

PROGRAMAÇÃO FULL STACK

Módulo 2 – Python

Aula 02 – Dicionários e Sets

Conteúdo da Lista: dicionários <class 'dict'>, métodos manipuladores de dicionários, sets <class 'set'> e métodos manipuladores de sets

Exercício 01

Crie um arquivo Python **exercicio01.py**. Após, faça a declaração de um dicionário vazio "pessoas".

Exercício 02

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio02.py**. Após, adicione uma entrada no dicionário "pessoas" com a chave "nome" e o valor "João".

Exercício 03

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio03.py**. Após, acesse o valor correspondente à chave "nome" no dicionário "pessoas".

Exercício 04

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio04.py**. Após, adicione mais duas entradas no dicionário "pessoas" com as chaves "idade" e "cidade", e seus respectivos valores.

Exercício 05

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio05.py**. Após, verifique se a chave "altura" existe no dicionário "pessoas".

Dica: utilize o operador relacional *in* para fazer essa verificação.

Exercício 06

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio06.py**. Após, remova a entrada com a chave "cidade" do dicionário "pessoas".

Exercício 07

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio07.py**. Após, atualize o valor correspondente à chave "idade" para um novo valor por meio da sobrescrita. Na sequência, faça o mesmo procedimento com o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

Exercício 08

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio08.py**. Após, imprima todas as chaves do dicionário "pessoas".

Dica 1: utilize o laço de repetição *for* para execução dessa tarefa.

Dica 2: utilize o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

Exercício 09

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio09.py**. Após, imprima todos os valores do dicionário "pessoas".

Dica 1: utilize o laço de repetição *for* para execução dessa tarefa.

Dica 2: utilize o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

Exercício 10

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio10.py**. Após, imprima todas as chaves e valores do dicionário "pessoas" em pares.

Exercício 11

Crie um arquivo Python **exercicio11.py**. Após, faça a declaração de um dicionário "estoque" com três entradas: "item1" com valor 10, "item2" com valor 5 e "item3" com valor 20.

Exercício 12

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio12.py**. Após, calcule o total de itens no estoque.

Exercício 13

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio13.py**. Após, verifique se o item "item2" está no estoque.

Exercício 14

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio14.py**. Após, remova o item "item3" do estoque.

Dica: utilize o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

Exercício 15

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio15.py**. Após, adicione um novo item ao estoque com chave e valor de sua escolha.

Exercício 16

Crie um arquivo Python **exercicio16.py**. Após, faça a declaração de um dicionário vazio "frutas".

Exercício 17

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio17.py**. Após, adicione três entradas no dicionário "frutas" com as chaves sendo nomes de frutas e os valores sendo as quantidades dessas frutas.

Exercício 18

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio18.py**. Após, imprima o valor correspondente à chave "banana" no dicionário "frutas". Na sequência, faça um comentário explicando a saída de dados.

Exercício 19

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio19.py**. Após, caso a chave "banana" não exista, crie a chave e atribua a ela um valor qualquer. Na sequência, incremente seu valor em uma quantidade.

Dica 1: crie a chave "banana" e atribua a ela o valor 5, caso não exista no dicionário.

Dica 2: caso já tenha definida a chave "banana", experimente utilizar o método descrito na tabela de métodos manipuladores de dicionários.

Exercício 20

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio20.py**. Após, verifique se o dicionário "frutas" está vazio.

Exercício 21

Crie um arquivo Python **exercicio21.py**. Após, faça a declaração de um dicionário "notas" com cinco entradas, em que as chaves são nomes de alunos e os valores são suas respectivas notas.

Exercício 22

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio22.py**. Após, calcule a média das notas dos alunos.

Dica: lembre dos exercícios anteriores e de como fizemos para calcular somar itens a partir do laço de repetição *for*. É desejável o uso do laço para resolução desse exercício.

Exercício 23

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio23.py**. Após, encontre o aluno com a nota mais alta no dicionário "notas".

Dica: lembre dos exercícios anteriores e de como fizemos para calcular somar itens a partir do laço de repetição *for*. É desejável o uso do laço para resolução desse exercício.

Exercício 24

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio24.py**. Após, remova o aluno com a nota mais baixa do dicionário "notas".

Dica 1: lembre dos exercícios anteriores e de como fizemos para calcular somar itens a partir do laço de repetição *for*. É desejável o uso do laço para resolução desse exercício.

Dica 2: utilize o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

Exercício 25

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio24.py**. Após, verifique se um aluno chamado "Maria" está presente no dicionário "notas".

Exercício 26

Crie um arquivo Python **exercicio26.py**. Após, faça a declaração de um dicionário "agenda" com três entradas, em que as chaves são nomes de pessoas e os valores são seus respectivos números de telefone.

Exercício 27

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio27.py**. Após, adicione uma entrada com a chave "Pedro" e o valor do número de telefone correspondente.

Exercício 28

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio28.py**. Após, imprima todos os nomes de pessoas na agenda.

Exercício 29

A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio29.py**. Após, imprima todos os números de telefone na agenda.

