Dashboard / My courses / CS23333-OOPUJ-2023 / Lab-08 - Polymorphism, Abstract Classes, final Keyword / Lab-08-Logic Building

| | |
|---|---|
| **Status** | Finished |
| **Started** | Wednesday, 23 October 2024, 12:12 PM |
| **Completed** | Monday, 28 October 2024, 9:18 AM |
| **Duration** | 4 days 21 hours |

Question **1**

Correct

Marked out of 5.00

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

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

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

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

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

input2: String array.

Example 1:

input1: 3

input2: {“oreo”, “sirish”, “apple”}

output: oreoapple

Example 2:

input1: 2

input2: {“Mango”, “banana”}

output: no matches found

Explanation:

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

Hence the output is no matches found.

Example 3:

input1: 3

input2: {“Ate”, “Ace”, “Girl”}

output: ateace

**For example:**

| Input | Result |
|---|---|
| 3<br>oreo sirish apple | oreoapple |
| 2<br>Mango banana | no matches found |
| 3<br>Ate Ace Girl | ateace |

**Answer:** (penalty regime: 0 %)

```
import java.util.*;
class prog{

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

        String v= "aeiou";
        String temp;
        String result="";
        for (int i=0;i<t.length;i++){
            temp=t[i].toLowerCase();
            int f=0;
            for (int j=0;j<v.length();j++){
                if (Character.compare(temp.charAt(0),v.charAt(j))==0){
                    f=1;
                    break;
                }
            }
            if (f==1){
                result+=temp;
            }
        }
        if (result.compareTo("")==0){
            System.out.println("no matches found");
        }
        else{
            System.out.println(result);
        }

    }
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>oreo sirish apple | oreoapple | oreoapple | ✓ |
| ✓ | 2<br>Mango banana | no matches found | no matches found | ✓ |
| ✓ | 3<br>Ate Ace Girl | ateace | ateace | ✓ |

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

## 1. Final Variable:

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

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

## 2. Final Method:

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

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

## 3. Final Class:

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

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

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

**For example:**

| Test | Result |
|---|---|
| 1 | `The maximum speed is: 120 km/h`<br>`This is a subclass of FinalExample.` |

**Answer:** (penalty regime: 0 %)

Reset answer

```java
class FinalExample {

    // Final variable
        final int maxSpeed = 120;

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

class SubClass extends FinalExample {

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

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

class prog {
    public static void main(String[] args) {
        FinalExample obj = new FinalExample();
        obj.displayMaxSpeed();

        SubClass subObj = new SubClass();
        subObj.showDetails();
    }
}
```

```
}

```

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

Passed all tests! ✓

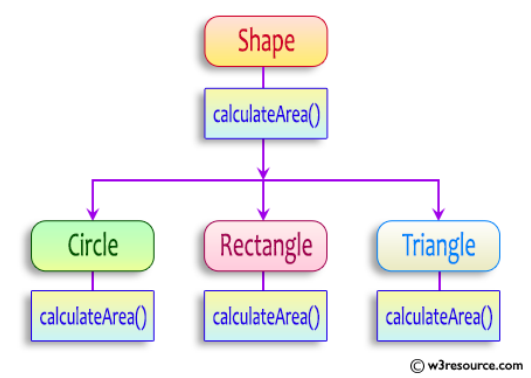
Question **3**

Correct

Marked out of 5.00

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

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



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

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

sample Input :

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

5  //  length of the rectangle

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

4   // base of the triangle

3  //  height of the triangle

**OUTPUT:**

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

**For example:**

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

**Answer:** (penalty regime: 0 %)

```
import java.util.*;

abstract class shape {
    public abstract double calculateArea();
}

class circle extends shape {
    float r;

    circle(float r) {
        this.r = r;
    }
    public double calculateArea() {
        return Math.PI * r * r;
    }
}

class rectangle extends shape {
    float l, b;

    rectangle(float l, float b) {
        this.l = l;
        this.b = b;
    }

    public double calculateArea() {
        return l * b;
    }
}

class triangle extends shape {
    float l, b;

    triangle(float l, float b) {
        this.l = l;
        this.b = b;
    }

    public double calculateArea() {
        return 0.5 * l * b;
    }
}

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

        float n1 = s.nextFloat();

        float n2 = s.nextFloat();
        float n3 = s.nextFloat();
```

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

Passed all tests! ✓

◄ Lab-08-MCQ

Jump to...

FindStringCode ►