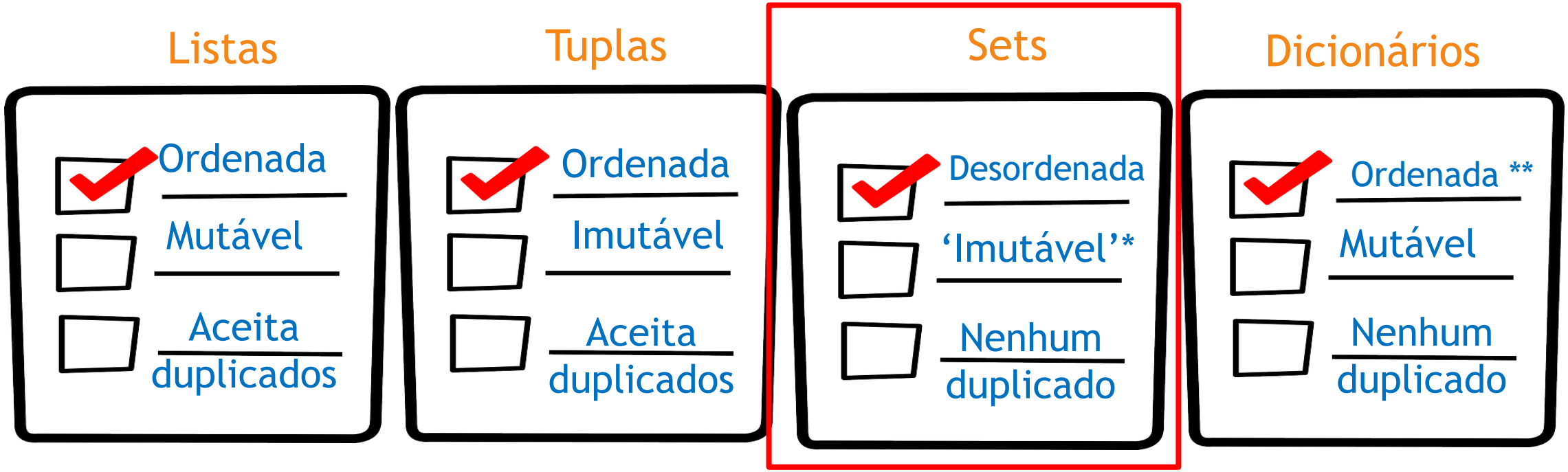
Coleções em Python
Sets e dicionários

Prof. Luciana Oliveira

Recife
2024

Coleções em Python

Existem quatro tipos de dados de coleção em Python:



*Set imutável: Não pode ser alterado mas, aceita remoção e adição de itens.

**Dicionário Ordenado: A partir da versão 3.7 do Python, os dicionários são *ordenados* . No Python 3.6 e anteriores, os dicionários não são *ordenados* .

Fonte: w3schools,2022

Tipos Sequenciais - Sets

Vamos conhecer um pouco sobre a coleção SET ☺

Uma SET tem característica de desordenação, ou seja, não tem índice especifico para os dados. Para criar uma set utiliza-se { } chaves inicializadas, caso contrário o Python irá identificar como um dicionário.

Declarando uma set

```
listaSet = { 4, 'texto', True, 2.5 }

print(listaSet)
print(type(listaSet))
```

Console

```
{True, 4, 2.5, 'texto' }

<class 'Set'>
```

Fonte: w3schools,2022

Como declarar um set?

O que acontece se eu não inicializar uma SET?

Declarando uma set

```
listaSet = { }

print(listaSet)
print(type(listaSet))
```

Console

```
{ }

<class  'dict'>
```

O Python identifica que é um dicionário e não uma SET

Fonte: w3schools,2022

Coleções em Python

O que fazer para deixar uma set não inicializada?

Usando a função set()

```
listaSet = set()

print(type(listaSet))
```

Console

```
<class 'set'>
```

Comandos SET

Cria uma lista set sem inicialização, comando: set()
Adiciona um elemento na coleção, comando: add()
Remove um elemento da coleção, comando remove()
Remove um elemento aleatório, comando pop()
Limpa todo o conjunto (o esvazia), comando clear()

O comando clear() também pode ser aplicado em lista. ☺

Usando o comando set e o método add

```
listaSet = set()
for recebe in range(3):
    nome = input('Nome: ')
    numero = int(input('numero: '))
    listaSet.add(nome)
    listaSet.add(numero)

print(listaSet)
```

Características do SET

Um set não possui valores repetidos. Ou seja, são únicos dentro de uma coleção set.

