On the first two lines of ***pol.txt*** you will find 2 lists of **m/n** numbers representing the coefficients two an **m/n** degree polynomials. Reading those numbers from the file, construct the two polynomials from those coefficients.

On next **k** lines of ***pol.txt*** you will have ***k*** commands, each corresponding to a specific function which is to be applied on one or both polynomials. Read the lines and execute the commands sequentially.

Structure your code such that you have ***at least*** a *Polynomial* class, a *Functions* class and InputOutput class. All the results should be written in *output.txt*

Example of input:

pol.txt

```
5 -4 2 0 -2 3 0 3 -17
4 -2 0 1

ADD
SUBTRACT
MULTIPLY
MUL_SCAL 4
EVAL 5
EVAL 2
EVAL 9
```

The first polynomial is equivalent to: *5x^8 – 4x^7 + 2x^6 – 2x^4 + 2x^3 + 3x – 17*

The second polynomial is equivalent to: *4x^3 – 2x^2 + 1*

ADD – Add the two polynomials together and write the resulting polynomial
SUBTRACT – Subtract the two polynomials and write the resulting polynomial
MULTIPLY – Multiply the two polynomials and write the resulting polynomial
MUL_SCAL ‘n’ – Multiply the two polynomials with a scalar value and write the resulting polynomials
EVAL ‘n’ – Evaluate both polynomials on ‘n’ and write the results in the output file

*polynomials should be written in the *mathematical* form (e.g. *5x^8 – 4x^7…*)

Twist:
Implement division of polynomials.

Twist2:
Implement root approximation using one of the following techniques: *Bisection, Linear Interpolation, Newton’s method, Birge-Vietta* (or whichever method you find fit)

Twist3:
Read the two polynomials in their mathematical form. (hint: use stacks)