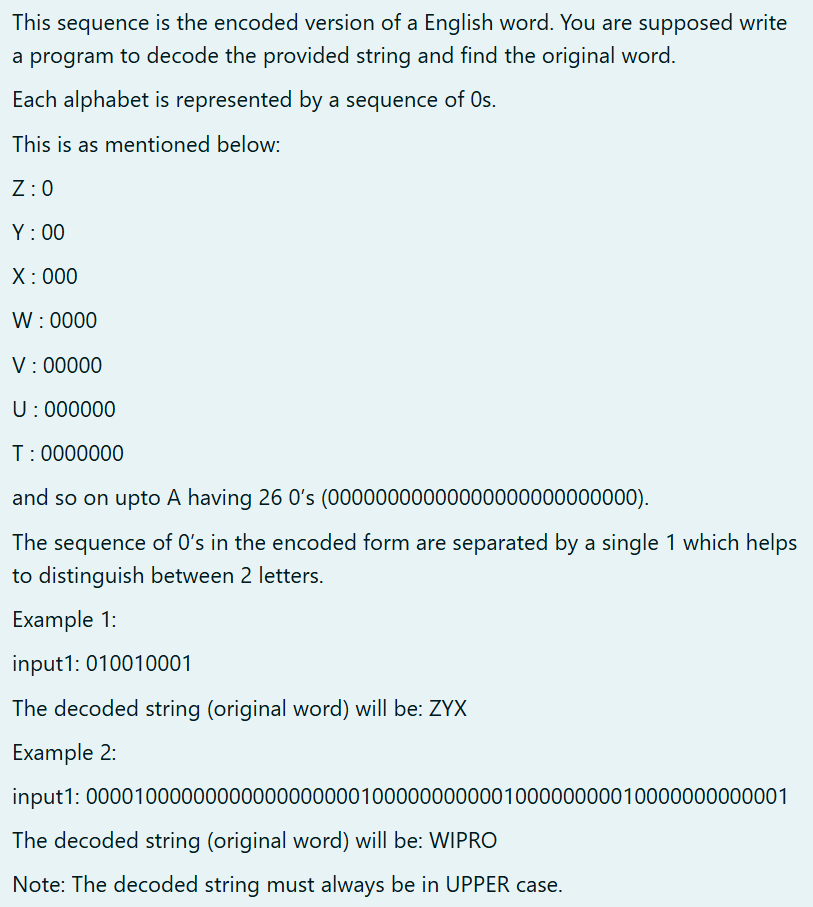
**PROGRAM:**

| import java.util.Scanner;  import java.util.Arrays;  public class ReverseWords {  public static String reverseWords(String sentence, int caseOption) {  String[] words = sentence.split("\\s+"); // Split by whitespace, including commas  StringBuilder reversedSentence = new StringBuilder();  for (String word : words) {  char[] chars = word.toCharArray();  char[] a=word.toCharArray();  int start = 0, end = chars.length - 1;  // Reverse word characters  while (start < end) {  char temp = chars[start];  chars[start] = chars[end];  chars[end] = temp;  start++;  end--;  }  int index=chars.length;  if (caseOption == 1) {  for (int i = 0; i < chars.length; i++) {  if (Character.isLetter(chars[i])&&i!=index) {  chars[i] = Character.isUpperCase(a[i]) ? Character.toUpperCase(chars[i]) : Character.toLowerCase(chars[i]);  }  else if(!Character.isLetter(chars[i])&&!Character.isWhitespace(chars[i])){  char c=a[i];  String d=String.valueOf(chars);  index=d.indexOf(c);  if(index!=-1)  chars[index]=a[i];    }      }  }  reversedSentence.append(String.valueOf(chars)).append(" ");  }  return reversedSentence.toString().trim();  }  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  String sentence = scanner.nextLine();  int caseOption = scanner.nextInt();  String reversedSentence = reverseWords(sentence, caseOption);  System.out.println(reversedSentence);  }  } |
| --- |