Declarando uma set com valores repetidos

```
listaSet = { 4, 'texto', 4, 4}

print(listaSet)
print(type(listaSet))
```

Console

```
{ 'texto', 4 }

<class 'Set'>
```

Fonte: w3schools,2022

Características do SET

Os conjuntos não são ordenados, portanto, você não pode ter certeza em qual ordem os itens aparecerão.

Declarando uma set

```
listaSet = { 1,2,3,4,5,6}
print(listaSet)
```

Descobrindo seu tamanho

```
listaSet = len(listaSet)
print(listaSet)
```

Console

```
{ 3,5,6,1,4,2}
6
```

Fonte: w3schools,2022

Características do SET

Os itens definidos não podem ser alterados, mas você pode remover itens e adicionar novos itens.

ADICIONAR = add()

```
planetas = {'Mercúrio'}
planetas.add('Vênus')
print(planetas)
```

Console

```
{'Vênus', 'Mercúrio'}
```

```
listaSet = set()
for x in range(2):
    planeta = input('Informe Planeta: ')
    listaSet.add(planeta)

print(listaSet)
```

Fonte: w3schools,2022

Características do SET

Os i*tens* definidos não podem ser alterados, mas você pode remover itens e adicionar novos itens.

REMOVER= remove()

```
planetas = {'Mercúrio', 'Vênus'}
planetas.remove('Vênus')
print(planetas)
```

Console

```
{ 'Mercúrio'}
```

Fonte: w3schools,2022

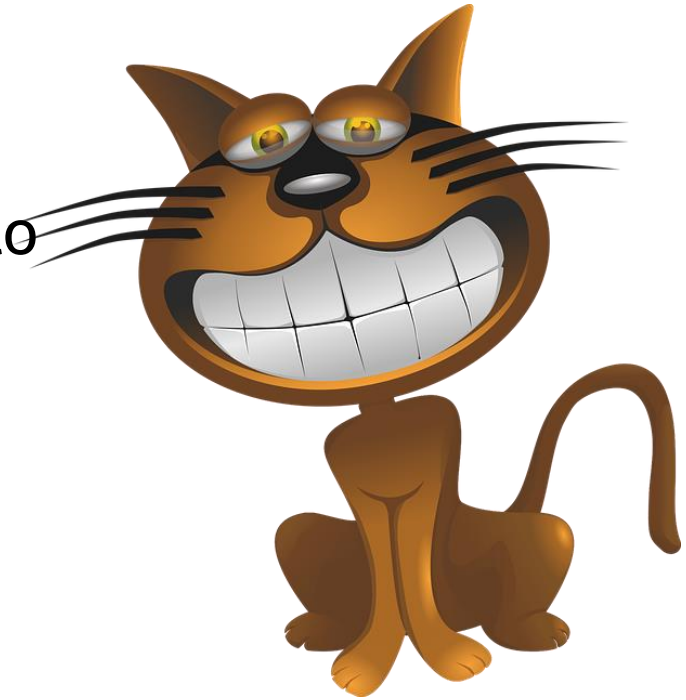
Coleções em Python

O que acontece se eu adicionar um elemento que já existe no SET?

Será desconsiderado!

O que acontece se eu remover um elemento que não existe no SET?

