FHO | FUNDAÇÃO HERMÍNIO OMETTO

Programação de Computadores III

Dicionário

Bibliografia

•DOWNEY A. Pense em Python. São Paulo: Novatec, 2016.

MENEZES, N. N. C. Introdução à Programação com Python. São Paulo: Novatec, 2014.

WAZLAWICK, R. S. Introdução a Algoritmos e Programação com Python. 1. ed. Elsevier, 2017, 232p.

•Sedgewick, Robert, Kevin Wayne, and Robert Dondero. Introduction to programming in Python: An interdisciplinary approach. Addison-Wesley Professional, 2015

•MARTELLI A., ASCHER D. Python Cookbook. O'Reilly, 2002, 575 pages.

•ASCHER D., LUTZ M. Aprendendo Python, Editora Bookman, 2ª edição, 2007, 566 páginas.

•FORBELLONE, A. L. V.; EBERSPÄCHER, H. F.. Lógica de programação : a construção de algoritmos e estruturas de dados. 3.ed. São Paulo:Prentice Hall, 2005. xii, 218p.

•BORGES, L. E. Python para Desenvolvedores. São Paulo: Novatec, 2014.

Variáveis Compostas - Dicionário

- Dicionários podem ser pensados como variáveis que armazenam diversos valores de diversos tipos de dados referenciados por **índices que podem ser personalizados.**
-

Variáveis Compostas - Dicionário

- Um Dicionário em Python constitui uma coleção de dados de diferentes tipos sob o mesmo nome. É possível inicializar uma variável como um dicionário vazio bastante atribuir a ele o seguinte comando.

Python

```
dados ={ }
dados = dict( )
```

Importante:

- Para criar um dicionário ja com dados inseridos basta declarar ele:

```
'''

AMBIENTE DE PROGRAMAÇÃO

'''

dados = {'nome':'Pedro','idade':25}
print(dados)
```

  - dentro das {} o primeiro campo antes do : é o índice e depois do : é o valor desse índice.

- Resultado apresentado.

```
{'nome': 'Pedro', 'idade': 25}
```

-

Importante:

- O comando append não funciona com dicionários.
- quando queremos adicionar um conteúdo em dicionário utilizamos dessa forma:

```
10 dados ['sexo']='M'
```

- Deixando nosso dicionário assim:

```
{'nome': 'Pedro', 'idade': 25, 'sexo': 'M'}
```

Comandos Importantes

- **Comando values:**
- retorna os valores que estão armazenando no dicionário.
    - dados.values()

- **Comando keys:**
- retorna as chaves (índices) que foram determinados no dicionário.
    - dados.keys()

- **Comando Items:**
- retorna os valores que estão armazenados no dicionário e os índices que forem determinados (values + keys).
    - dados.items()

- é interessante utilizar o comando items vinculado a uma estrutura de repelão For (comando se assemelha ao comando enumerate em lista).

Biblioteca Time e operator

- Podemos utilizar algumas bibliotecas extra para brincar e deixar nossos programas mais intuitivos para o usuário, uma dela é a time.

```
15 import time
16
17 time.sleep(1)
```

- Com essa função, o sistema aguarda 1 segundo antes de executar a nova linha de comando.
- Também podemos utilizar a biblioteca operator.

```
16 import operator
17
18 operator.itemgetter(1)
```

- Com essa função, o sistema irá ordenar os elementos da lista partindo da comparação com o valor selecionado
-

Exercício 01

- Crie uma agenda de telefone, receba do usuários nomes e telefones enquanto ele desejar. Exiba ao final a agenda completa.

Exercício 01

```
'''

AMBIENTE DE PROGRAMAÇÃO

'''
agenda = []
dados = {}
resp = 'SIM'
while resp == 'SIM':
    dados['nome'] = input('Nome:')
    dados['telefone'] = input('Telefone : ')
    agenda.append(dados.copy())
    resp = input('Deseja cadastrar mais números na agenda? - [SIM/NÃO]')
    resp = resp.upper()
print(agenda)
```

```
Nome:Thiago Milani

Telefone : 3456-6575

Deseja cadastrar mais números na agenda? - [SIM/NÃO]sim

Nome:Ana Luza

Telefone : 4564-3324

Deseja cadastrar mais números na agenda? - [SIM/NÃO]não
[{'nome': 'Thiago Milani', 'telefone': '3456-6575'}, {'nome': 'Ana Luza',
'telefone': '4564-3324'}]
```