**OUTPUT:**



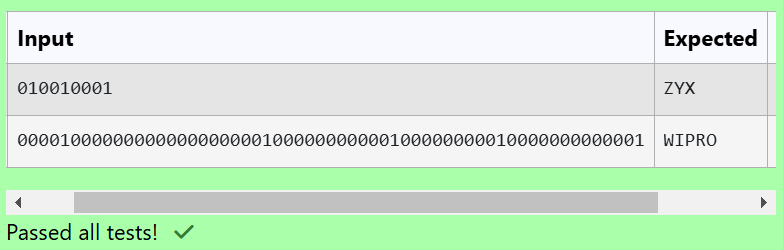
12.2



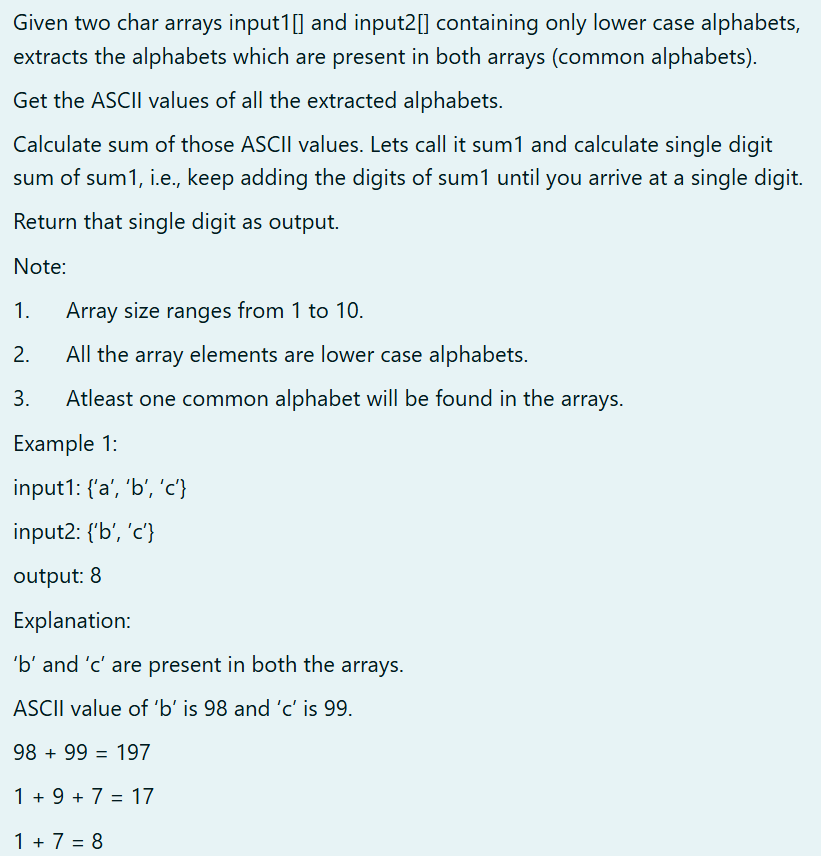
**PROGRAM:**

| import java.util.Scanner;  public class Decoder {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  String input = sc.nextLine(); // Read the input string of 1's and 0's  sc.close();  // Decode the string  String decodedWord = decode(input);  System.out.println(decodedWord);  }  private static String decode(String input) {  String[] segments = input.split("1"); // Split by '1'  StringBuilder decoded = new StringBuilder();  for (String segment : segments) {  int length = segment.length();  if (length > 0 && length <= 26) { // Ensure the length is valid  // Calculate the corresponding character  char letter = (char) ('Z' - (length - 1));  decoded.append(letter);  }  }  return decoded.toString();  }  } |
| --- |

**OUTPUT:**



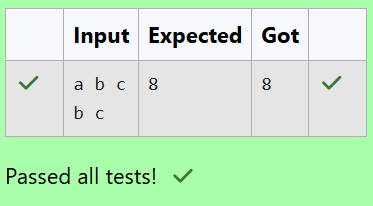
12.3



**PROGRAM:**

| import java.util.HashSet;  import java.util.Scanner;  public class CommonCharacterASCII {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);    String input1 = scanner.nextLine();  char[] charArray1 = input1.replace(" ", "").toCharArray(); // Convert to char array  // Prompt the user for the second input    String input2 = scanner.nextLine();  char[] charArray2 = input2.replace(" ", "").toCharArray(); // Convert to char array  // Calculate the result  int result = calculateSingleDigitSum(charArray1, charArray2);  System.out.println( result); // Display the result  scanner.close();  }  public static int calculateSingleDigitSum(char[] input1, char[] input2) {  // Step 1: Find common characters  HashSet<Character> commonChars = new HashSet<>();  for (char c1 : input1) {  for (char c2 : input2) {  if (c1 == c2) {  commonChars.add(c1);  }  }  }  // Step 2: Calculate the sum of ASCII values of common characters  int sum1 = 0;  for (char c : commonChars) {  sum1 += (int) c; // Get ASCII value  }  // Step 3: Calculate single digit sum  return getSingleDigit(sum1);  }  private static int getSingleDigit(int sum) {  while (sum >= 10) {  int tempSum = 0;  while (sum > 0) {  tempSum += sum % 10; // Add last digit  sum /= 10; // Remove last digit  }  sum = tempSum; // Update sum to the new sum of digits  }  return sum;  }  } |
| --- |

**OUTPUT:**

****

## 

## 

## 