Let's create decorators for the basic arithmetic operations: addition, subtraction, multiplication, and division of two numbers. Each decorator will log the operation being performed along with the operands and the result.

1. Addition Decorator:
```typescript
function LogAddition(target: any, methodName: string, descriptor: PropertyDescriptor) {
    const originalMethod = descriptor.value;

    descriptor.value = function(a: number, b: number): number {
        const result = originalMethod.apply(this, [a, b]);
        console.log(`Addition: ${a} + ${b} = ${result}`);
        return result;
    };

    return descriptor;
}

class Calculator {
    @LogAddition
    add(a: number, b: number): number {
        return a + b;
    }
}

const calculator = new Calculator();
calculator.add(3, 5);

// Output:
// Addition: 3 + 5 = 8
```

2. Subtraction Decorator:
```typescript
function LogSubtraction(target: any, methodName: string, descriptor: PropertyDescriptor) {
    const originalMethod = descriptor.value;

    descriptor.value = function(a: number, b: number): number {
        const result = originalMethod.apply(this, [a, b]);
        console.log(`Subtraction: ${a} - ${b} = ${result}`);
        return result;
    };

    return descriptor;
```

```
}

class Calculator {
    @LogSubtraction
    subtract(a: number, b: number): number {
        return a - b;
    }
}

const calculator = new Calculator();
calculator.subtract(7, 4);

// Output:
// Subtraction: 7 - 4 = 3
```

3. Multiplication Decorator:
```typescript
function LogMultiplication(target: any, methodName: string, descriptor: PropertyDescriptor) {
    const originalMethod = descriptor.value;

    descriptor.value = function(a: number, b: number): number {
        const result = originalMethod.apply(this, [a, b]);
        console.log(`Multiplication: ${a} * ${b} = ${result}`);
        return result;
    };

    return descriptor;
}

class Calculator {
    @LogMultiplication
    multiply(a: number, b: number): number {
        return a * b;
    }
}

const calculator = new Calculator();
calculator.multiply(4, 6);

// Output:
// Multiplication: 4 * 6 = 24
```

4. Division Decorator:
```typescript
function LogDivision(target: any, methodName: string, descriptor: PropertyDescriptor) {
    const originalMethod = descriptor.value;

    descriptor.value = function(a: number, b: number): number {
        if (b === 0) {
            throw new Error("Division by zero is not allowed.");
        }
        const result = originalMethod.apply(this, [a, b]);
        console.log(`Division: ${a} / ${b} = ${result}`);
        return result;
    };

    return descriptor;
}

class Calculator {
    @LogDivision
    divide(a: number, b: number): number {
        return a / b;
    }
}

const calculator = new Calculator();
calculator.divide(10, 2);

// Output:
// Division: 10 / 2 = 5
```

In each example, we define a class `Calculator` with methods for performing arithmetic operations. Each method is decorated with a corresponding decorator (`LogAddition`, `LogSubtraction`, `LogMultiplication`, and `LogDivision`). When the methods are called, the decorators log the operation being performed along with the operands and the result. This demonstrates how decorators can be used to add behavior or functionality to methods in TypeScript.