Python para os fortes: temas avançados em computação científica

Prof. Dr. Luiz T. F. Eleno (EEL/USP)
Minicurso durante a SEMEF/SBEF 2022

Resumo

Iterators, *generators*, *decorators*, *classes*, *constructors*, etc.: palavras-chaves na biblioteca padrão python que a maioria dos que o usam não conhece! A maior parte dos usuários de python utiliza menos de 10% das capacidades da linguagem. Com isso, frequentemente encontramos códigos com dezenas de linhas que poderiam ser compactados para menos de cinco... o que leva a códigos com tempo de execução de minutos que poderiam demorar apenas frações de segundo. Além desse problema, boa parte dos usuários não estrutura seus códigos, copiando e colando fragmentos muitas vezes obsoletos da internet, sem saber o que realmente fazem. Os códigos acabam então se tornando dinossauros-frankensteins digitais: lentos, ineficientes, pesados e fadados à extinção. Mas a boa notícia é que estruturar códigos é mais fácil do que muitos pensam. Basta um pouco de dedicação para se conseguir uma visão estruturada e criar códigos limpos e racionais, extremamente eficientes e claros. É o que pretendo demonstrar neste curso.

Advertência

Apesar da pretensão que transpira do resumo acima, é preciso dizer que todos estamos sempre aprendendo. Examino meus códigos antigos e vejo que poderia ter implementado coisas de modo diferente, usando recursos que conheci apenas depois, aproveitando sacadas e macetes que fui descobrindo e inventando ao longo do tempo. A caminhada nunca termina. Portanto, se seu código parece longo e ineficiente, é porque você ainda não conhece python o suficiente. E, mesmo que conheça, e ache que não dá pra fazer melhor — você está quase certamente enganado! É provável que exista algum recurso do python padrão, ou em algum módulo adicional, que tornará seu código ainda mais eficiente. A caminhada nunca termina. Existem aqueles que se dão por satisfeitos quando o código funciona, ainda que não seja muito eficiente. Mas existem outros que buscam sempre, assintoticamente, a perfeição inatingível. Para estes, e não aqueles, a caminhada nunca termina. Mas chegarão mais longe.

Contatos

Prof. Dr. Luiz T. F. Eleno (EEL/USP)

luizeleno@usp.br

https://www.demar.eel.usp.br/docentes/luiz-tadeu-fernandes-eleno.html

Tópicos do curso

1. Generalidades: variáveis de ambiente e atributos
2. Objetos iteráveis
   c. abrangência (*comprehension*) e iteráveis
   d. `iter()` e `next()`
   e. a declaração `yield`
   f. percorrendo iteráveis simultaneamente – `zip()`
3. Funções
   a. argumentos posicionais e nomeados; obrigatórios e opcionais
   b. anotações de função
   c. funções anônimas (com o construtor `lambda`)
   d. variáveis locais, não-locais, livres e globais
   e. *closure* e funções de funções
   f. módulo `functools`: `partial` e `@cache`
   g. `decorators`
      i. alguns decoradores padrão: `classmethod` e `staticmethod`
4. Classes e objetos
   a. `constructors`
   b. sobrecarga de operadores
   c. herança

Referências

1. Mark Lutz, *Python Pocket Reference*, 5ed.
2. Luciano Ramalho, *Python fluente*.
3. https://realpython.com/python-kwargs-and-args
4. https://realpython.com/python-zip-function
5. https://docs.python.org/3/library/functools.html
6. https://realpython.com/python-class-constructor
7. https://realpython.com/primer-on-python-decorators
8. https://realpython.com/inheritance-composition-python