En cada uno de los siguientes programas, identifique si se usa el patrón adapter o no y justifique el porqué.

1. **Adaptación de calculadora avanzada a calculadora simple**

```java
class CalculadoraSimple {
    public double sumar(double a, double b) {
        return a + b;
    }

    public double restar(double a, double b) {
        return a - b;
    }

    public double multiplicar(double a, double b) {
        return a * b;
    }

    public double dividir(double a, double b) {
        if (b == 0) {
            throw new ArithmeticException("División por cero no permitida.");
        }
        return a / b;
    }
}
```

```java
class CalculadoraAvanzada {
    public double agregar(double a, double b) {
        return a + b;
    }

    public double sustraer(double a, double b) {
        return a - b;
    }

    public double producto(double a, double b) {
        return a * b;
    }

    public double cociente(double a, double b) {
        if (b == 0) {
            throw new ArithmeticException("División por cero no permitida.");
        }
        return a / b;
    }
}
```

```java
class AdaptadorCalculadoraAvanzada {
    private CalculadoraAvanzada calculadoraAvanzada;

    public AdaptadorCalculadoraAvanzada(CalculadoraAvanzada calculadoraAvanzada) {
        this.calculadoraAvanzada = calculadoraAvanzada;
    }

    public double sumar(double a, double b) {
        return calculadoraAvanzada.agregar(a, b);
    }

    public double restar(double a, double b) {
        return calculadoraAvanzada.sustraer(a, b);
    }

    public double multiplicar(double a, double b) {
        return calculadoraAvanzada.producto(a, b);
    }

    public double dividir(double a, double b) {
        return calculadoraAvanzada.cociente(a, b);
    }
}
```

2. Adaptación de cuadrado a circulo

```java
class Circulo {
    private double radio;

    public Circulo(double radio) {
        this.radio = radio;
    }

    public double getRadio() {
        return radio;
    }
}
```

```java
class CalculadoraAreaCirculo {
    // Calcula el área de un círculo usando la fórmula: π * radio^2
    public double calcularArea(Circulo circulo) {
        return Math.PI * Math.pow(circulo.getRadio(), 2);
    }
}
```

```java
class Cuadrado {
    private double lado;

    public Cuadrado(double lado) {
        this.lado = lado;
    }

    public double getLado() {
        return lado;
    }
}
```

```java
class CuadradoAdapter extends Circulo {
    private Cuadrado cuadrado;

    public CuadradoAdapter(Cuadrado cuadrado) {
        // Calculamos el radio equivalente para un cuadrado que quepa en un círculo
        super(cuadrado.getLado() / Math.sqrt(2));
        this.cuadrado = cuadrado;
    }

    // Override opcional para explicar la conversión (no es necesario pero es ilustrativo)
    @Override
    public double getRadio() {
        return cuadrado.getLado() / Math.sqrt(2);
    }
}
```

3. Adaptación de calculadora avanzada a calculadora básica

```java
class CalculadoraBasica {
    public double sumar(double a, double b) {
        return a + b;
    }

    public double multiplicar(double a, double b) {
        return a * b;
    }
}
```

```java
class CalculadoraAvanzada {
    public double potencia(double base, double exponente) {
        return Math.pow(base, exponente);
    }

    public double raizCuadrada(double numero) {
        return Math.sqrt(numero);
    }
}
```

```java
class CalculadoraAvanzadaAdapter {
    private CalculadoraAvanzada calculadoraAvanzada;

    public CalculadoraAvanzadaAdapter(CalculadoraAvanzada calculadoraAvanzada) {
        this.calculadoraAvanzada = calculadoraAvanzada;
    }

    public double sumar(double base, double exponente) {
        return calculadoraAvanzada.potencia(base, 1) + calculadoraAvanzada.potencia(exponente, 1);
    }

    public double multiplicar(double base, double exponente) {
        return calculadoraAvanzada.potencia(base, exponente);
    }

    public double raizCuadrada(double numero) {
        return calculadoraAvanzada.raizCuadrada(numero);
    }
}
```

4. Adaptación de 220v a 5v

```java
// Clase que espera 5V para funcionar
public class Cargador5V {
    public void cargarDispositivo() {
        System.out.println("Cargando dispositivo con 5V...");
    }
}
```

```java
// Clase que representa un enchufe de 220V
public class Enchufe220V {
    public int obtenerVoltaje() {
        System.out.println("Voltaje entregado por el enchufe: 220V");
        return 220;
    }
}
```

```java
// Adapter que adapta el voltaje de 220V a 5V
public class AdaptadorVoltaje extends Cargador5V {
    private Enchufe220V enchufe220V;

    public AdaptadorVoltaje(Enchufe220V enchufe220V) {
        this.enchufe220V = enchufe220V;
    }

    // Sobrescribimos el método para adaptar el comportamiento
    @Override
    public void cargarDispositivo() {
        int voltaje = enchufe220V.obtenerVoltaje();
        if (voltaje > 5) {
            System.out.println("Adaptando voltaje de 220V a 5V...");
        }
        super.cargarDispositivo();
    }
}
```