Computação I - Python
Aula 9 - Laços aninhados e matrizes

Introdução aos laços aninhados

Apresentado por: Anamaria Martins Moreira

Produção DCC-UFRJ

Metodologia de referência https://doi.org/10.5753/wei.2016.9683

Lidando com listas ou tuplas de elementos indexáveis

- Elementos de listas e tuplas podem ser de qualquer tipo de dados. P. ex., listas de strings ou listas heterogêneas, onde alguns dos seus elementos são indexáveis.
- Exemplo frequente: a função str.split, que gera uma lista de strings.
- Uma vez gerada essa lista, podemos precisar que cada caracter de cada uma das strings seja testado para a realização de alguma ação.
- Para conseguir realizar este tipo de construção algorítmica, usamos laços aninhados:
  - um laço (chamado de laço externo) para percorrer todas as strings da lista e
  - para cada uma destas strings, um outro laço (laço interno) para percorrer seus caracteres.

Lidando com listas ou tuplas de elementos indexáveis

- Esta não é a única situação onde usamos laços aninhados, mas é muito comum.
- O aninhamento pode ter quantos níveis forem necessários, e não apenas 2.
- Quando queremos controlar melhor a complexidade de uma solução com vários níveis de aninhamento podemos fazer uso de funções.

Exemplo: contando ocorrências

Defina uma função que recebe uma frase e uma letra e devolve uma lista contendo o número de ocorrências da letra em cada uma das palavras da frase.

```
def conta_ocorrencias_v1(frase, letra):
    ''' retorna uma lista contendo a quantidade de vezes em que a letra
    aparece em cada palavra da frase.
    str, str --> list'''
    palavras = str.split(frase)
    ocorrencias = []
    for palavra in palavras:
        list.append(ocorrencias, str.count(palavra, letra))
    return ocorrencias
```

```
>>> conta_ocorrencias_v1(' By using Python Tutor ', 't')
[0, 0, 1, 1]
>>>
```

Exemplo: contando ocorrências

Agora com uma pequena alteração para ficar mais fácil de entender o que esta sendo feito em list.append:

```
def conta_ocorrencias_v2(frase, letra):
    ''' retorna uma lista contendo a quantidade de vezes em que a letra
    aparece em cada palavra da frase.
    str, str --> list'''
    palavras = str.split(frase)
    ocorrencias = []
    for palavra in palavras:
        qtd = str.count(palavra, letra)
        list.append(ocorrencias, qtd)
    return ocorrencias
```

E cadê os laços aninhados? Só tem um laço aí...

- Na realidade, o que faz a função str.count?
- Ela percorre a string posição a posição, verificando se cada um dos caracteres é aquele que procuramos.
- Esse é um dos casos onde quebramos a complexidade da solução usando uma função (str.count) para fazer parte da computação necessária.
- Como seria então o código sem esta facilidade?

Exemplo: contando ocorrências

```
def conta_ocorrencias_v3(frase, letra):
    ''' retorna uma lista contendo a quantidade de vezes em que a letra
    aparece em cada palavra da frase.
    str, str --> list'''
    palavras = str.split(frase)
    ocorrencias = []
    for palavra in palavras:
        qtd = 0
        for c in palavra:
            if c == letra:
                qtd = qtd + 1
        list.append(ocorrencias, qtd)
    return ocorrencias
```

Vamos fazer um teste de mesa desta função no Python Tutor?