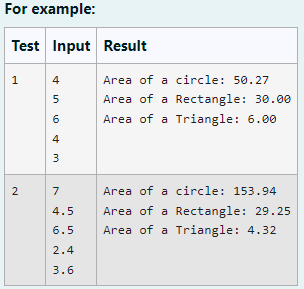
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WEEK-8\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ROLL NO:230701233

1. 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.



CODE:

import java.util.Scanner;

// Abstract class Shape abstract class Shape {

public abstract double calculateArea();

}

// Circle class

class Circle extends Shape { private double radius;

public Circle(double radius) {

this.radius = radius;

}

@Override

public double calculateArea() {

return Math.PI \* radius \* radius; // Area of circle: πr²

}

}

// Rectangle class

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; // Area of rectangle: length \* breadth

}

}

// Triangle class

class Triangle extends Shape { private double base;

private double height;

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

this.height = height;

}

@Override

public double calculateArea() {

return 0.5 \* base \* height; // Area of triangle: 0.5 \* base \* height

}

}

// Main class to test the shapes public class ShapeTest {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input for Circle

double radius = scanner.nextDouble(); Circle circle = new Circle(radius);

System.out.printf("Area of a circle: %.2f%n", circle.calculateArea());

// Input for Rectangle

double length = scanner.nextDouble();

double breadth = scanner.nextDouble();

Rectangle rectangle = new Rectangle(length, breadth);

System.out.printf("Area of a Rectangle: %.2f%n", rectangle.calculateArea());

// Input for Triangle

double base = scanner.nextDouble();

double height = scanner.nextDouble(); Triangle triangle = new Triangle(base, height);

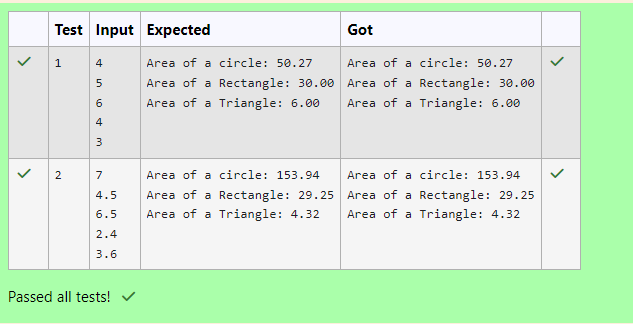
System.out.printf("Area of a Triangle: %.2f%n", triangle.calculateArea());

scanner.close();

}

}

OUTPUT:



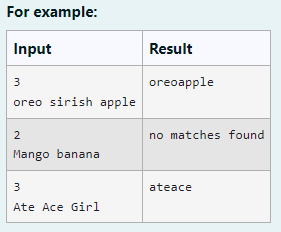
1. As a logic building learner you are given the task to extract the string which has vowel as the first and last characters from the given array of Strings.

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

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

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

input1: an integer representing the number of elements in the array. input2: String array.



Code:

import java.util.Scanner;

public class VowelStringExtractor {

// Method to extract strings with vowels as first and last characters public static String extractVowelStrings(String[] stringArray) {

StringBuilder result = new StringBuilder();

String vowels = "aeiouAEIOU"; // String containing all vowels

// Iterate through the array of strings for (String s : stringArray) {

// Check if the string is not empty and if both the first and last characters are vowels

if (s.length() > 0 && vowels.indexOf(s.charAt(0)) != -1 && vowels.indexOf(s.charAt(s.length() - 1)) != -1) {

result.append(s); // Append matching string to the result

}

}

// Return the concatenated string in lowercase or "no matches found"

return result.length() > 0 ? result.toString().toLowerCase() : "no matches found";

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input for the number of strings

int n = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

// Input for the strings in one line

String input = scanner.nextLine();

String[] strings = input.split(" "); // Split input into an array

// Process and output the result

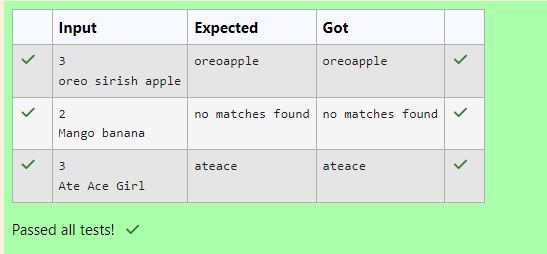
String result = extractVowelStrings(strings); System.out.println(result);

scanner.close(); // Close the scanner

}

}

Output:



# 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

# 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.");

}

# Final Class:

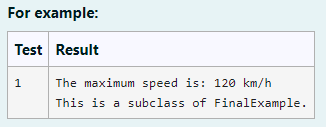
* + A class declared as final cannot be subclassed (i.e., no other class can inherit from it).
  + It is used to prevent a class from being extended and modified.
  + public final class Vehicle {

// class code

}

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

**you should delete any piece of code.**



Code:

final class FinalExample {

// Final variable

final int MAX\_SPEED = 120; // Constant value

// Final method

public final void display() {

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

}

}

// Main class to test the final class public class Test {

public static void main(String[] args) {

// Create an instance of FinalExample

FinalExample example = new FinalExample(); example.display();

// Uncommenting the following line will result in a compile-time error

// because FinalExample is a final class and cannot be subclassed.

// class SubclassExample extends FinalExample { }

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

}

}

Output:

