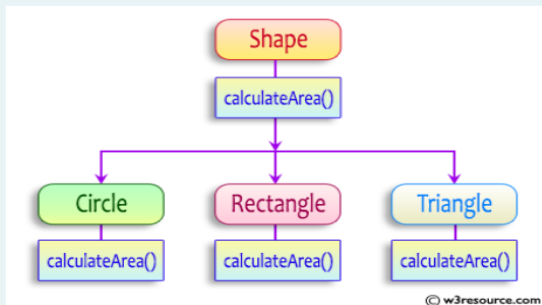


## WEEK 8: Polymorphism, abstract class, final key...

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

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



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

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

sample Input :

```
4 // radius of the circle to calculate area PI*r*r  
5 // length of the rectangle  
6 // breadth of the rectangle to calculate the area of a rectangle  
4 // base of the triangle  
3 // height of the triangle
```

### OUTPUT:

**Area of a circle :50.27**

**Area of a Rectangle :30.00**

**Area of a Triangle :6.00**

**For example:**

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

### PROGRAM:-

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

```
interface Shape{
```

```
    void calculate(double... a);
```

```
}
```

```
class Circle implements Shape{
```

```
    public void calculate(double... a){
```

```
        System.out.printf("Area of a circle: %.2f\n",(3.141632653061224*a[0]*a[0]));
```

```

    }
}

class Rectangle implements Shape{
    public void calculate(double... a){
        System.out.printf("Area of a Rectangle: %.2f\n", (a[0]*a[1]));
    }
}

class Triangle implements Shape{
    public void calculate(double... a){
        System.out.printf("Area of a Triangle: %.2f\n", (0.5*a[0]*a[1]));
    }
}

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

        Circle a=new Circle();

        Rectangle b=new Rectangle();

        Triangle c=new Triangle();

        a.calculate(sc.nextDouble());

        b.calculate(sc.nextDouble(),sc.nextDouble());

        c.calculate(sc.nextDouble(),sc.nextDouble());}}

```

	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! ✓

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

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

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

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

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

input2: String array.

Example 1:

input1: 3

input2: {"oreo", "sirish", "apple"}

output: oreoapple

Example 2:

input1: 2

input2: {"Mango", "banana"}

output: no matches found

Explanation:

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

Hence the output is no matches found.

Example 3:

input1: 3

input2: {"Ate", "Ace", "Girl"}

output: ateace

### For example:

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

### PROGRAM:-

```
import java.util.Scanner;

public class Main{

    public static void main(String[] args){

        Scanner sc=new Scanner(System.in);

        int a=sc.nextInt(),c=0;
```

```

sc.nextLine();

String []arr=sc.nextLine().split(" ");

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

    String w=arr[i].toLowerCase();

    char s1=w.charAt(0);

    char s2=w.charAt(arr[i].length()-1);

    int f1=0,f2=0;

    if(s1=='a' || s1=='e' || s1=='i' || s1=='o' || s1=='u') f1=1;

    if(s2=='a' || s2=='e' || s2=='i' || s2=='o' || s2=='u') f2=1;

    if(f1==1 && f2==1)System.out.print(w);

    else c++;

}if(c==a)System.out.println("no matches found");

}
}

```

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

Passed all tests! ✓

## 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. |

### PROGRAM:-

```
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! ✓