Step 1 would be to set our “Polynomial.java”, i.e. **Polynomial** class - as a linked list of our coefficients of the polynomial representation. The data container storage would be in decreasing power of the polynomial coefficients.

**import java.util.\*;**

**public class Polynomial**

**{**

**private LinkedList<Integer> coefficients;**

To proceed, we will set the constructors that represent the polynomial data for the program. A default constructor, a constructor to set 0 coefficients with the size passed, and a constructor that sets on the array. The last constructor would be handy when we work with data values from file reading, or command line input.

**public Polynomial()**

**{**

**coefficients = new LinkedList<Integer>();**

**}**

**public Polynomial(int size)**

**{**

**coefficients = new LinkedList<Integer>();**

**for (int p=0; p<=size; p++)**

**coefficients.addLast(0);**

**}**

**public Polynomial(int[] coeffs)**

**{**

**coefficients = new LinkedList<Integer>();**

**for (int i = 0; i<coeffs.length; i++)**

**coefficients.addFirst(coeffs[i]);**

**}**

The next part, can be a few methods we need for the overall designing process. Like the polynomial size :  
  
  
 **public int getSize()**

**{**

**return coefficients.size()-1;**

**}**

And, a method to see if the polynomial is 0

**private boolean isZero()**

**{**

**for (int i = getSize(); i>=0; i--)**

**{**

**if (coefficients.get(i) != 0)**

**return false;**

**}**

**return true;**

**}**

Now the 2 methods which output the polynomial in decreasing, and increasing order

**public void outputDecreasingOrder()**

**{**

**int pow = getSize();**

**String output = "";**

**while (pow >= 0)**

**{**

**if (coefficients.get(pow) == 0);**

**else**

**{**

**if (coefficients.get(pow) >= 0)**

**output += " + ";**

**else**

**output += " - ";**

**if (pow == 0)**

**output += (Math.abs(coefficients.get(pow)));**

**else if (pow == 1)**

**output += (Math.abs(coefficients.get(pow)) + "X");**

**else**

**output += (Math.abs(coefficients.get(pow)) + "X^" + pow);**

**}**

**pow--;**

**}**

**if (isZero())**

**output = "0";**

**else if (output.charAt(1) != '-')**

**output = output.substring(3, output.length());**

**else**

**output = output.substring(1, output.length());**

**System.out.println(output);**

**}**

**public void outputIncreasingOrder()**

**{**

**int pow = 0;**

**String output = "";**

**while (pow <= getSize())**

**{**

**if (coefficients.get(pow) == 0);**

**else**

**{**

**if (coefficients.get(pow) >= 0)**

**output += " + ";**

**else**

**output += " - ";**

**if (pow == 0)**

**output += (Math.abs(coefficients.get(pow)));**

**else if (pow == 1)**

**output += (Math.abs(coefficients.get(pow)) + "X");**

**else**

**output += (Math.abs(coefficients.get(pow)) + "X^" + pow);**

**}**

**pow++;**

**}**

**if (isZero())**

**output = "0";**

**else if (output.charAt(1) != '-')**

**output = output.substring(3, output.length());**

**else**

**output = output.substring(1, output.length());**

**System.out.println(output);**

**}**

**public double evaluate(double x)**

**{**

**int pow = getSize();**

**double result = 0.0;**

**while (pow >= 0)**

**{**

**if (coefficients.get(pow) != 0);**

**{**

**result += (coefficients.get(pow) \* Math.pow(x, pow));**

**}**

**pow--;**

**}**

**return result;**

**}**

Also the method to “evaluate” is added above.

