Task

Implement the polynomial type. Represent a polynomial as a sequence of its real-valued coefficients. Implement as methods: adding two polynomials, multiplying two polynomials, evaluating the polynomial (substituting a value to the variable).

Polynomial type

Set of values

Integer degree;

Vector vec;

Operations

1. Getting Degree

Getting the degree of any given polynomial. As there can not be a polynomial with negative degree so it returns only positive values.

Formally:

A = (p:Poly,d:Int)

Pre : (p=p`)

Post : (d = p.getdegree)

2. Setting degree

Setting the degree of a polynomial to some user given number. It can not set the degree of a polynomial to a negative value.

Formally:

A = (p:Poly,n:Int)

Pre = (p=p` & n>=0)

Post = (p.setdegree(n))

1. Adding Polynomials

The sum of two polynomials is a new polynomial, which will have the degree of the polynomial with higher degree among the two. Difference in the degrees of the polynomials does not hinder the addition of polynomials.

Formally:

A: (p,q,r:Poly)

Pre: (p=p` & q = q`)

Post: (r = p+q)

2. Multiplying Polynomials

Multiplication of two polynomials results in a new polynomial with the degree equal to the sum of the original polynomials. Two polynomials with different degrees can be multiplied too.

Formally:

A: (p,q,r:Poly)

Pre: (p=p` & q = q`)

Post: (r = p*q)

3. Evaluating a Polynomial

Evaluating a polynomial gives back a number. The value given by the user is put into the variable and result is calculated.

Formally:

A: (p:Poly,val,result:Real)

Pre : (p=p` & val = val`)

Post : (r = p.eval(val))

Representation:

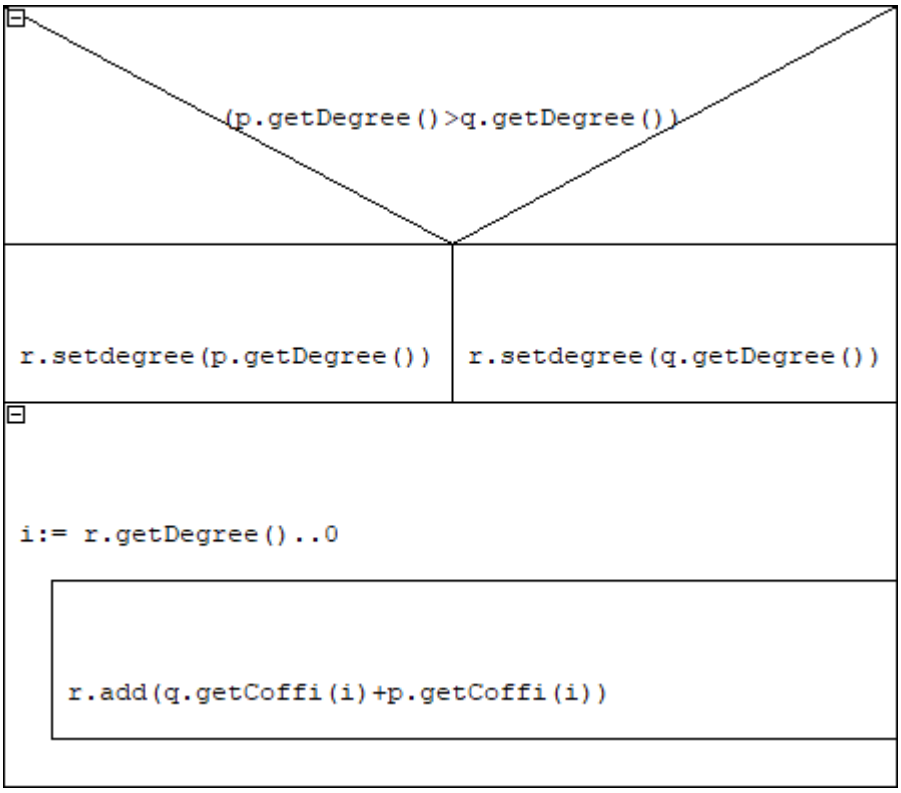
We need an integer degree to store the degree of the polynomial and a vector to store the coefficients of the variables.

ax^2 + bx + c ←→ vec = <a,b,c> and degree = 2

Implementation:

