Departamento de Ciência da Computação - IME - USP

MAC2166 Introdução à Computação

Escola Politécnica - Primeiro Semestre de 2020 - Turmas Python

Segundo Exercício Programa   Entrega: até 16 de maio de 2020 às 23h55m

Polinômios íntegros e coerentes

"With great power comes
great difficulty in factorizing the polynomial."
- Autor desconhecido

" I had a polynomial once. My doctor removed it. "
- Michael Grant, escritor

Objetivos

Neste exercício-programa, você terá que desenvolver seis funções relacionadas com raízes de polinômios. O EP exercitará conceitos de funções e listas em Python.

Clique aqui para baixar o esqueleto do código do EP2.

Introdução

Seja um inteiro $n \geq 1$ e seja $p(x) = p_n x^n + \cdots + p_1 x + p_0$, com $p_n > 0$, um polinômio de grau n. Como se sabe, este polinômio admite uma *raiz* real b se $p(b) = p_n b^n + \cdots + p_1 b + p_0 = 0$. Em Python, podemos representar o polinômio $p(x)$ de grau n através de uma lista p com $n+1$ elementos, guardando em cada `p[i]` o respectivo coeficiente p_i, para $i = 0, 1, 2, \ldots, n$. Neste exercício programa, a correspondência entre lista e polinômio será feita sempre desta forma.

Tarefa 1

Escreva uma função **`polinomioComRaiz`** que recebe dois parâmetros **`p`** e **`b`** e devolve **`True`** caso o real **`b`** seja raiz do polinômio representado pela lista **`p`**, ou **`False`** caso contrário. Por exemplo, para o polinômio $p(x) = x^2 - 3.5x + 3$:

- `polinomioComRaiz([3, -3.5, 1], 1.5)` deve devolver `True`
- `polinomioComRaiz([3, -3.5, 1], -1.5)` deve devolver `False`

Atenção: Não modifique os nomes das funções obrigatórias do EP, nem seus parâmetros e tipos dos valores de retorno.

Observe que o polinômio $p(x)$ acima definido admite até n raízes reais. Ademais, quando o polinômio $p(x)$ admite uma raiz b, existe também um *polinômio quociente* $q(x)$ de grau $n-1$ tal que $p(x) = q(x)(x-b)$,

$$q(x) = \frac{p(x)}{(x-b)}.$$

Por exemplo:

- O quociente do polinômio $x^2 - 3.5x + 3$ por $x - 1.5$ é o polinômio $q(x) = x - 2$
- O quociente do polinômio $x^3 - x^2 + 2$ por $x + 1$ é $x^2 - 2x + 2$

$$\begin{array}{rrrr|l}
x^3 & -x^2 & +0x & +2 & x+1 \\
\cline{5-5}
-x^3 - x^2 & & & & x^2 \; -2x \; +2 \\
\cline{1-2}
 & -2x^2 & & & \\
 & +2x^2 & +2x & & \\
\cline{2-3}
 & & +2x & +2 & \\
 & & -2x & -2 & \\
\cline{3-4}
 & & & 0 &
\end{array}$$

Tarefa 2

Escreva uma função **`polinomioQuociente`** que recebe dois parâmetros **`p`** e **`b`** e devolve a lista que representa o polinômio quociente $q(x) = \frac{p(x)}{(x-b)}$, onde $b =$ **`b`** é uma raiz do polinômio $p(x)$ representado por **`p`**. Suponha que $p(x)$ tem grau ao menos 1 (não é necessário verificá-lo). Por exemplo:

- `polinomioQuociente([3, -3.5, 1], 1.5)` deve devolver a lista `[-2, 1]`
- `polinomioQuociente([2, 0, -1, 1], -1)` deve devolver a lista `[2, -2, 1]`

A partir de agora neste EP, considere que os coeficientes de $p(x)$ são todos inteiros. Ademais, suponha agora que $p_n = 1$.

No caso em que tenhamos exatamente n raízes reais $b_1, b_2, b_3, \ldots, b_n$, o polinômio $p(x)$ pode ser fatorado no seguinte produto de polinômios:

$$p(x) = p_0 + p_1 x + \cdots p_n x^n = (x - b_1)(x - b_2)(x - b_3) \cdots (x - b_n).$$

