Lista de Computação I (Python)

DCC / UFRJ

Pedro Maciel Xavier (monitor)
pedromxavier@poli.ufrj.br

28 de dezembro de 2019

Introdução

Essa lista de exercícios ainda se encontra em desenvolvimento. A intenção é que ela tenha um gabarito bem aberto, deixando muito das respostas para a criatividade do aluno. As questões são, em geral, grandes para se resolver e podem necessitar de alguma pesquisa adicional. Elas tem estrelinhas $\star$ indicando a dificuldade estimada. Alguns exercícios foram inspirados em outros propostos em materiais cujas fontes estão devidamente referenciadas no final. É importante você tire suas dúvidas e dê um retorno do que achou dos exercícios através do `e-mail` no cabeçalho.

Boa diversão!

Sumário

1 Condições (`if`, `elif`, `else`)

2 Números (`int`, `float`, `complex`)

2.1 Números Primos ⋆ ⋆

Desafio: Faça uma função que verifique se um número é primo ou não.

2.2 Gerando números "aleatórios" ⋆

Gerar um numero aleatório é um pouco complicado em geral. Muitas formas de se fazer isso hoje em dia se baseiam em processos da natureza como fluidos em regime turbulento ou fenômenos quânticos. No entanto, é possível chegar perto disso com muito menos. Para começar, escolhemos uma potência de 2, como $\mathbf{N} = 2^{16} = 65536$. Depois disso, escolhemos um número primo no meio do caminho, digamos, $\mathbf{p} = 25773$. Por fim, um númeror ímpar menor que o número primo $\mathbf{p}$, por exemplo, $\mathbf{b} = 13849$.

Partimos de um número $\mathbf{n}_0$ qualquer no intervalo $[0, 25536)$ e calculamos o sucessor da seguinte forma:

$$\mathbf{n}_j = \mathbf{p} \times \mathbf{n}_{j-1} + \mathbf{b} \mod \mathbf{N}$$

Para obter um número "aleatório"distribuído de maneira uniforme no intervalo $[0, 1)$, basta calcular

$$\mathbf{x}_j = \frac{\mathbf{n}_j}{\mathbf{N}}$$

Desafio: Faça o seu próprio gerador de números aleatórios!

2.3 Música I ⋆

A	A#	B	C	C#	D	D#	E	F	F#	G	G#
0	1	2	3	4	5	6	7	8	9	10	11

Desafio: Faça uma função que calcule a frequência $\mathbf{f}$ de cada nota, a partir do seu número, supondo que $\mathbf{f}(0) = 440$Hz.

2.4 Diferencial I ⋆

Aprendemos em Cálculo I que a derivada de uma função $f(x)$ pode ser calculada como:

$$f'(x) = \lim_{h \to 0}$$

Se escolhemos um h pequeno, algo como 0.001, podemos deixar a ideia de limite de lado e partir para uma aproximação

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \text{ para } h \ll 1$$

Outras aproximações razoáveis são

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

$$f'(x) \approx \frac{\mathrm{Im}\{f(x + \mathbf{i}h)\}}{h}$$

sendo essa última um pouco mais holística.

Desafio: Faça uma função `d(f, h)` que receba uma função `f` e um número bem pequeno `h` e retorne uma outra função, que aproxime a derivada de `f`.

3 *Strings* e Texto (`str`)

3.1 DNA ⋆

Uma sequência de DNA (*ácido desoxirribonucleico*) é composta por 4 nucleotídeos: **A**denina, **C**itosina, **G**uanina **T**imina. Já no RNA mensageiro (mRNA), a **T**imina está ausente, e dá lugar para a **U**racila. Quando um ribossomo realiza a transcrição de DNA em mRNA ele segue uma regra muito simples:

$$\begin{array}{ccc} A & \rightarrow & U \\ T & \rightarrow & A \\ C & \rightarrow & G \\ G & \rightarrow & C \end{array}$$

Podemos representar uma fita de DNA usando uma *string* (`str`), como por exemplo:

```
"ATCGTACGACTATACTACGTAGCTAAACGCATACGATCGACAAAGC"
```

Desafio: Faça uma função que traduza uma fita de DNA como faria um ribossomo.

