CS102- Algorithms and Programming II

Lab 01

Lab Objectives: Arrays

Notes:

- For all labs in CS 102, your solutions must conform to these CS101/102 style guidelines.
- Create a Java Project named Lab01. Put all of your classes in this project.
- Remember to include **javadoc comments** for each class and method.
- Upload your solution **as a single .zip file** to the Lab01 assignment for your section by the end of your section's lab session on the week of February 8. You must use the following naming convention: Lab01_Surname_FirstName.zip where Surname is your family name and FirstName is your first name. You may upload multiple times; the last upload will be considered.

Question In this lab, you are going to implement a **`Polynomial`** class that represents polynomials of the form $P(x) = c_0 + c_1x + c_2x^2 + ... + c_nx^n$

The class should do the following:

1. `Polynomial` class should contain its coefficients in an array. Use double type for coefficients.
2. Include a constructor that takes an integer, `d`, and a double, `c`, to construct polynomials of the form $P(x) = cx^d$.

   Include a default constructor that takes no argument and constructs a zero polynomial (`P(x) = 0`).

3. Include another constructor that takes an array of coefficients and produces a polynomial with these coefficients.
4. Add a getter method for a coefficient which takes degree and returns the coefficient of the term with that degree.
5. Include **`getDegree()`** method that returns the degree of the polynomial. Degree of a polynomial is the degree of highest non-zero term in a polynomial. For example, the degree of polynomial $P(x) = 4 - 5x^2 + 12x^3$ is 3. You can assume that the degree of zero polynomial is 0.
6. Add **`toString()`** method that returns `String` representation of the polynomial. Zero terms in the polynomial should not be included in the string.

   For instance, for the polynomial $P(x) = 4 - 5x^2 + 2x^3$, `toString()` method should return "`4.0 - 5.0x^2 + 2.0x^3`".

7. Add **`eval( double x )`** method that evaluates the polynomial at `x` and returns the result.
   1. Use `Math.pow( double a, double b )` method to evaluate each term individually and the polynomial as a sum of the terms.

2. Implement another method, **`eval2( double x )`** that evaluates the polynomial using Horner's method. Horner's method is an efficient way of evaluating polynomials at a given point. A polynomial $P(x) = c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$ can be evaluated at x_0 by rearranging computation as $P(x_0) = ((\ldots ((c_n) x_0 + c_{n-1}) x_0 \ldots + c_4) x_0 + c_3) x_0 + c_1) x_0 + c_0$ and computing the result from the innermost parentheses to outwards.

8. Implement a class called **`PolynomialTester`** to test your `Polynomial` class.