Dará um erro de chave!
KeyError



Verificando se um elemento está presente na lista set

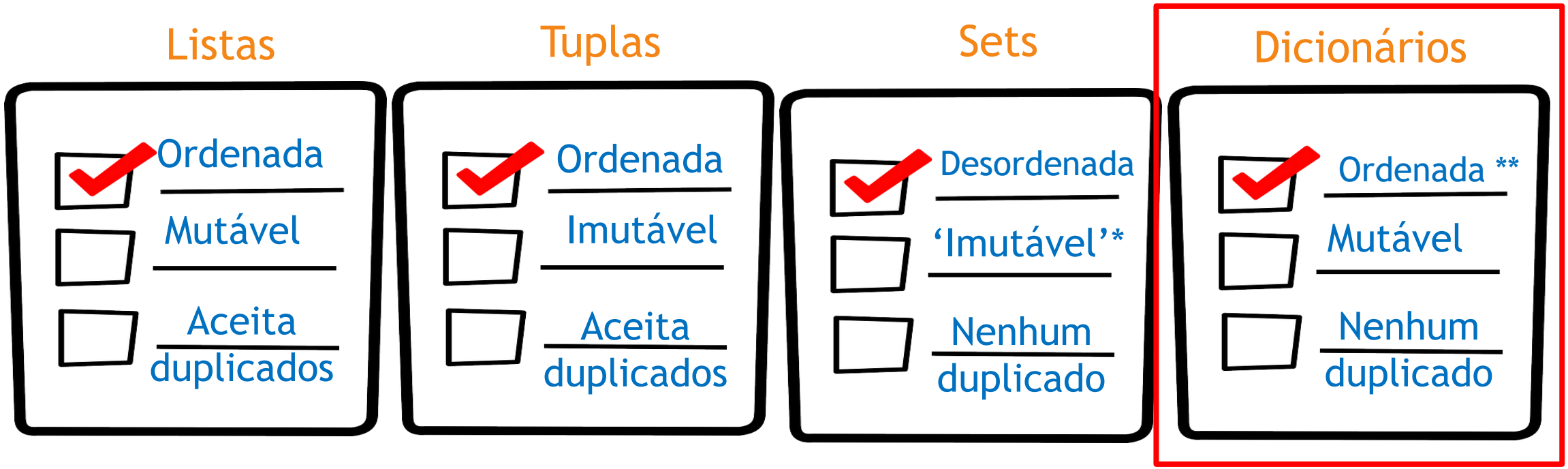
```
listaSet = {“maça", "banana", ”uva"}

print("banana" in listaSet)
```

O retorno será um boleano : True

Fonte: Reis,2021

Coleções em Python

Existem quatro tipos de dados de coleção em Python:



*Set imutável: Não pode ser alterado mas, aceita remoção e adição de itens.

**Dicionário Ordenado: A partir da versão 3.7 do Python, os dicionários são *ordenados* . No Python 3.6 e anteriores, os dicionários não são *ordenados* .

Fonte: w3schools,2022

Coleções em Python - Dicionários

Os dicionários são coleções de itens e seus elementos são armazenados de forma ordenada (a partir da versão 3.7).

Sua sintaxe básica é: {'chave': 'valor'}. Utiliza-se {} para delimitar o dicionário e a chave é separada do valor por dois pontos :

Coleções em Python - Dicionários

EXEMPLO DE UTILIZAÇÃO

Quero fazer uma lista de contatos telefônicos em Python. Para isso, preciso armazenar os números dos contatos. A princípio, colocaremos em uma lista.

```
telefones = ['1234-5678', '9999-9999', '8765-4321', '8877-7788']
```

Lista de contato

Coleções em Python - Dicionários

EXEMPLO DE UTILIZAÇÃO

Beleza. Temos os números de telefone armazenados. Mas... qual o sentido de termos uma lista de números sem saber de quem é?

Lista de contato telefônico

```
telefones = ['1234-5678', '9999-9999', '8765-4321', '8877-7788']
```

Coleções em Python - Dicionários

EXEMPLO DE UTILIZAÇÃO

Daí que surge o dicionário!

Com o dicionário irá ser possível que seus contatos tenham o nome e o telefone conectados, assim, ficará mais fácil você lembrar qual telefone é referente aos contatos.

Coleções em Python - Dicionários

Criando um dicionário sem inicialização

```
dicionario = dict()
```

ou

```
dicionario = { }

print(type(dicionario))
```

Console

```
<class ‘dict’>
```

Inicializando um dicionário

```
dicionario = {
    'Yan': '1234-5678‘,
    ‘Carla’: 23333
}
print(dicionario)
```

Coleções em Python - Dicionários

Exemplo de dicionário retornando apenas o valor

```
pessoas = {
    'nome': 'Ana',
    'sexo': 'F',
    'idade': 22
}
print(pessoas['idade'])
```

→ Retorna 22

Referenciando com o print formatado

```
print(f 'O {pessoas["nome"]} tem {pessoas["idade"]} anos.')
```

Chama os valores por colchetes

Usa-se aspas duplas

Coleções em Python - Dicionários

Como verificar todos os valores do dicionário?