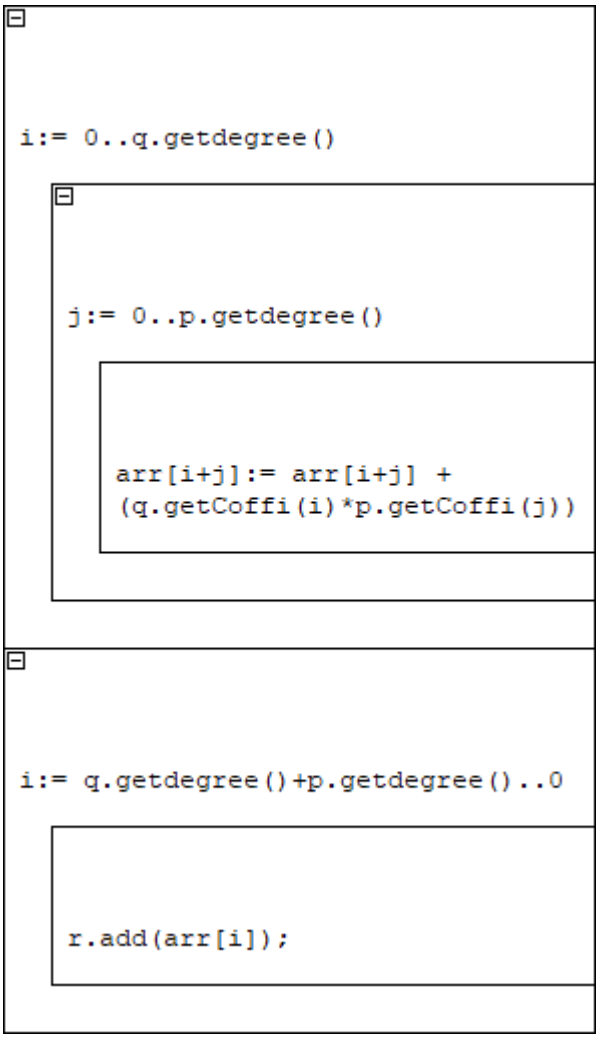
1. Adding Polynomials

The degrees of the two given polynomials are compared and the degree of the resultant polynomial is set equal to the higher degree between two. Then it adds the coefficients of the variables have same exponent/power.



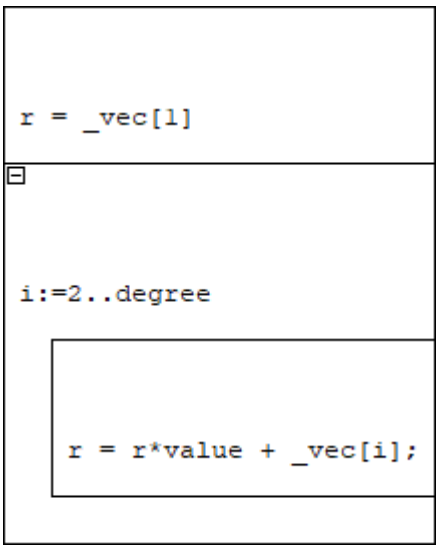
2. Multiplying Polynomials

An array of the size one more than sum of degrees of two given polynomials is initiated as 0. Then the function uses two nested loops to multiply each coefficient of the one polynomial with all the coefficients of the other polynomial. Then coefficients of variables with same exponents are added to their corresponding places in the array. Then another loop is used to copy all the elements of the array into the vector.



3. Evaluating a Polynomial

A number is initialized at the coefficient of the variable with highest exponent. Then loop is started and at every recursion, this number is multiplied by the given value and then the corresponding coefficient is added into it.



Class

Polynomial type is implemented as a class. The degree of the polynomial can be set by the degree variable.

Poly
-degree: int -_vec : vector<float>
+fill(int): void +getDegree(): int +setdegree(int): void +getCoffi(int): float +eval(float): float +addition(Poly,Poly): Poly +product(Poly,Poly): Poly +write(ostream,Poly): ostream

Exceptions
+NEGATIVE_DEGREE

Setter and getter functions are made to access and change degree as well as the coefficients. Coefficients are stored in a vector of floats vector<float>. The size of vector does not need to be set because vector adjusts its size whenever an element is pushed in or popped from it.

Write method is used as operator to write out a polynomial. The only exception is NEGATIVE_DEGREE which occurs when user enters a negative degree for a polynomial.

Testing

Following testing has been done:

1. Setting and getting degree and Coefficients of a Polynomial:

- Degree is set to two by setderee() method and coefficients for x^2, x and constant is given
- Using the getDegree() and getCoffi() functions, it is tested that values have been assigned correctly

2. Adding two Polynomials:

- Two polynomials are given and then added to give a third polynomial
- Then the third polynomial is tested

3. Multiplication of two Polynomials:

- Two polynomials are given and then multiplied to give a third polynomial
- Then the third polynomial is tested

4. Evaluation of a Polynomial by a given value:

- A polynomial is evaluated by giving a value to the eval function and then tested against the correct solution

5. Assignment Testing:

- Polynomial q is assigned the values of given Polynomial p and then tested

- Then Polynomial r is set equal to both p and q and tested against p
- Then p is made equal to p and compared

6. Copying a Polynomial:

- A given polynomial p is copied to a new polynomial q
- Then degrees and coefficients of p and q are compared

7. Exception Handling

- Polynomial p is assigned -2 degree
- The error is checked if it is Negative degree error or not