Now the 3 arithmetic methods (for addition, subtraction, and multiplication). The multiplication is with a scalar, the other 2 are on polynomials :

**public static Polynomial sum(Polynomial poly1, Polynomial poly2)**

**{**

**Polynomial res = new Polynomial();**

**int size1 = poly1.getSize();**

**int size2 = poly2.getSize();**

**int index1 = 0;**

**while (size1 > size2)**

**{**

**res.coefficients.addLast(poly1.coefficients.get(index1));**

**index1++;**

**size1--;**

**}**

**int index2 = 0;**

**while (size2 > size1)**

**{**

**res.coefficients.addLast(poly2.coefficients.get(index2));**

**index2++;**

**size2--;**

**}**

**size1 = poly1.getSize();**

**while (index1 <= size1)**

**{**

**res.coefficients.addLast(poly1.coefficients.get(index1) + poly2.coefficients.get(index2));**

**index1++;**

**index2++;**

**}**

**return res;**

**}**

**public static Polynomial difference(Polynomial poly1, Polynomial poly2)**

**{**

**Polynomial res = new Polynomial();**

**int size1 = poly1.getSize();**

**int size2 = poly2.getSize();**

**int index1 = 0;**

**while (size1 > size2)**

**{**

**res.coefficients.addLast(poly1.coefficients.get(index1));**

**index1++;**

**size1--;**

**}**

**int index2 = 0;**

**while (size2 > size1)**

**{**

**res.coefficients.addLast(-1 \* poly2.coefficients.get(index2));**

**index2++;**

**size2--;**

**}**

**size1 = poly1.getSize();**

**while (index1 <= size1)**

**{**

**res.coefficients.addLast(poly1.coefficients.get(index1) - poly2.coefficients.get(index2));**

**index1++;**

**index2++;**

**}**

**return res;**

**}**

**public static Polynomial multiply(Polynomial poly, int val)**

**{**

**Polynomial res = new Polynomial();**

**int size = poly.getSize();**

**int index = 0;**

**while (index <= size)**

**{**

**res.coefficients.addLast(val \* poly.coefficients.get(index));**

**index++;**

**}**

**return res;**

**}**

These methods I have made as static for direct class on our driver program

That completes this class

**}**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now, the **PolynomialList.java**. Here, we will require to define our array list of polynomials, and the operations relevant to the polynomial functioning.

**import java.util.\*;**

**import java.io.\*;**

**public class PolynomialList**

**{**

The key data that we need to maintain is an ArrayList of polynomials

**private static ArrayList<Polynomial> polynomials = new ArrayList<Polynomial>();**

**private static Scanner input = new Scanner(System.in);**

Also added the user input scanner, since it will be used throughout the program.

The simple main() method skeleton

**public static void main(String args[])**

**{**

**args = new String[]{"4", "0", "5", "7", "0", "3"};**

**int choice = -1;**

**do**

**{**

**menu();**

**System.out.print("Pick your choice : ");**

**choice = input.nextInt();**

**switch (choice)**

**{**

**case 1: createNoTerms(); break;**

**case 2: createReadFile(); break;**

**case 3: createReadCommandLine(args); break;**

**case 4: printAllIncreasing(); break;**

**case 5: printAllDecreasing(); break;**

**case 6: addPolynomials(); break;**

**case 7: substractPolynomials(); break;**

**case 8: multiplyPolynomial(); break;**

**case 9: evaluatePolynomial(); break;**

**case 0: System.out.println("Bye bye!!"); break;**

**default: System.out.println("Wrong choice, try again");**

**}**

**System.out.println();**

**} while (choice != 0);**

**}**

This show all the methods that we are looking at to define for the overall functioning of the program.