Utilizando o método values()

```
pessoas = {
    ‘nome’: ‘Ana’,
    ‘sexo’: ‘F’,
    ‘idade’: 22}
print(pessoas.values())
```

Console

```
dict_values(['Ana', 'F', 22])
```

Coleções em Python - Dicionários

Como verificar todos as chaves do dicionário?

Utilizando o método keys()

```
pessoas = {
    ‘nome’: ‘Ana’,
    ‘sexo’: ‘F’,
    ‘idade’: 22}
print(pessoas.keys())
```

Console

```
dict_values([‘nome', ‘sexo', ‘idade’])
```

Coleções em Python - Dicionários

Como verificar as chaves e valores juntas do dicionário?

Utilizando o método items()

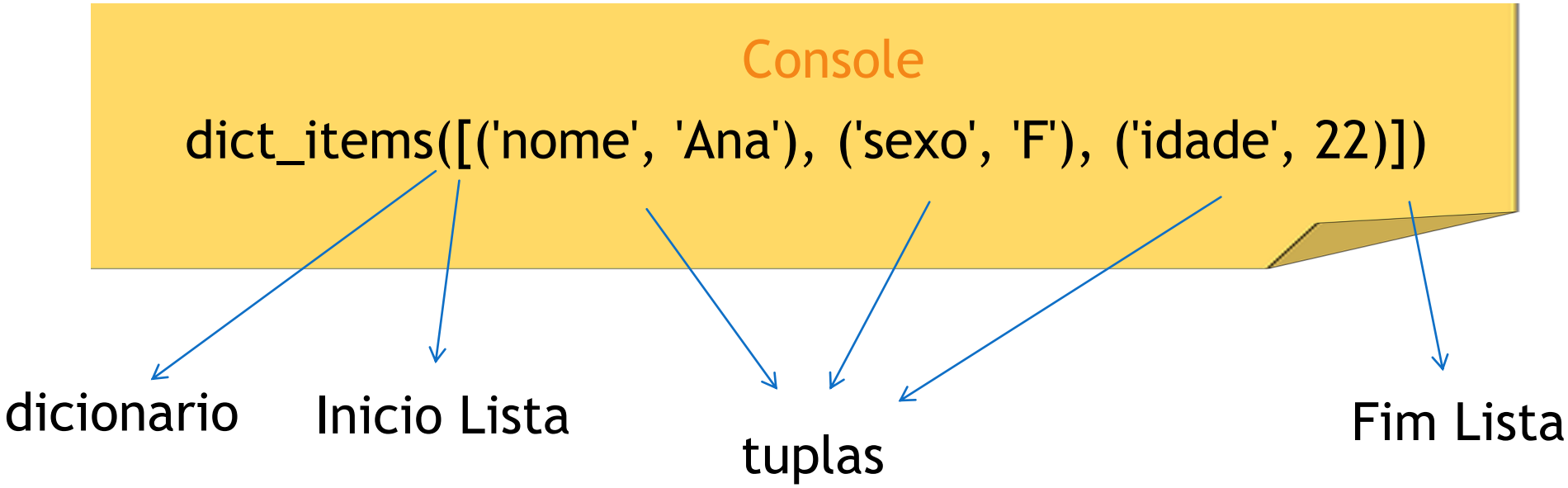
```
pessoas = {
    ‘nome’: ‘Ana’,
    ‘sexo’: ‘F’,
     ‘idade’: 22
}
print(pessoas.items())
```

Console

```
dict_items([('nome', 'Ana'), ('sexo', 'F'), ('idade', 22)])
```

Coleções em Python - Dicionários

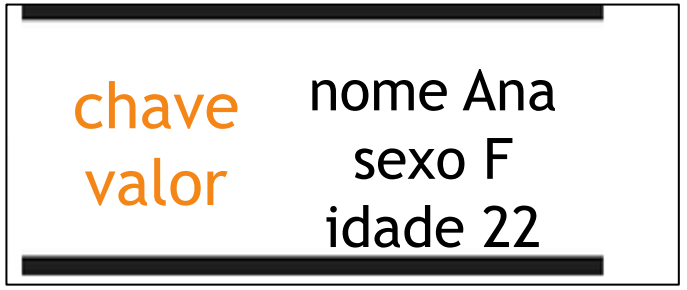
Vamos observar que quando retornamos os itens completos o dicionário representa em formato de lista e tuplas.



