ALGORITMOS E ESTRUTURAS DE DADOS - IF969

2018.2

LISTA DE MONITORIA II

Q1. Utilizando vetores da biblioteca numpy, escreva uma classe Polinômio que contenha pelo menos os seguintes atributos:  
  
grau: o atributo grau deve guardar o valor do maior grau entre os monômios que formam o seu objeto Polinomio.  
Por exemplo: f = Polinomio(3, 2, 0) tem como monômio de maior grau 3x², logo o maior grau do monômio é 2.  
Note que em polinômios inicializados como por exemplo Polinomio(0, 0, 2, 0), o maior grau não é o do monômio formado pelo primeiro argumento (nesse caso seria 0\*x^4), mas sim o do primeiro argumento válido (2x), sendo este um polinômio de grau 1.  
  
E os seguintes métodos:  
  
*\_\_init\_\_*: o método construtor da classe deve receber como argumento os coeficientes do polinômio na ordem do monômio de maior grau até a constante.  
Por exemplo: f = Polinomio(3, -2, 1) seria equivalente ao polinômio F(x) = 3x²-2x+1;  
  
*derivar*: o método derivar deve retornar um novo objeto do tipo polinômio equivalente a derivada do objeto atual.  
Relembrando: a derivada de um monômio de coeficiente c e grau n é igual a um monômio de coeficiente c\*n e grau n-1; a derivada da soma de duas funções é a soma das derivadas dessas funções; a derivada de uma constante é 0.  
  
*\_\_call\_*\_: o método call é usado quando você chama a instância como uma função. Implemente esse método de modo a retornar o resultado do polinômio quando aplicado o argumento da função.  
Por exemplo: f = Polinomio (2, 0, 0); chamar f(2) resultaria num output de 8 (2\*2²);  
  
*\_\_getitem\_\_*: esse é o método responsável pela indexação de um objeto. Por exemplo, quando buscamos um objeto por índices em uma lista, é esse o método que controla essa ação. Você deve implementar o método \_\_getitem\_\_ de modo que a busca por índice retorne um objeto do tipo Polinômio contendo apenas o monômio de grau equivalente ao parâmetro índice.  
Por exemplo: f = Polinomio(4, 0, 3, 2); f[0] deve retornar a constante 2, equivalente ao objeto Polinomio(2), enquanto f[3] deve retornar o Polinomio (4, 0, 0, 0), equivalente ao monômio 4x^3.  
Caso o usuário tente acessar um grau inexistente no polinômio, sua classe deve levantar a exceção ArithmeticError (use a keyword raise).  
  
*\_\_iter\_\_*: o método iter é o que normalmente controla a forma como o for se comporta no seu objeto, tornando-o um objeto iterável. Esse método, normalmente, retorna um iterador do objeto self. A forma como ele "avança" no objeto é controlada pelo próximo método.  
  
*\_\_next\_\_*: o método next controla como o iterador aponta cada parte do seu objeto iterável. No nosso caso, queremos que, quando um for seja usado na classe Polinomio, o iterador caminhe do monômio de maior grau em direção ao monômio de menor grau. Para isso podemos usar a seguinte estratégia.  
Quando instanciar o método iter, crie um atributo privado que aponte o primeiro monômio (self.\_\_ponteiro, por exemplo). Ao chamar o método next, mova o valor de self.\_\_ponteiro para o próximo monômio e retorne o valor anteriormente guardado.  
Ao chegar ao fim da iteração, o método next deve levantar uma exceção StopIteration (use a keyword raise), ou o seu laço for nunca terá fim.  
  
*\_\_str\_\_*: o método de impressão deve retornar o polinômio em forma de função de grau n.  
Por exemplo: f = Polinomio (2, -2, 1); print(f) deve resultar em "f(x) = 2x^2-2x+1".  
Para o método \_\_str\_\_ não podem ser exibidos monômios de coeficiente 0.  
Por exmeplo: f = Polinomio(2, 0, 0); print(f) deve resultar em "f(x) = 2x^2".  
  
*\_\_repr\_\_*: o método de reprodução deve retornar uma forma válida de inicializar o objeto usando apenas o texto do output. Nesse caso, ele deve retornar uma string formatada com o nome da classe e, entre parêntesis, os argumentos necessários para inicializar um objeto de valor igual.  
Por exemplo: f = Polinomio(0, 0, 2); chamar f no console deve ter como output "Polinomio(2)".

[*newton\_raphson*](https://pt.wikipedia.org/wiki/Método_de_Newton–Raphson): esse método deve calcular o zero da função a partir de dois parâmetros: um valor aproximado para a raiz e um coeficiente de tolerância.

O algoritmo de newton-raphson é um modo iterativo de se estimar as raízes de um polinômio a partir do cálculo da reta tangente (derivada) deste.

Para a formulação desse método, assuma uma das três hipóteses para o coeficiente de tolerância:

1. |x-x0| < tol;
2. || < tol;
3. |f(x)| < tol

Escolha uma das condições acima como condição de parada do seu algoritmo newton-raphson.

Q2. Construa uma classe Matriz com as seguintes especificações:

Sua classe deve receber como parâmetros o número de linhas e colunas da matriz e inicializar a classe usando um vetor de apoio, no qual você deve dividir o número de linhas. Por exemplo, uma classe matriz com duas linhas e três colunas deve ser formada por um vetor de 6 elementos.

Você deve implementar o método \_\_getitem\_\_ de modo que o usuário possa facilmente acessar os valores de uma linha inteira (passando como parâmetro um índice inteiro de linha) ou um elemento específico (passando como parâmetro o índice exato, linha e coluna, do elemento).

Implemente também os métodos \_\_str\_\_ e \_\_repr\_\_ de acordo com instruções da questão anterior.

O que entregar: você deve entregar os códigos usados para responder ambas as questões, preferencialmente com cabeçalhos contendo as suas informações.

Quando entregar: a data limite para entrega é às 16h da quarta-feira dia 5 de Setembro.

*“Nós só podemos ver um pouco do futuro, mas o suficiente para perceber que há muito a fazer.”*

**— Alan Turing**