Lab 05.2: Creating Calculator Classes

Objective

This lab will introduce the first of the larger systems that you will be programming. This system is a set of classes that perform basic math-oriented calculations. You will utilize the existing Java and **Math** class functionality to implement these classes and it will provide practice in creating classes, methods and attributes.

Overview

In this lab you will:

- Create the necessary packages
- Create the necessary calculator classes
- Add the appropriate attributes
- Provide the appropriate method definitions (minus the actual code to make them work at this point)

Step by Step Instructions

Create a Java Project and Package

1. Create a new Java Project within the same workspace named **Calculators**.

2. Create a package named **com.javaoo.calculators**.

Exercise 1: Creating the Calculator classes

3. Create each of the following classes in the **com.javaoo.calculators** package:
    a. **BasicCalculator**
    b. **ScientificCalculator**
    c. **TrigonometricCalculator**

Exercise 2: Adding Attributes and Methods

4. For each of the calculator classes, add the necessary attributes and methods as specified below.

5. **BasicCalculator**:

    a. Declare the following **public** methods
        i. `add()`
        ii. `subtract()`
        iii. `multiply()`
        iv. `divide()`
    b. Each method must accept two parameters, both of type `double`

    c. Each method must return a `double`. Add code to the return statement that calculates and returns the correct value. Example:

```
public final double multiply(double x, double y) {
        return x * y;
}
```

6. **ScientificCalculator**:

    d. Must declare the following attributes:
        i. A `double` named `PI` to hold the value of **PI (3.14159).** This attribute will be shared by all instances of the class and will be constant so it should be declared as `static` and `final`. Since it is a final, it can have public visibility so that anyone can use it.
        ii. A `double` named `holdValue` to hold a value in memory. Since it will be managed within the **ScientificCalculator** class, it should have `private` visibility.

    e. Declare the following **public** methods
        i. `exp()` which has one parameter of type `double` and returns a `double` [This method will be used to calculate e^x]
        ii. `log()` which has one parameter of type `double` and returns a `double` [This method will be used to calculate $\ln x$]
        iii. `putValueInMemory()` which has one parameter of type `double` and returns a `void`. Implement this method.
        iv. `getValueFromMemory()` which has no parameters and returns a `double` Implement this method

f. We will not provide the details of each method in this lab. In order for your code to compile, add the following single statement to **each method block that returns a `double`**:

```
return(0);
```

7. **TrigonometricCalculator**:

g. Declare the following **public** methods

```
i.sine()
ii.cosine()
iii.tangent()
iv.arcsine()
v.arccosine()
vi.arctangent()
```

h. Each method must accept one parameter of type `double`

i. Each method must return a `double`

j. We will not provide the details of each method in this lab. For your code to compile, add the following single statement to each method block:

```
return(0);
```

8. **BRAIN TEASER: Could and should any of these classes or methods be made *static*?**