Aula 06

LIP

Linguagem de Programação

Python

Sequências

As sequencias permitem armazenar múltiplos itens e tipos dentro de uma única unidade ou variável. Listas são variáveis completamente mutáveis.

Python

Listas

Podemos utilizar a estrutura de lista em Python para armazenar múltiplos dados.

Listas podem ser criadas de forma implícita, listando os elementos entre colchetes.

Podemos também declarar uma lista de maneira explícita utilizando a função list.

Listas

Python

Exemplos:

```
#LISTAS EXPLICITAS
num=[1,2,3,4,5] #Somente um tipo
dados=[1.34, "SENAI", True, -100] #tipos diversos
escola=list(["predio", "professores", "alunos", 501, False]) #criação de listas
```

Python

Listas

Podemos acessar um elemento em uma lista indicando a sua posição (o primeiro elemento fica na posição 0 da lista)

```
num=[1,2,3,4,5] #Somente um tipo
dados=[1.34, "SENAI", True, -100] #tipos diversos
escola=list(["predio", "professores", "alunos", 501, False]) #criação de listas
print(num[0],num[1],sep="----")
print(dados[2], dados[3], sep="----")
print(escola[0], escola[3], sep="----")
print(len(escola))
```

```
1----2
True-----100
predio----501
```

Listas

Python

A função len() retorna o número de elementos de uma lista.

```
num=[1,2,3,4,5] #Somente um tipo
dados=[1.34, "SENAI", True, -100] #tipos diversos
escola=list(["predio", "professores", "alunos", 501, False]) #criação de listas
print(len(num))
print(len(dados))
print(len(escola))
```

```
5
4
5
```

Listas

Python

Podemos acrescentar novos itens ou substituir elementos de uma lista. Nesse exemplo substituindo o elemento:

```
exemplo1=[1,2,3,4,5,6]
print(exemplo1)
exemplo1[2]="senai"
print(exemplo1)
exemplo1[2:5]=["cfp","alunos",501]
print(exemplo1)
```

```
[1, 2, 3, 4, 5, 6]
[1, 2, 'senai', 4, 5, 6]
[1, 2, 'cfp', 'alunos', 501, 6]
```

Python

Listas

Para acrescentar podemos usar os métodos append ou insert. No método append o item é inserido no fim da lista e no insert é possível escolher a posição de sua inserção

```
exemplo2=["cachorro", "gato", "peixe"]
print(exemplo2)
exemplo2.append("periquito")
print(exemplo2)
```

```
['cachorro', 'gato', 'peixe']
['cachorro', 'gato', 'peixe', 'periquito']
```

Listas

Python

```
exemplo3=[0,1,3,4,5]
print(exemplo3)
exemplo3.insert(2,2) # primeiro argumento é a posição
                     # segundo é o elemento de inserção
print(exemplo3)
```

```
[0, 1, 3, 4, 5]
[0, 1, 2, 3, 4, 5]
```

Listas

Python

Da mesma forma que inserimos, trocamos, podemos também remover um item da lista, através do método remove.

```
print(exemplo2)
exemplo2.remove("periquito")
print(exemplo2)
```

```
['cachorro', 'gato', 'peixe', 'periquito']
['cachorro', 'gato', 'peixe']
```

Listas

Python

Para verificar a existência de um item numa lista;

```
paises=["Brasil", "Argentina", "Chile", "Uruguai"]
dado=input("Digite o nome de um país: ")
if (dado in paises): # Se encontrar o país, tem que ser a mesmissima grafia, retorna sim,
                     # caso contrário não
   print("Esse pais existe na lista")
else:
   print("Esse não!")
```

```
Digite o nome de um país: Brasil
Esse pais existe na lista
```

```
Digite o nome de um país: EUA
Esse não!
```

Listas

Pode também verificar a posição do item numa lista através do método index().

```
if (dado in paises): # Se encontrar o país, tem que ser a mesmissima grafia, retorna sim,
                     # caso contrário não
   print(f"Esse pais existe na lista, é o {paises.index(dado)+1:1}° item dessa lista")
else:
   print("Esse não!")
```

```
Digite o nome de um país: Brasil
Esse pais existe na lista, é o 1° item dessa lista
```

