```java
public abstract class MathOp {

    private String opName;

    public MathOp(String op) {
        opName = op;
    }

    public abstract double result();

    public  String toString() {
        return opName;
    }
}
```

```java
public class Addition extends MathOp {

    private double op1;
    private double op2;

    public Addition(double a, double b) {
        super("Addition");
        op1 = a;
        op2 = b;
    }

    public double result() {
        return op1 + op2;
    }

    public String toString() {
        return super.toString() + ": " + op1 + " + " + op2 + " = " + result();
    }

}
```

```java
public class Subtraction extends MathOp {

    private double op1;
    private double op2;

    public Subtraction(double a, double b) {
        super("Subtraction");
        op1 = a;
        op2 = b;
    }

    public double result() {
        return op1 - op2;
    }

    public String toString() {
        return super.toString() + ": " + op1 + " - " + op2 + " = " + result();
    }

}
```

```java
public class Multiplication extends MathOp {

    private double op1;
    private double op2;

    public Multiplication(double a, double b) {
        super("Multiplication");
        op1 = a;
        op2 = b;
    }

    public double result() {
        return op1 * op2;
    }

    public String toString() {
        return super.toString() + ": " + op1 + " * " + op2 + " = " + result();
    }

}
```

```java
public class Division extends MathOp  {

    private double op1;
    private double op2;

    public Division(double a, double b) {
        super("Division");
        op1 = a;
        op2 = b;
    }

    public double result() {
        if (op2 != 0)
            return op1 / op2;
        else
            return -99999.99999;
    }

    public String toString() {
        if(op2 == 0)
            return "Error: Divide by zero";
        else
            return super.toString() + ": " + op1 + " / " + op2 + " = " + result();
    }

}
```

```java
public class TestMath {

    public static void main(String [] args) {
        MathOp [] m = new MathOp[4];

        m[0] = new Addition(4,3);
        m[1] = new Subtraction(2,6);
        m[2] = new Multiplication(3,5);
        m[3] = new Division(4,9);

        for (int i = 0; i < m.length; i++)
            if (!(m[i] instanceof Subtraction))
                System.out.println(m[i].toString());
    }

}
```

```java
public abstract class Shape {

    protected double x;

    public Shape(double y) {
        x = y;
    }

}
```

```java
public abstract class TwoDimensionalShape extends Shape {

    public TwoDimensionalShape(double x) {
        super(x);
    }

    public abstract double getArea();

}
```

```java
public abstract class ThreeDimensionalShape extends Shape {

    public ThreeDimensionalShape(double x) {
        super(x);
    }

    public abstract double getArea();

    public abstract double getVolume();

}
```

```java
public class Square extends TwoDimensionalShape {

    public Square(double x) {
        super(x);
    }

    public double getArea() {
        return x * x;
    }

}
```

```java
public class Circle extends TwoDimensionalShape {

    public Circle(double x) {
        super(x);
    }

    public double getArea() {
        return Math.PI * x * x;
    }

}
```

```java
public class Cube extends ThreeDimensionalShape {

    public Cube(double x) {
        super(x);
    }

    public double getArea() {
        return 6 * x * x;
    }

    public double getVolume() {
        return x * x * x;
    }

}
```

```java
public class Sphere extends ThreeDimensionalShape {
    public Sphere(double x) {
        super(x);
    }

    public double getArea() {
        return 4 * Math.PI * x * x;
    }

    public double getVolume() {
        return 4 / 3 * Math.PI * x * x * x;
    }

}
```

```
public class TestShape {

    public static void main(String [] args) {
        Shape [] s = new Shape[6];

        s[0] = new Circle(4);
        s[1] = new Cube(2);
        s[2] = new Sphere(3);
        s[3] = new Square(7);
        s[4] = new Circle(1);
        s[5] = new Sphere(4);

        for(int i = 0; i < s.length; i++) {
            if(s[i] instanceof TwoDimensionalShape) {
                TwoDimensionalShape x = (TwoDimensionalShape)s[i];

                System.out.println("This is 2D shape");
                System.out.println("Its area = " + x.getArea());
                System.out.println("--------------------");
            }
            else if(s[i] instanceof ThreeDimensionalShape) {
                ThreeDimensionalShape x = (ThreeDimensionalShape)s[i];

                System.out.println("This is 3D shape");
                System.out.println("Its area = " + x.getArea());
                System.out.println("Its volume = " + x.getVolume());
                System.out.println("--------------------");
            }
        }

    }

}
```