3.2 Separando sílabas ⋆ ⋆ ⋆ ⋆

Separa sílabas é uma tarefa interessante. O Português é uma das poucas línguas que possui um algoritmo que permite saber como separar as sílabas de uma palavra qualquer, isto é, se uma palavra nova for criada hoje em português, já é possível dividí-la mesmo sem conhecer a sua origem.

Flamengo → Fla-men-go

Um algoritmo de separação silábica, para uma língua qualquer, normalmente conta com duas coisas: Um conjunto de regras e um conjunto de exceções.

As regras nada mais são do que padrões que, uma vez observados, implicam em uma cisão na palavra. Por exemplo, consoantes dobradas como **rr** e **ss** sempre vão indicar uma separação:

Pássaro → Pás-sa-ro
Carrossel → Car-ros-sel

As exceções nada mais são do que um conjunto de palavras que já sabemos como separar, armazenadas em pares da forma (palavra, separação).

Desafio: Crie um conjunto de regras para que, dada uma palavra (`str`), a função retorne uma lista com a respectiva separação silábica. Exemplo:

```
>>> f('pássaro')
['pás', 'sa', 'ro']
>>> f('carrossel')
['car', 'ros', 'sel']
```

Listing 1: "Separando sílabas"

4 Listas e Tuplas (`list`, `tuple`)

4.1 Crivo ⋆ ⋆

O Eratóstenes foi um cara legal que viveu em 200 a.C., bolou um crivo e calculou a curvatura da terra.

Figura 1: Eratóstenes de Cirene

Um **crivo** é uma forma de saber quem são os números primos até um determinado limite **N**. O Crivo de Eratóstenes funciona de forma bem simples:

1. Escrevemos em uma tabela todos os números de 0 até **N**.
2. Riscamos o 0 e o 1.
3. Se um número não estiver riscado, riscamos todos os múltiplos deste, menos o próprio.
4. Andamos para o próximo número e repetimos a etapa anterior.

Uma maneira interessante de fazer isso é criando uma lista de tamanho **N**+1, ou seja, cujos índices vão de 0 até **N**.

5 Funções e Recursividade (`def`, `return`)

Interlúdio: Decoradores

Decoradores ...

```
1 def dec(f):
2     def g(x):
3         return -f(x)
4     return g
5
6 @dec
7 def f(x):
8     return x
```

---

6 Laços (`for`, `while`)

6.1 Sequência de Collatz ⋆ ⋆

A sequência de Collatz é descrita recursivamente por:

$$f(n) \triangleq \begin{cases} 3n+1, & \text{se } n \text{ for ímpar} \\ n \div 2, & \text{se } n \text{ for par} \end{cases}$$

Por exemplo, começamos com $n = 26$. Após sucessivas aplicações temos:

$$26 \to 13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$$

Isso nos dá uma sequência com 11 números. 40 é maior que 26, mas sua sequência só teria 9 números.

Ainda não se sabe se todos os números induzem uma sequência que termina em 1. No entanto, até agora não foi encontrado um número sequer em que isso não tenha acontecido!

Deasfio: Faça uma função que calcule o comprimento da sequência gerada a partir de um número natural n qualquer.

7 Conjuntos e Dicionários (`set`, `dict`)

7.1 Música II ⋆ ⋆

Usando a função do exercício **Música I**[2], podemos usar um dicionário para obter a frequência a partir do símbolo que representa a nota.

8 Arquivos (`with`, `open`)

Interlúdio: `csv`

Arquivos no formato `.csv` (*comma separated values*) são muito utilizados para representar tabelas em um arquivo de texto. São facilmente lidos por programas como o Excel™.

Por exemplo, o arquivo

```
1 nome,CPF,idade
2 João,82252631704,11
3 Pedro,12432236702,21
4 Anna,13241232392,15
```

é interpretado como

nome	CPF	idade
João	82252631704	11
Pedro	12432236702	21
Anna	13241232392	15