Exercício 30

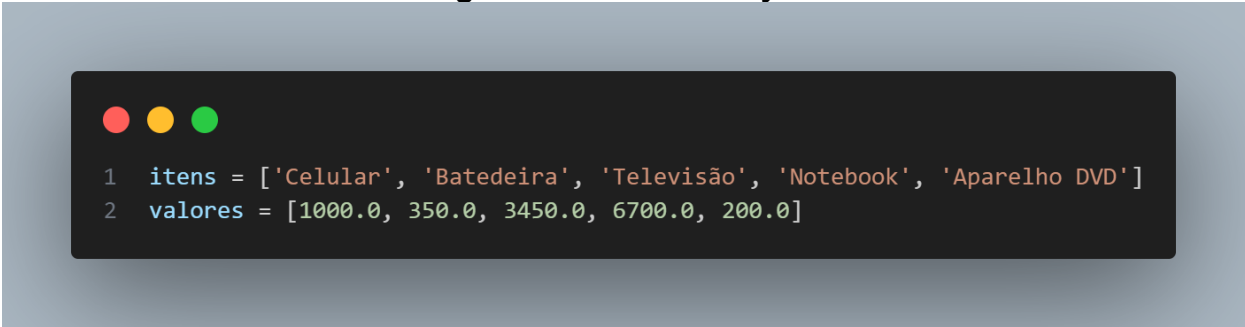
A partir do exercício desenvolvido anteriormente, crie um arquivo Python **exercicio29.py**. Após, remova todas as entradas da agenda. Na sequência, utilize a função print para exibir o dicionário após a remoção dos valores.

Dica: utilize o método reservado para esse fim descrito na tabela de métodos manipuladores de dicionários.

DESAFIO 1 – DICIONÁRIOS EM PYTHON

Observe o código a seguir:

Imagem 1 – Listas em Python



```
1 itens = ['Celular', 'Batedeira', 'Televisão', 'Notebook', 'Aparelho DVD']
2 valores = [1000.0, 350.0, 3450.0, 6700.0, 200.0]
```

Fonte: capturado pelos autores (2023).

Bruno Portela é um comerciante de produtos eletroeletrônicos de uma lojinha do Shopping Oiapoque. Recentemente, o Sr. Portela recebeu um novo carregamento de produtos e seus respectivos preços, conforme a Imagem 1. Sabendo que o item do índice zero [0] da lista "itens" corresponde ao item do índice zero [0] da lista "valores", o comerciante decidiu criar um dicionário em que suas chaves serão os nomes dos produtos e seus valores os preços. Assim, sempre que precisar saber o preço de um produto, poderá acessar o valor pela chave e recuperar a informação.

Sabendo disso, crie um arquivo Python **sistema.v0.1.py**. Após, declare as listas "itens"

e "valores" com os seus respectivos conteúdos. Na sequência, utilize a função descrita na tabela de métodos manipuladores de dicionários para criar um dicionário a partir das listas declaradas. Finalmente, crie um pequeno algoritmo que:

- Uma vez informado um valor a partir da função `input()`, verifique se esse valor é uma chave do dicionário.
- Caso o valor informado anteriormente esteja no dicionário, retorne a seguinte saída de dados: **"Produto: {nome_produto}.**
Valor: R\$ {valor_produto}."
- Caso o valor informado anteriormente não esteja no dicionário, pergunte ao usuário se ele deseja adicioná-lo.
- Caso o usuário informe que deseja adicionar um novo produto ao dicionário, pergunte o nome do produto e qual o valor (preço em R\$).
- Caso o usuário informe que não deseja adicionar um novo produto ao dicionário, retorne a lista de produtos e seus respectivos valores formatada conforme a saída de dados do item **(b)** do desafio.
- Após exibir a lista de produtos e de valores do dicionário, pergunte ao usuário se ele deseja atualizar o valor de algum produto. Se positivo, pergunte qual o valor será atualizado, faça a atualização do valor e exiba a lista atualizada formatada conforme o item (b) do desafio. Se negativo, encerre.

DESAFIO 2 – DICIONÁRIOS EM PYTHON

Sérgio de Freitas é um professor acadêmico responsável pela revista *Pedagogia em Ação*, periódico da área de educação da Pontifícia Universidade Católica de Minas Gerais (PUC-MG). Recentemente, Sérgio descobriu um problema no sistema de cadastro da revista, em que as informações do endereço do usuário não eram devidamente carregadas. Por essa razão, não era possível, por exemplo, após o usuário do site da revista informar o CEP, recuperar suas informações como logradouro, cidade, estado etc. Por essa razão, o professor contratou **VOCÊ** para resolver essa situação.

Tende em vista o problema anterior, faça um programa que pergunte ao usuário (função `input()`) o seu CEP conforme um dos padrões a seguir: **00000-000**, **00000.000** ou **00.000-000**. Caso o usuário informe qualquer outro caracter diferente de número, de traço (-) ou de ponto (.), você deve encerrar o programa e pedir que ele tente novamente. O mesmo deve acontecer caso o CEP informado contenha menos ou mais do que oito caracteres. Após receber o CEP do usuário, você deve usar a função nativa do Python (*built-in*) `replace()` para remover eventuais pontos, traços ou espaços do CEP. Adiante, após os devidos tratamentos da sua *string* CEP, implemente a linha de código a seguir:

```
url = f'https://viacep.com.br/ws/{cep}/json/'  
requisicao = requests.get(url)
```

Note que o CEP informado pelo usuário deve ser passado para a URL do site que fará a busca. Após, implemente uma variável do tipo *dicionário* (dict) '`informacoes_endereco`' e atribua a ela o seguinte valor: `requisicao.json()`. Finalmente, exiba as informações do endereço com os seguintes itens: rua, bairro, cidade, estado, CEP e o DDD da região. Seja criativo para exibir essas informações.

Dica: importe a biblioteca `requests` na primeira linha do seu código para resolver o exercício. Para isso, utilize o comando `import requests`.