Nem sempre existe esta fatoração de $p(x)$ como produto de n polinômios de grau 1, e mesmo que exista a fatoração, é natural que nos indaguemos se suas raízes são todas inteiras ou não. Dizemos que o polinômio $p(x)$ é *íntegro* se TODAS as suas raízes $b_1, b_2, b_3, \ldots, b_n$ são inteiras, o que supõe a existência da fatoração acima. Alguns exemplos de polinômios íntegros incluem:

$$p(x) = x^2 - 5x + 6 = (x - 2)(x - 3),$$

$$p(x) = x^3 - 3x^2 + 3x - 1 = (x - 1)^3.$$

Já o polinômio $x^2 - 2x + 2 = (x - 1)^2 + 1$ é convexo (tem concavidade "para cima"), tem menor valor 1 para $x = 1$ e não possui nenhuma raiz real, não sendo portanto íntegro. Igualmente não é íntegro o polinômio $x^3 - x^2 + 2 = (x^2 - 2x + 2)(x + 1)$, embora admita uma raiz inteira -1. Já o polinômio $x^3 + 3x^2 + 3x + 1$ admite apenas a raiz inteira -1, mas é íntegro pois ela tem multiplicidade 3 já que o polinômio pode ser fatorado como $(x + 1)^3$.

Pelo teorema das raízes racionais, toda raiz inteira de $p(x)$ é um divisor do termo independente p_0.

Ainda que o polinômio $p(x)$ possa não ser íntegro e nem todas as suas n raízes serem inteiras, sempre é possível obter a *lista canônica das raízes inteiras* $b_1, b_2, b_3, \ldots, b_m$, de forma que existe um polinômio $q(x)$ de grau $n - m$ sem raízes inteiras tal que

$$p(x) = q(x)(x - b_1)(x - b_2)(x - b_3) \cdots (x - b_m).$$

O polinômio $p(x)$ é íntegro se e só se $m = n$ e $q(x) = 1$. Se por um lado nos interessa verificar se um dado polinômio $p(x)$ é íntegro, por outro lado interessa-nos também conhecer a lista de raízes inteiras $b_1, b_2, b_3, \ldots, b_n$ que o demonstre. Observe que qualquer uma das $n!$ permutações desta lista demonstra que $p(x)$ é íntegro. Assim, escolhemos definir como *lista canônica das raízes inteiras* a seguinte permutação:

> A lista $b_1, b_2, b_3, \ldots, b_n$ em que as raízes estejam em ordem crescente de valores absolutos, com raízes negativas precedendo raízes positivas de mesmo valor absoluto.

Exemplo:

- A lista canônica de raízes inteiras de $x^6 - 13x^4 + 36x^2$ é: $0, 0, -2, 2, -3, 3$
- A lista canônica de raízes inteiras de $x^5 - 11x^4 + 41x^3 - 61x^2 + 30x$ é: $0, 1, 2, 3, 5$

Tarefa 3

Escreva uma função **`listaCanonicaDeRaizes`** que recebe um parâmetro **`p`** com a lista que representa o polinômio $p(x)$ e devolve a lista canônica das raízes inteiras de $p(x)$. Por exemplo:

- `listaCanonicaDeRaizes([6, -5, 1])` deve devolver `[2, 3]`
- `listaCanonicaDeRaizes([2, -2, 1])` deve devolver `[]`
- `listaCanonicaDeRaizes([2, 0, -1, 1])` deve devolver `[-1]`
- `listaCanonicaDeRaizes([0, 0, 36, 0, -13, 0, 1])` deve devolver `[0, 0, -2, 2, -3, 3]`

A função `listaCanonicaDeRaizes` deve usar as anteriores quando necessário e devolve a lista vazia se não há raízes inteiras.

Num caso mais geral, suponha agora que $p_n > 0$ seja um natural qualquer, mantendo a restrição de que os coeficientes de $p(x)$ são inteiros.

Seja um racional b/a, com a e b primos entre si, e $a > 0$. Conforme a demonstração do **teorema das raízes racionais pelo lema de Gauss**, $p(x)$ possui raiz b/a se e só se $(ax - b)$ divide $p(x)$, que por sua vez implica que b é um divisor de p_0 e a é um divisor de p_n. Com isso, podemos criar uma versão mais geral da função `polinomioQuociente`, que permita dividir por $(ax - b)$.

Como as alterações necessárias para permitir essa divisão são poucas, aproveitaremos para dar uma nova funcionalidade a esta nova versão. Considere o polinômio $p(x) = 15x^2 - x - 2$, com raízes $2/5$ e $-1/3$. Talvez você consiga verificar a primeira raiz com a função `polinomioComRaiz` que implementou, mas não consiga fazer o mesmo para a segunda. Mesmo que o algoritmo pareça correto, pode acontecer que ele não funcione adequadamente devido a um problema de instabilidade numérica gerada pelos arredondamentos efetuados nos sucessivos cálculos com números reais. Faça um teste e verifique se sua função dá a resposta correta nesse exemplo.

