



IN

dev full stack- Python
Aula 2 - Listas

Listas



Python



Listas

- Coleção de valores indexada
- cada valor é identificado por um índice
- Os índices sempre começam em 0
- Representação sempre com colchetes

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]
```

```
print(type(alunos))
```

```
print(len(alunos))
```

```
print(alunos)
```

```
<class 'list'>
```

```
5
```

```
['Adalberto', 'Alejandro', 'Giovana', 'Fernando', 'Icaro']
```

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]
```

```
print(alunos[0])
```

```
print(alunos[1])
```

```
print(alunos[2])
```



1, 2, 3, 4

```
print(alunos[3])
```

```
print(alunos[4])
```

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]  
alunos[0] = "Antônio"
```

O valor do índice 0 foi substituído

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]  
  
i=0  
while (i<5):  
    print(alunos[i])  
    i+=1
```

Alternativa 1

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]  
  
for aluno in alunos:  
    print(aluno)
```

Alternativa 2

Listas

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]
```

```
for i in range(0,5,1):  
    print(alunos[i])
```

Alternativa 3

Inserir elemento na lista

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]
```

```
alunos.append("João Pedro")
```

```
for aluno in alunos:  
    print(aluno)
```

Adiciona elementos no final de uma lista.

Inserir elemento na lista

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro"]
```

```
alunos.insert(0, "Acássia")
```

```
for aluno in alunos:  
    print(aluno)
```

Adiciona elementos em uma posição indicada de uma lista.

Remover elemento da lista

- Pop():
 - Remove um elemento da posição indicada
- Remove():
 - Remove um item específico da lista



Remover elemento da lista

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro", "teste"]
```

```
alunos.pop(5)
```

```
for aluno in alunos:  
    print(aluno)
```

Apaga o item 5 da lista (lembre-se, começa do zero!)

Remover elemento da lista

```
alunos = ["Adalberto", "Alejandro", "Giovana", "Fernando", "Icaro", "teste"]
```

```
alunos.remove("teste")
```

```
for aluno in alunos:  
    print(aluno)
```

Atividade 1

Elaborar um algoritmo que captura 5 nomes na tela e adicione em uma lista. Exiba-os em seguida.



Atividade 2

Elaborar um algoritmo que captura 10 valores inteiros na tela e adicione-os em uma lista. Exiba apenas os ímpares.



Tuplas



Python



Tupla

- Sequências de valores, da mesma forma que listas;
- Diferenças:
 - Os valores de uma tupla, ao contrário de uma lista, são imutáveis;
 - Tuplas usam parênteses enquanto listas usam colchetes

```
>>> lista = [1, 2, 3, 4]
```

```
>>> tupla = (1, 2, 3, 4)
```

Tuplas

- Tupla vazia
`>>> tupla = ()`
- Tupla com um único elemento (note a necessidade da vírgula, mesmo sendo um único elemento)
`>>> tupla = (1,)`

Tuplas

```
dias = ("domingo", "Segunda", "terça", "quarta", "quinta", "sexta", "sabado")  
print(dias[0])
```

Exibir dados: Igual a lista

Tuplas: Usando Slices

```
dias = ("domingo", "Segunda", "terça", "quarta", "quinta", "sexta", "sabado")  
print(dias[0:7])
```



Primeira posição de exibição : qual posição vai parar (não incluída)

Tuplas

```
dias = ("domingo", "Segunda", "terça", "quarta", "quinta", "sexta", "sabado")  
dias [0] = "DiaNovo"
```

Como são imutáveis não se pode atualizar o valor da tupla.

```
Traceback (most recent call last):  
  File "C:/Users/frana/Documents/Python/tuplas.py", line 3, in <module>  
    dias [0] = "DiaNovo"  
TypeError: 'tuple' object does not support item assignment
```

Operadores Básicos sobre Tuplas

Expressão	Resultado	Descrição
<code>len((1,2,3))</code>	3	Número de elementos que a tupla contém
<code>(1, 2, 3) + (4, 5, 6)</code>	<code>(1, 2, 3, 4, 5, 6)</code>	Concatenação
<code>(1,) * 4</code>	<code>(1,1,1,1)</code>	Repetição
<code>3 in (1, 2, 3)</code>	True	Pertencimento
<code>for x in (1,2,3): print(x)</code>	1 2 3	Iteração

Dicionários



Python



Dicionário

- Representam coleções de dados que contém na sua estrutura um conjunto de pares chave/valor
- Cada chave individual tem um valor associado
- A associação nos dicionários é feita por meio de uma chave que faz referência a um valor



Dicionário

```
dados_aluno = {  
    'nome': 'Giovana',  
    'endereco': 'Rua abc',  
    'telefone': '71 99999999'  
}  
  
print("Nome: "+dados_aluno['nome'])
```

Dicionário - Adicionando campo

```
dados_aluno = {  
    'nome': 'Giovana',  
    'endereco': 'Rua abc',  
    'telefone': '71 9999999'  
}
```

#adicionando novo campo

```
dados_aluno['cpf'] = '123456'  
  
print(dados_aluno)
```

Nome: Giovana

```
{'nome': 'Giovana', 'endereco': 'Rua abc', 'telefone': '71 9999999', 'cpf': '123456'}
```

Dicionário - Removendo campo

```
dados_aluno = {  
    'nome': 'Giovana',  
    'endereco': 'Rua abc',  
    'telefone': '71 99999999',  
    'idade': '18'  
}
```

#adicionando novo campo