Coleções em Python - Dicionários

Retornando o dicionário por um laço de repetição for

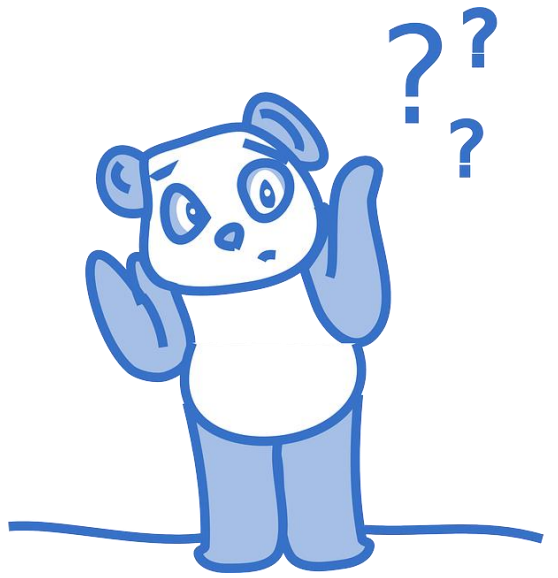
```
pessoas = {‘nome’: ‘Ana’, ‘sexo’: ‘F’, ‘idade’: 22}

for recebe in pessoas.values(): #recebendo valores
    print(recebe)

for recebe in pessoas.keys(): #recebendo as chaves
    print(recebe)

for chave, valor in pessoas.items(): #recebendo os dois
    print(chave, valor)
```



Coleções em Python - Dicionários

Criando um dicionário desta forma, o que acontece?

```
dicionario = { }
for x in range(2):
    nome = input('Nome: ')
    numero = int(input('Número: '))
    dicionario = {nome, numero}
print(dicionario)
```



Só irá retornar o último valor digitado

Coleções em Python - Dicionários

Adicionando os valores pelo o usuário, através do auxílio de uma lista

```
dicio =dict()
lista=list()
for recebe in range(2):
    dicio['nome'] = str(input('nome: '))
    dicio['idade'] = int(input('idade: '))
    lista.append(dicio.copy())

print(lista)
```

Criando um dicionário

Criando uma lista

Lista está adicionando uma cópia do dicionário

Coleções em Python - Dicionários

Outra forma de inserção de elementos no dicionário

Quero inserir nome e telefone no dicionário pelo usuário

```
dicionario = {}
for recebe in range(2):
    nome = str(input('Seu nome: '))
    telefone = int(input('Telefone: '))
    dicionario[nome] = telefone

print(dicionario)
```

Console

{‘joao': 12222, 'carla': 33344}

Coleções em Python - Dicionários

Agora veja no pycharm o que acontece se você colocar nomes iguais ou números iguais

```
dicionario = {}
for recebe in range(2):
    nome = str(input('Seu nome: '))
    telefone = int(input('Telefone: '))
    dicionario[nome] = telefone

print(dicionario)
```

Coleções em Python - Dicionários

O dicionário não aceita informações duplicadas, por essa razão, deveremos utilizar comando de verificação.

Comando in

Coleções em Python - Dicionários

```
dicionario = {}
for recebe in range(2):
    nome = input('Seu nome: ')
    telefone = int(input('Telefone: '))
    dicionario[nome] = telefone

    print('Dicionário Atual: ', dicionario)
    nome = input('Seu nome: ‘)

    if nome in dicionario:
        print("Já existe essa pessoa", nome)
    else:
        dicionario[nome] = telefone
        print(dicionario)
```

Comando in

Hora de praticar!
Atividade - Dicionários e Sets

REFERÊNCIAS

- REIS. F **Sets – Conjuntos em Python. 2021. Disponível em:** <http://www.bosontreinamentos.com.br/programacao-em-python/sets-conjuntos-em-python/> Acesso:20 set 2022.
- W3SCHOOLS. **Python Tutorial**.2022. Disponível em: https://www.w3schools.com/python/python_sets.asp