De forma a prover um método alternativo para reduzir este problema de instabilidade numérica, pediremos mais uma alteração na nova computação do polinômio quociente.

Tarefa 4

Escreva uma função **`polinomioQuocienteRacional`** que recebe três parâmetros **`p`**, **`b`** e **`a`** e devolve a lista **`q`** que representa o polinômio quociente $q(x)$ e o resto da divisão **`r`** $= r$ tais que $p(x) = q(x)(ax - b) + r$. Isto naturalmente pressupõe que $p(x)$ tenha grau ao menos 1. Caso contrário, a função deve devolver **`None`** e **`-1`** como valores de **`q`** e **`r`**. Por exemplo:

- `polinomioQuocienteRacional([6, -7, 2], 3, 2)` devolve `[-2.0, 1.0]` e `0.0`,
  já que $p(x) = 2x^2 - 7x + 6 = q(x)(ax - b) + r = (x - 2)(2x - 3) + 0$
- `polinomioQuocienteRacional([6, -7, 2], -3, 2)` devolve `[-5.0, 1.0]` e `21.0`,
  já que $p(x) = 2x^2 - 7x + 6 = q(x)(ax - b) + r = (x - 5)(2x + 3) + 21$
- `polinomioQuocienteRacional([4], 2, 1)` devolve `None` e `-1`,
  já que $p(x) = 4$ tem grau 0

Atenção: Você não precisa corrigir o problema de instabilidade numérica na sua implementação da função **`polinomioQuociente`** (caso ele exista).

Observe que $p(b/a) = r$, de forma que b/a é uma raiz do polinômio $p(x)$ representado por `p` se e só se $r = 0$. Com este método verifica-se que $15x^2 - x - 2$ tem raiz $-1/3$.

Assim sendo, de forma semelhante ao caso já visto em que $p_n = 1$, definimos uma *lista canônica das raízes racionais* de $p(x)$, associando uma fatoração

$$p(x) = p_0 + p_1x + \cdots p_nx^n = q(x)(a_1x - b_1)(a_2x - b_2)(a_3x - b_3)\cdots(a_mx - b_m),$$

tal que: cada b_i/a_i é uma raiz racional de $p(x)$, para $i = 1, \ldots, m$; e $q(x)$ não admite nenhuma raiz racional.

O polinômio $p(x)$ é dito *coerente* se suas n raízes são todas racionais, o que equivale a dizer que $m = n$ e $q(x)$ só possui o termo independente. Definimos a *lista canônica das raízes racionais* de $p(x)$ como a lista das raízes b_1/a_1, b_2/a_2, b_3/a_3, $\ldots$, b_m/a_m na seguinte ordem: primeiramente as raízes nulas; em seguida, aquelas com denominador 1; depois as com denominador 2, e assim por diante. Como cada raiz racional b_i/a_i deve ser tal que b_i seja um divisor de p_0 e cada a_i seja um divisor positivo de p_n, é natural que só se considerem como possíveis denominadores aqueles que dividem p_n segundo a ordem de valores absolutos crescentes. Ademais, para o mesmo denominador a, testamos candidatos a raízes $-b/a$ e b/a, nesta ordem, listando-os segundo valores crescentes de b dentre os possíveis divisores de p_0. Cada raiz b_i/a_i assim listada é naturalmente obtida com b por a primos entre si, pois raízes com quociente menor que a já terão sido extraídas antes.

Tarefa 5

Escreva uma função **`listaRaizesRacionais`** que recebe um parâmetro **`p`** com a lista que representa o polinômio $p(x)$ e devolve a lista canônica das raízes racionais de $p(x)$. Por exemplo:

- `listaRaizesRacionais([0, 20, 12, -113, -113, 12, 20])` deve devolver `[0, -1.0, -2.0, -0.5, 2.5, 0.4]`
- `listaRaizesRacionais([0, -6, 5, 73, -40, -296, 80, 400])` deve devolver `[0, -0.5, -0.5, 0.5, 0.5, 0.4, -0.6]`
- `listaRaizesRacionais([-2, -1, 15])` deve devolver `[-0.3333333333333333, 0.4]`
- `listaRaizesRacionais([1, -34, 404, -1934, 3003])` deve devolver `[0.3333333333333333, 0.14285714285714285, 0.09090909090909091, 0.07692307692307693]`

A função `listaRaizesRacionais` deve usar `polinomioQuocienteRacional` quando necessário e devolve a lista vazia se não há raízes racionais.

