Project 1. Efficient Polynomial Evaluations

(Early Line: 7/5/Mon, Deadline: 7/11/Sun)

Description:

In this project, we will compare three different algorithms that are used to *evaluate polynomials*. The goal is to *understand the importance of the efficiency of an algorithm*. The first two algorithms are based on the *straightforward method*, but they use different ways to evaluate the monomials. The third algorithm uses the *Horner's Rule* to evaluate polynomials. We will use random number generating functions to create large polynomials to be evaluated by these three algorithms, and compare their response times.

Requirements:

1. (*Format of your program*)

   It is desirable to use Java as the programming language. But if you use C/C++ or Python, it is still acceptable. Try to use one program to implement all three algorithms and do testing in the same program. When you submit your project, place your program or programs in one folder with your name and `Proj1, P1, Project1, etc.` as the folder name, then zip your folder and submit it.

   If you use Java, try to put your program in the package: `edu.umsl.algorithms`. In this way, it is more convenient for me to run your programs.

2. (*Generate the polynomial coefficients and the point to be evaluated*)

   Each coefficient should be an integer between 1 and 100. You use an integer array to store all the coefficients generated, and the size of the array is determined by the degree of the polynomial. Based on my experiments, a good choice of the degree number should be between $100,000$ and $200,000$.

   The value of the variable x should be a `double` between `0.0` and `1.0`, exclusive. The reason to choose the value in this range is simple: We can easily avoid the *overflow* problem. If the value is `0.0` or `1.0`, the evaluation time is very fast, and it does not reflect the general case. When you detect if this case happens, call the

random function another time.

3. (*User's interface*)

   In order to provide the users a convenient way to test your program, you should create a simple *user's interface*. When a user runs your program, you display a simple menu by printing out appropriate messages on the screen. Your program asks the user to choose one of the three methods to evaluate the polynomial using `1, 2,` or `3` to represent the selected method. Put your testing part in a `while` loop, so that the user can test all the methods in one execution, and use `0` for the exit code.

4. (*Monomial evaluation*)

   In the first two methods, we use the *straightforward* method to evaluate the polynomials, in which we need to evaluate the monomials x^k. Our first algorithm just uses direct multiplications in a for-loop. Our second algorithm uses a power function, such as `Math.pow(x, k)`, to calculate the monomials.

5. (*Horner's Rule*)

   Our third algorithm uses the *Horner's Rule* to evaluate the whole polynomial without calculating the monomials. The main objective of this project is to see how much faster this algorithm is comparing with the first two simple algorithms.

6. (*Output*)

   Before you display the menu on the screen, you print out the degree number of the polynomial first, because this number only needs to be displayed once. Then for each algorithm, you need to display the result first, and then the execution time in milliseconds. We expect that all three algorithms produce the same evaluation result. After the user enters `0` code to terminate the program, display a (`Bye!`) message to tell the user that the whole program completes the execution.

Any updates about this project will be posted in Canvas as announcements.