POLY – REPRESENTAÇÃO DE POLINÔMIOS
PROFESSOR: ADELARDO ADELINO DANTAS DE MEDEIROS

Escreva uma classe em C++, denominada `Poly`, capaz de representar polinômios *P(x)* de qualquer grau *n* (*n* ≥ *0*) com coeficientes reais a_i, do tipo:

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

Um polinômio de grau *n* será armazenado e representado internamente pela sua dimensão inteira *D (=n+1)* e pelos seus *D* coeficientes (ou seja, um array de *n+1* números reais). O coeficiente a_n nunca pode ser nulo, exceto para polinômios de grau 0, que correspondem a um número.

Desenvolva no mínimo as seguintes funcionalidades para a classe `Poly`:

- Crie os construtores (*default*, por cópia e específicos) e o destrutor apropriados:
  - O construtor *default* deve criar um polinômio inválido, com dimensão nula (*D=0*).
  - Um construtor específico deve criar um polinômio do grau passado como parâmetro (inteiro sem sinal):
    - Se o grau for nulo, deve criar um polinômio com dimensão *D=1* cujo valor do único coeficiente, a_0, será 0.0.
    - Se o grau for maior que zero, o polinômio deve ter todos os coeficientes nulos, com exceção do último, a_n, que terá valor 1.0.
    - Verifique se o construtor específico deve ou não poder ser utilizado como conversor de tipo (ou seja, se ele deve ou não ser `explicit`).

- Sobrecarregue o operador de atribuição (`operator=`). Lembre-se que o objeto que está sendo atribuído pode já conter informação anterior, sendo necessário prever a liberação de memória que eventualmente já tenha sido alocada previamente.

- Defina um método de consulta (função membro) `getGrau` para retornar o grau do polinômio (igual a *D-1*). O valor retornado é um número inteiro, que será negativo se o polinômio for inválido (ou seja, se *D=0*).

- Defina um método de consulta (função membro) `getCoef` para retornar o valor do *i*-ésimo coeficiente do polinômio. Por exemplo, se:
  $$P(x) = 3x^2 + 5x + 6$$
  então `P.getCoef(0)` deve retornar *6.0*. Lembre-se de checar a validade do índice. O valor retornado é um número em ponto flutuante.

- Sobrecarregue o `operator[]` utilizando o método `getCoef`, de tal forma que `P[0]` retorne o mesmo que `P.getCoef(0)`.

- Defina um método de consulta (função membro) `getValor` para retornar o valor do polinômio para um dado valor real de *x*, passado como parâmetro. Por exemplo, se:
  $$P(x) = 3x^2 + 5x + 6$$
  então `P.getValor(1.5)` deve retornar um número em ponto flutuante igual a:
  $$3(1.5)^2 + 5(1.5) + 6 = 20.25$$

- Sobrecarregue o `operator()` utilizando o método `getValor`, de tal forma que `P(1.5)` retorne o mesmo que `P.getValor(1.5)`.

- Crie um método `setGrau` que permita alterar o grau de um polinômio. Lembre-se que o objeto que está sendo alterado pode já conter informação anterior, sendo necessário prever a liberação de memória. Caso o grau seja alterado, o valor dos novos coeficientes deve ser o mesmo que no caso do construtor específico.

- Crie um método `setCoef` que permita alterar o valor de um coeficiente. Lembre-se de checar a validade do índice e que o último coeficiente a_n nunca pode ser nulo (exceto no caso do polinômio de grau 0).

- Sobrecarregue o operador << para escrever um polinômio na sua representação usual:
  - Iniciando com o coeficiente de maior grau.
  - Fazendo as adaptações necessárias para coeficientes negativos, unitários ou nulos.

  A tabela a seguir mostra alguns exemplos de valores de coeficientes e o resultado esperado da impressão.

Coeficientes: [a_0 a_1 … a_{n-1} a_n]	**Impressão**
[5.7 1.4 3.2 0.2]	`0.2*x^3+3.2*x^2+1.4*x+5.7`
[5.7 1.0 3.2 1.0]	`x^3+3.2*x^2+x+5.7`
[5.7 -1.4 3.2 -0.2]	`-0.2*x^3+3.2*x^2-1.4*x+5.7`
[5.7 -1.0 3.2 -1.0]	`-x^3+3.2*x^2-x+5.7`
[0.0 1.4 0.0 0.2]	`0.2*x^3+1.4*x`
[0.0 1.0]	`x`
[0.0]	0.0

- Sobrecarregue o operador >> de tal forma que permita que o usuário digite os coeficientes de um `Poly` qualquer (o grau ou dimensão deve ser estabelecido antes da entrada de dados). Lembre-se que o último coeficiente não pode ser nulo (exceto no polinômio de grau 0).

- Sobrecarregue os operadores + e - para fazer a soma e a subtração de dois polinômios, retornando o resultado. Lembre-se que as operações podem gerar polinômios de grau inferior ao esperado. Por exemplo, a soma de dois polinômios de segundo grau pode gerar um polinômio de primeiro grau, caso o último coeficiente do polinômio resultante seja nulo:
  $(-3x^2 + 5x + 6) + (3x^2 + x - 1) = 6x + 5$

- Sobrecarregue o operador - (unário) para retornar o negativo de um polinômio.

- Sobrecarregue o operador * para retornar o produto de dois polinômios.

OPCIONALMENTE (não faz parte da avaliação, fica apenas como estímulo ao aprendizado), desenvolva também as seguintes funcionalidades para a classe `Poly`:

- Sobrecarregue o operador / para retornar o resultado (o quociente) da divisão de dois polinômios.

- Sobrecarregue o operador % para retornar o resto da divisão de dois polinômios.

A classe `Poly` deve permitir compilar e executar o programa principal de uma minicalculadora de polinômios que está disponível no SIGAA e que oferece as seguintes opções a cada iteração (os itens em azul são opcionais):
Entrar um novo polinômio.
- Somar os polinômios.
- Subtrair os polinômios.
- Multiplicar os polinômios.
- Dividir os polinômios (retornar quociente)
- Dividir os polinômios (retornar resto)
- Calcular o último polinômio para um valor de x.
- Inverter o sinal do último polinômio.
- Trocar os polinômios.
- Terminar.