A neatly formatted user interface needs a MENU

**public static void menu()**

**{**

**System.out.println("1. Create no terms polynomial");**

**System.out.println("2. Create a polynomial by reading from file");**

**System.out.println("3. Create a polynomial by reading from command line");**

**System.out.println("4. Print all polynomials in Increasing Order");**

**System.out.println("5. Print all polynomials in Decreasing Order");**

**System.out.println("6. Add 2 Polynomials");**

**System.out.println("7. Substract 2 Polynomials");**

**System.out.println("8. Multiply a Polynomail with a scalar value");**

**System.out.println("9. Evalute a polynomial");**

**System.out.println("0. EXIT");**

**}**

Then, the first 3 methods of creation of polynomials

**public static void createNoTerms()**

**{**

**Polynomial poly = new Polynomial(0);**

**polynomials.add(poly);**

**}**

**public static void createReadFile()**

**{**

**System.out.print("Enter file to read from : ");**

**String fileName = input.next();**

**try**

**{**

**Scanner file = new Scanner(new File(fileName));**

**String[] values = new String[100];**

**int index = 0;**

**while (file.hasNext())**

**{**

**values[index] = file.next();**

**index++;**

**}**

**int[] array = new int[index];**

**for (int i=0; i<index; i++)**

**array[i] = Integer.parseInt(values[i]);**

**Polynomial poly = new Polynomial(array);**

**polynomials.add(poly);**

**}**

**catch (FileNotFoundException e)**

**{**

**System.out.println("Oops. Error opening file. Please try again.");**

**}**

**}**

**public static void createReadCommandLine(String[] args)**

**{**

**if (args.length == 0)**

**{**

**System.out.println("No valid argument having a polynomial specified.");**

**return;**

**}**

**int[] array = new int[args.length];**

**for (int i=0; i<args.length; i++)**

**array[i] = Integer.parseInt(args[i]);**

**Polynomial poly = new Polynomial(array);**

**polynomials.add(poly);**

**}**

Next, the printing methods

**public static void printAllIncreasing()**

**{**

**for (int i=0; i<polynomials.size(); i++)**

**{**

**System.out.print("Polynomial " + i + " : ");**

**polynomials.get(i).outputIncreasingOrder();**

**}**

**}**

**public static void printAllDecreasing()**

**{**

**for (int i=0; i<polynomials.size(); i++)**

**{**

**System.out.print("Polynomial " + i + " : ");**

**polynomials.get(i).outputDecreasingOrder();**

**}**

**}**

Now methods for the arithmetic

**public static void addPolynomials()**

**{**

**printAllDecreasing();**

**System.out.print("Select a polynomial: ");**

**int polyNum1 = input.nextInt();**

**while (polyNum1 < 0 || polyNum1 >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum1 = input.nextInt();**

**}**

**System.out.print("Select another polynomial: ");**

**int polyNum2 = input.nextInt();**

**while (polyNum2 < 0 || polyNum2 >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum2 = input.nextInt();**

**}**

**Polynomial res = Polynomial.sum(polynomials.get(polyNum1), polynomials.get(polyNum2));**

**System.out.print("Added polynomial on the ArrayList : ");**

**res.outputDecreasingOrder();**

**polynomials.add(res);**

**}**

**public static void substractPolynomials()**

**{**

**printAllDecreasing();**

**System.out.print("Select a polynomial: ");**

**int polyNum1 = input.nextInt();**

**while (polyNum1 < 0 || polyNum1 >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum1 = input.nextInt();**

**}**

**System.out.print("Select another polynomial: ");**

**int polyNum2 = input.nextInt();**

**while (polyNum2 < 0 || polyNum2 >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum2 = input.nextInt();**

**}**

**Polynomial res = Polynomial.difference(polynomials.get(polyNum1), polynomials.get(polyNum2));**

**System.out.print("Added polynomial on the ArrayList : ");**

**res.outputDecreasingOrder();**

**polynomials.add(res);**

**}**

**public static void multiplyPolynomial()**

**{**

**printAllDecreasing();**

**System.out.print("Select a polynomial: ");**

**int polyNum = input.nextInt();**

**while (polyNum < 0 || polyNum >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum = input.nextInt();**

**}**

**System.out.print("Enter a scalar value to multiply with: ");**

**int val = input.nextInt();**

**Polynomial res = Polynomial.multiply(polynomials.get(polyNum), val);**

**System.out.print("Added polynomial on the ArrayList : ");**

**res.outputDecreasingOrder();**

**polynomials.add(res);**

**}**

And, the last one for the “evaluate”

**public static void evaluatePolynomial()**

**{**

**printAllDecreasing();**

**System.out.print("Select a polynomial: ");**

**int polyNum = input.nextInt();**

**while (polyNum < 0 || polyNum >= polynomials.size())**

**{**

**System.out.print("Kindly select a valid polynomial number: ");**

**polyNum = input.nextInt();**

**}**

**System.out.print("Enter the value of x: ");**

**int x = input.nextInt();**

**double res = polynomials.get(polyNum).evaluate(x);**

**System.out.println("Result = " + res);**

**}**

**}**

That completes it all.

SAMPLE OUTPUT:













