Uso de dicionário em Python

Neste documento, vamos discutir brevemente o que é **dicionário** na linguagem Python. No material de aula do Teams temos mais programas de exemplo que exploram o conceito.

Relembrando...

Lembre-se que em Python nós temos alguns tipos de dados chamados **sequenciais**, pois podem armazenar vários dados. São alguns deles:

- String (`str`) (sequência de caracteres)
- Lista (`list`)
- Tupla (`tuple`)
- Dicionário (`dict`)

Na lista e na tupla, o conteúdo é armazenado de forma sequencial e acessado de acordo com **índices**, começando do zero. Assim, se criamos uma lista como:

```
produtos = ['banana', 'uva', 'alface', 'iogurte']
```

O conteúdo `'banana'` está no índice 0 (zero). Os demais elementos estão, em ordem, nos índices 1, 2 e 3.

Se quisermos acessar esses dados, fazemos isso através desses índices.

```
print(produtos[3])      # imprime a palavra "iogurte"
produto[1] = 'kiwi'     # troca uva por kiwi na lista
```

E o dicionário?

O dicionário, que vamos tratar aqui, também é um tipo sequencial. No entanto, seus dados não estão associados a índices, e sim a **chaves**.

As chaves no dicionário podem ser strings, inteiros, e muitas outras coisas.

(Tecnicamente, o Python permite como chaves do dicionário quaisquer tipos **imutáveis**, tais como `int`, `float`, `str`, `bool`, `tuple`. Na prática, quase sempre usamos string e inteiros, mas os outros tipos também têm aplicações possíveis.)

Por exemplo:

```
pessoa = {
    'nome': 'Alessandra',
    'sobrenome': 'Nunes',
    'idade': 39,
```

```
    'CPF': '123456789-0'
}
```

Se quisermos acessar cada um desses campos, usamos a mesma sintaxe da lista, mas entre `[]` colocamos a chave em vez de um índice numérico:

```
print(pessoa['CPF'])    # imprime o CPF
pessoa['idade'] += 1    # aumenta a idade em uma unidade
print(f'Nome completo: {pessoa['nome']} {pessoa['sobrenome']}')
```

Adicionando novos dados no dicionário

Para adicionar um novo dado a um dicionário já existente, é mais simples do que na lista. Basta escolhermos uma nova chave e fazer a atribuição "como se a chave já existisse". Ela será criada.

```
pessoa['estado'] = 'MG' # adiciona o dado 'MG' no novo campo 'estado'
```

Com isso, temos duas formas principais de criar um dicionário!

Forma 1: adicionar em um único comando todos os campos

Podemos fazer como no exemplo inicial: criar o dicionário já com os campos desejados. A sintaxe é:

```
nome_do_dicionário = {
    chave_1 : dado_1,
    chave_2 : dado_2,
    # ... (e assim por diante)
    chave_N : dado_N
}
```

Isto é, a cada linha escrevemos a chave, depois dois pontos `:`, e finalmente o dado associado. Além disso, a cada linha adicionamos uma vírgula (`,`) exceto no caso do último dado. Recapitulando o exemplo mais concreto mostrado anteriormente, fica assim:

```
pessoa = {
    'nome': 'Alessandra',
    'sobrenome': 'Nunes',
    'idade': 39,
    'CPF': '123456789-0'
}
```

Forma 2: criar o dicionário vazio e adicionar os campos depois

Como vimos, é fácil criar um novo dado dentro do dicionário. Isso nos permite criar o dicionário da seguinte forma: começamos com o dicionário vazio e depois adicionamos os dados desejados, um por um.

```
pessoa = {} # começa como dicionário vazio
pessoa['nome'] = 'Alessandra'
pessoa['sobrenome'] = 'Lima'
pessoa['idade'] = 39
pessoa['CPF'] = '123456789-0'
```