O último exemplo de uma lista canônica das raízes racionais bem revela o quanto a impressão de raízes racionais como se fossem números reais quaisquer pode ser desagradável e pouco informativa, sem evidenciar qual é a fração correspondente.

Tarefa 6

Escreva uma função **`racionalToString`** que recebe como parâmetros o coeficiente **`pn`** do termo de maior grau de um polinômio de coeficientes inteiros e uma raiz racional **`r`** deste polinômio. A função devolve a *string* que apresente a raiz **`r`** como: um inteiro, caso **`r`** seja inteiro; na forma **`b/a`**, com **`b`** e **`a`** primos entre si e **`a`** > 0, caso contrário. Observe que basta simplificar o quociente **`round(r*pn) / pn`** dividindo numerador e denominador pelo seu MDC (Máximo Divisor Comum). Por exemplo:

- `racionalToString(20, 0)` deve devolver a *string* `'0'`
- `racionalToString(20, -2.0)` deve devolver a *string* `'-2'`

- `racionalToString(400, -0.6)` deve devolver a *string* `'-3/5'`
- `racionalToString(3003, 0.07692307692307693)` deve devolver a *string* `'1/13'`

O programa principal

No **esqueleto de código que vocês devem usar no EP2**, a função `main` já está pronta. Ela usa as funções que vocês devem implementar no exercício.

A `main` lê o grau n do polinômio $p(x)$ bem como seus coeficientes, todos inteiros e com MDC unitários. Caso $p_n = 1$, ela usa a função `listaCanonicaDeRaizes` e imprime a lista canônica de raízes inteiras de $p(x)$; caso contrário, ela usa a função `listaRaizesRacionais` e imprime a lista canônica de raízes racionais de $p(x)$.

Antes da lista propriamente dita, a `main` usa a função `polinomioToString` (também fornecida no esqueleto) para imprimir o polinômio $p(x)$. Essa função usa a *string* devolvida pela função `racionalToString` para a impressão de cada raiz racional.

Veja abaixo código da função `polinomioToString` usada na impressão do polinômio no programa principal. Seu uso do operador % serve de exemplo na formatação da *string* de `racionalToString`:

```
def polinomioToString(p):
    n = len(p)-1
    s = ''
    for m in range(n-1):
        if p[n-m] != 0:
            s = '%s%s%dx^%d ' % (s, sig(m, p[n-m]), p[n-m], n-m)
    if p[1] != 0:
        s = '%s%s%dx ' % (s, sig(n-1, p[1]), p[1])
    if p[0] != 0:
        s = '%s%s%d' % (s, sig(n, p[0]), p[0])
    return s

def sig(nTermAnte, coef):
    if nTermAnte > 0 and coef >= 0:
        return '+'
    else:
        return ''
```

Exemplos de execução do programa

Nos exemplos, tudo que aparece em **vermelho** foi digitado pelo usuário.

Exemplo 1

```
Digite o grau: 6
Digite p[0]: 0
Digite p[1]: 0
Digite p[2]: 36
Digite p[3]: 0
Digite p[4]: -13
Digite p[5]: 0
Digite p[6]: 1
A lista canonica das raizes inteiras de p(x) = 1x^6 -13x^4 +36x^2  eh:
0 0 -2 2 -3 3
```

Exemplo 2

```
Digite o grau: 5
Digite p[0]: 20
Digite p[1]: 12
Digite p[2]: -113
Digite p[3]: -113
Digite p[4]: 12
Digite p[5]: 20
A lista canonica das raizes racionais de p(x) = 20x^5 +12x^4 -113x^3 -113x^2 +12x +20 eh:
-1 -2 -1/2 5/2 2/5
```

Exemplo 3

```
Digite o grau: 4
Digite p[0]: 1
Digite p[1]: -34
Digite p[2]: 404
Digite p[3]: -1934
Digite p[4]: 3003
A lista canonica das raizes racionais de p(x) = 3003x^4 -1934x^3 +404x^2 -34x +1 eh:
1/3 1/7 1/11 1/13
```

Importante

- No EP2, você só pode usar as funções `print` e `input` dentro da função `main`.
- Você pode modificar a função `main` da forma que quiser para testar suas demais funções.
- O código e as saídas impressas da função `main` não serão avaliados na correção do EP.
- Você não precisa se preocupar com situações em que o usuário não digite os números como esperado. A correção do EP não levará em conta essas situações.
- As únicas construções da linguagem Python que você poderá usar em seu programa são as constantes deste enunciado e as dadas em aula.

Entrega do EP

Leia as **INFORMAÇÕES SOBRE ENTREGA DE EPs** antes de entregar o seu EP.

Certifique-se de que o seu programa foi realmente depositado no site.