Python

Listas

Python

Outras funções importantes para manusear listas:

```
paises=["Brasil", "Argentina", "Chile", "Uruguai"]
#pop()
print(paises.pop(0)) #Também remove porém pela posição do elemento e retorna o elemento removido
print(paises)
print(paises.pop()) #se não por nada apaga o último
print(paises)
paises = ["Brasil", "brasil", "Brazil", "Brasil"]
#count()
print(paises.count("Brasil")) #retorna a quantidade de vezes que o item aparece na lista
#copy()
numeros=[2,4,5,1,5,10,-100]
numeros1=numeros.copy() #usado para copiar uma lista exsitente
numeros2=numeros.copy()
print(numeros1)
```

Listas

Outras funções importantes para manusear listas:

Python

```
#ordenando listas
numeros1.sort() # ordenação de forma crescente
print(numeros1)
numeros2=numeros1.copy()
numeros2.reverse() # ordenação de forma crescente
print(numeros2)
print(min(numeros1))#retorna o menor item da lista
print(max(numeros2))#retorna o maior item da lista
```

Python

Listas

Quadro resumo de funções built-in:

Método	Parâmetros	Resultado	Descrição
append	item	mutador	Acrescenta um novo item no final da lista
insert	posição, item	mutador	Insere um novo item na posição dada
pop	nenhum	híbrido	Remove e returno o último item
pop	posição	híbrido	Remove e retorna o item da posição.
sort	nenhum	mutador	Ordena a lista
reverse	nenhum	mutador	Ordena a lista em ordem reversa
index	item	retorna idx	Retorna a posição da primeira ocorrência do item
count	item	retorna ct	Retorna o número de ocorrências do item
remove	item	mutador	Remove a primeira ocorrência do item

Python

Listas

É possível concatenar e repetir itens de uma lista como é feito com strings:

```
frutas = ["maca", "laranja", "banana", "cereja"]
print([1, 2] + [3, 4])
print(frutas + [6, 7, 8, 9])

print([0] * 4)
print([1, 2, ["ola", "adeus"]]*2)
```

```
[1, 2, 3, 4]
['maca', 'laranja', 'banana', 'cereja', 6, 7, 8, 9]
[0, 0, 0, 0]
[1, 2, ['ola', 'adeus'], 1, 2, ['ola', 'adeus']]
```

Python

Tuplas

Vimos que é possível adicionar, remover ou alterar elementos de uma lista.

Já tuplas, uma vez criadas, não permitem modificações.

Ou seja, tuplas são listas imutáveis. • Listas e tuplas podem armazenar.

Tuplas

Python

Dados homogêneos
(Exemplos: listas/tuplas de emails, salários ou notas).

Dados heterogêneos
(Exemplo: cadastro de uma pessoa em uma academia com as informações de nome, idade e peso).

Tuplas

Python

- Podemos declarar uma tupla utilizando ( ).

```
variavel = (elemento_1, elemento_2, ..., elemento_n)
```

- Também podemos declarar uma tupla de maneira explícita utilizando a função `tuple`.

```
variavel = tuple([elemento_1, elemento_2, ..., elemento_n])
```

Python

Tuplas

- Tudo o que vimos para listas também podemos aplicar para tuplas, exceto operações, métodos ou funções que adicionem, removam ou modifiquem elementos.
- Selecionando o primeiro elemento de uma tupla:

```
1 letras = ("A", "B", "C", "D", "E", "F", "G", "H")
2 print(letras[0])
3 # A
```

- Selecionando o último elemento de uma tupla:

```
1 letras = ("A", "B", "C", "D", "E", "F", "G", "H")
2 print(letras[-1])
3 # H
```

Tuplas

Python

```
### TUPLAS

##Criando tuplas a partir de entradas
n = int(input("Quantos números terão a tupla: "))
tupla = ()
for i in range(n):
    x = int(input("Entre com um número: "))
    tupla = tupla + tuple([x])
print(tupla)
```

A ação de concatenar, junto com o construtor tuple, pode em parte alterar uma tupla existente, porém, sem condições de remover itens dentro dela

```
Quantos números terão a tupla: 3
Entre com um número: 1
Entre com um número: 2
Entre com um número: 3
(1, 2, 3)
```

Tuplas

Python

```
n = int(input("Quantos números terão a tupla: "))
tupla = ("Senai",[1,2], 0.56, 90)
print(tupla)
for i in range(n):
    x = int(input("Entre com um número: "))
    tupla = tupla + tuple([x])
print(tupla)
```

A ação de concatenar, junto com o construtor tuple, pode em parte alterar uma tupla existente, porém, sem condições de remover itens dentro dela

```
Quantos números terão a tupla: 3
('Senai', [1, 2], 0.56, 90)
Entre com um número: 1
Entre com um número: 2
Entre com um número: 3
('Senai', [1, 2], 0.56, 90, 1, 2, 3)
```