Lab 1: Polynomial

Zheran Fang

22 Sep 2014

This lab is designed to give you practice working with linked lists. This is an individual assignment; you may not share code with other students.

Specification

Your task is to implement a polynomial class that uses a linked list to store the polynomial's terms. Each node of the list holds the coefficient and exponent for one term. The terms are kept in order from the largest to the smallest exponent.

In addition, you are required to implement the `toString()` method of the polynomial class to provide a more natural representation, as well as the operation to add two polynomials.

Term data structure. To model a single term in a polynomial, create a class `Term` with the following API:

```
public class Term {
    // create a term
    public Term(double coef, int exp)

    // get the value of the coefficient
    public double getCoefficient()

    // set the value of the coefficient
    public void setCoefficient(double coef)

    // get the value of the exponent
    public int getExponent()

    // set the value of the exponent
    public void setExponent(int exp)
```

```
    // get the following term
    public Term getNext()

    // set the following term
    public void setNext(Term next)
}
```

Polynomial data structure. To model a polynomial, create a class `Polynomial` with the following API:

```
public class Polynomial {
    // create a polynomial
    public Polynomial(Term firstTerm)

    // get the first term
    public Term getFirst()

    // set the first term
    public void setFirst(Term first)

    // add a single term to the polynomial
    public void addTerm(Term term)

    // add another polynomial, return the sum
    public Polynomial add(Polynomial another)

    // convert to string representation
    // example: 4.0x^3+3.2x^2-2.1x^1+1.0x^0
    // example: -12.0x^9-1.0x^7+3.0x^5+10.0x^2+5.0x^0
    public String toString()

    // write your own code to test your implementation
    public static void main(String[] args)
}
```

Throw a `NullPointerException` if the client attempts to add a null item with the polynomial or add a null term to the polynomial.

Analysis of running time and memory usage (optional and not graded). If the polynomials have `m` and `n` terms respectively, what is the time and space complexity of your implantation to add the two polynomials?

Submission

Create a zip file named *YourStudentID*.zip that contains your code project and upload your zip file to the FTP server.

You should implement *your own* linked list data structure and may not call any library functions other than those in `java.lang` in this lab.

Deadline

22 Sep 2014 17:00 GMT+08:00