```
dados_aluno.pop('idade')
```

```
print(dados_aluno)
```

```
{'nome': 'Giovana', 'endereco': 'Rua abc', 'telefone': '71 99999999'}
```

Dicionário : Chave? Valor?

Qual dado deve servir como chave?

Por qual elemento quero fazer o acesso?

Qual dado deve servir como conteúdo?

Qual(is) valor(es) quero associar à chave?

Acesso a dados de Dicionário

- Feito sempre pela chave
- Ex:

```
dados_aluno = {  
    'nome': 'Giovana',  
    'endereco': 'Rua abc',  
    'telefone': '71 99999999',  
    'idade': '18'  
}  
  
|  
print(dados_aluno['nome'])
```

Dicionário com múltiplos valores

```
dados_aluno = {  
    'nome': 'Giovana',  
    'endereco': 'Rua abc',  
    'telefone': '71 9999999',  
    'idade': '18'}, {  
    'nome': 'Icaro',  
    'endereco': 'Rua cba',  
    'telefone': '71 8888888',  
    'idade': '25'}  
  
i=0  
while (i<2):  
    print(dados_aluno[i]['nome'])  
    i+=1
```

Acrescentar novos valores

```
dados_aluno = {}  
  
dados_aluno["nome"] = input("Informe o nome: ")  
dados_aluno["endereco"] = input("Informe o endereco: ")  
dados_aluno["telefone"] = input("Informe o telefone: ")  
  
print(dados_aluno["nome"])
```


Copy(): Listas e Dicionários

- Retorna um outro dicionário com os mesmos pares chave/conteúdo

```
>>> d1 = {"Joao": 10, "Maria": 20}
>>> d2 = d1.copy()
>>> d2["Pedro"] = 30
>>> d1["Joao"] = 40
>>> d1
{"Joao": 40, "Maria": 20}
>>> d2
{"Pedro": 30, "Joao": 10, "Maria": 20}
```

Copy(): Listas e Dicionários

- Se conteúdo for lista, o que é copiado é apenas a referência...

```
>>> d1 = {"Joao": [1, 2], "Maria": [3, 4]}
>>> d2 = d1.copy()
>>> d2["Pedro"] = [5, 6]
>>> d1["Joao"] += [3]
>>> d1
{"Joao": [1, 2, 3], "Maria": [3, 4]}
>>> d2
{"Pedro": [5, 6], "Joao": [1, 2, 3],
 "Maria": [3, 4]}
```

Clear()

- Remove todos os elementos do dicionário

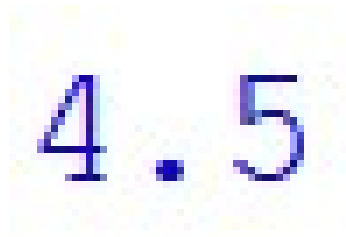
```
>>> idades = {"Joao":10, "Maria":12}
>>> idadesCrianças = idades
>>> idades.clear()
>>> idades
{}
>>> idadesCrianças
{}

```

Função dict()

- Usada para criar dicionários
- Pode receber dois tipos de parâmetros

```
produtos = dict([(10, 4.5), (20, 5.99)])  
print(produtos[10])
```



4.5

Items()

- Retorna uma lista com todos os pares chave/conteúdo do dicionário no formato de tupla

```
>>> notas = { "Joao": [9.0, 8.0], "Maria":  
[10.0] }  
>>> notas.items()  
[("Joao", [9.0, 8.0]), ("Maria", [10.0])]
```

keys()

- Retorna uma lista com todas as chaves do dicionário

```
>>> notas = {"Joao": [9.0, 8.0], "Maria":  
[10.0]}  
>>> notas.keys()  
["Joao", "Maria"]
```

values()

- Retorna uma lista com todos os valores do dicionário

```
>>> notas = {"Joao": [9.0, 8.0],  
             "Maria": [10.0]}  
>>> notas.values()  
[[9.0, 8.0] , [10.0]]
```

popitem()

- Retorna e remove o último par chave/valor do dicionário
- Pode ser usado para iterar sobre todos os elementos do dicionário

```
>>> notas = {"Joao": [9.0, 8.0],  
             "Maria": [10.0]}  
>>> notas.popitem()  
{"Maria": [10.0]}  
>>> notas  
{"Joao": [9.0, 8.0]}
```


pop()

- Retorna e remove um par chave/valor do dicionário identificado pela chave, ou retorna uma mensagem quando não encontra a chave.

```
aluno = dict([('Ana',3),('Pedro',2),('Joao',7),('Edu',5)])  
print(aluno.pop("Edu","Não achei"))
```

Iterando com Dicionários

- A iteração em elementos de um dicionário é feita a partir da chave
- Lembre-se de que com dicionários não temos ordem pré-definida

```
notas = {"Joao":[9.0,8.0], "Maria":[10.0]}  
for nome in notas:  
    media = sum(notas[nome])/len(notas[nome])  
    print("A média de ", nome, " é: ", media)
```

Atividade 1

Escreva uma função que conta a quantidade de vogais em um texto e armazena tal quantidade em um dicionário, onde a chave é a vogal considerada.

Atividade 2

Escreva um programa que lê duas notas de vários alunos e armazena tais notas em um dicionário, onde a chave é o nome do aluno. A entrada de dados deve terminar quando for lida uma string vazia como nome. Escreva uma função que retorna a média do aluno, dado seu nome.

The logo consists of the letters 'IN' in a white, serif font, centered within a solid red square. The background of the entire image is a dark red with abstract, layered geometric shapes in lighter shades of red, creating a sense of depth and movement.

71 3901 1052 | 71 9 9204
0134

@infinity.school

www.infinityschool.com.br

Salvador Shopping Business | Torre Europa Sala 310
Caminho das Árvore, Salvador - BA CEP: 40301-155