Educação Profissional Paulista

Técnico em Ciência de Dados

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO

Variáveis e tipos de dados

Tuplas e conjuntos

Aula 1

Código da aula: [DADOS]ANO1C2B2S14A1

Exposição

Objetivo da Aula

Aprender o conceito de tuplas, entender sua estrutura e suas funções.

Competências da Unidade (Técnicas e Socioemocionais)

- Ser proficiente em linguagens de programação, para manipular e analisar grandes conjuntos de dados;
- Usar técnicas para explorar e analisar dados, aplicar modelos estatísticos, identificar padrões, realizar inferências e tomar decisões baseadas em evidências;
- Colaborar efetivamente com outros profissionais, como cientistas de dados e engenheiros de dados; trabalhar em equipes multifuncionais, colaborando com colegas, gestores e clientes.

Recursos Didáticos

- Recurso audiovisual para a exibição de vídeos e imagens;
- Acesso ao laboratório de informática e/ou à internet;
- Software Anaconda/Jupyter Notebook instalado ou similar.

Duração da Aula

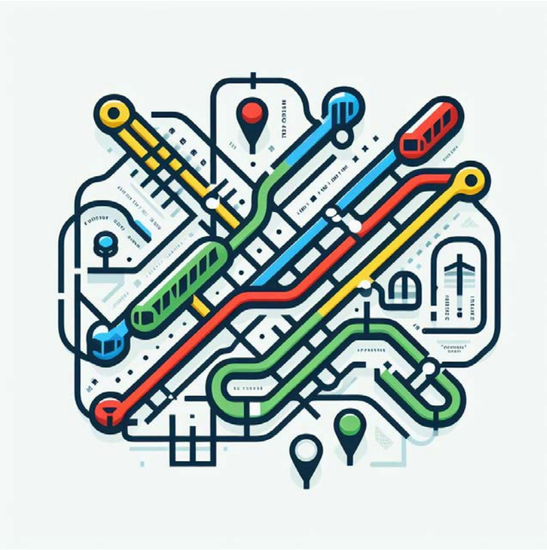
50 minutos.

Exposição

Motivação

Você mudou de cidade e quer saber se tem um metrô perto da sua casa. Seu amigo passou o código abaixo:

O que entendeu do código? O que é (-23.5234, -46.6731)?



Elaborado especialmente para o curso com a ferramenta Microsoft Copilot.

```python
coordenadas_estacoes_metro = [
    (-23.5505, -46.6333),
    (-23.5678, -46.6522),
    (-23.5234, -46.6731),
    (-23.5489, -46.6112)
]

for indice, coordenada in enumerate(coordenadas_estacoes_metro, start=1):
    lat, lon = coordenada
    print(f"Estação {indice}: Localização - Latitude: {lat}, Longitude: {lon}")

posicao_procurada = (-23.5234, -46.6731)
if posicao_procurada in coordenadas_estacoes_metro:
    print(f"Uma estação de metrô está localizada em {posicao_procurada}")
else:
    print(f"Nenhuma estação de metrô encontrada em {posicao_procurada}")
```

```
Estação 1: Localização - Latitude: -23.5505, Longitude: -46.6333
Estação 2: Localização - Latitude: -23.5678, Longitude: -46.6522
Estação 3: Localização - Latitude: -23.5234, Longitude: -46.6731
Estação 4: Localização - Latitude: -23.5489, Longitude: -46.6112
Uma estação de metrô está localizada em (-23.5234, -46.6731)
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Tuplas

Em Python, uma **tupla é uma estrutura de dados que armazena uma coleção ordenada e imutável de elementos**.

Tome nota

A principal característica das tuplas é que, uma vez criadas, elas não podem ser modificadas. Isso significa que você não pode adicionar, remover ou modificar elementos individualmente em uma tupla após sua criação.

Exposição

Tuplas

As tuplas são delimitadas por parênteses e podem conter elementos de tipos diferentes. Por exemplo:

```
minha_tupla = (1, "Olá", 3.14, True)
```

Elaborado especialmente para o curso com Jupyter Notebook.

Neste exemplo, minha_tupla é uma tupla que contém quatro elementos:

- um inteiro;
- uma *string*;
- um número de ponto flutuante;
- um booleano.

Exposição

Tuplas

- ✓ São frequentemente utilizadas em situações em que é necessário representar um conjunto de valores relacionados de forma **imutável**.
- ✓ Podem ser utilizadas para retornar múltiplos valores a partir de uma função, como chaves em **dicionários** e até mesmo para representar **coordenadas** em geometria.
- ✓ Ao usá-las, você está informando a outros desenvolvedores que esses dados são **constantes** e **não** devem ser alterados, mantendo um código claro e limpo.

Tome nota

A imutabilidade das tuplas garante que os dados não serão alterados acidentalmente, o que é útil para garantir a integridade dos dados ao longo de um programa.

Exposição

Criar uma tupla

Para criar uma tupla em Python, você pode utilizar parênteses () e inserir os elementos separados por vírgulas.

```
minha_tupla = (1, "Olá", 3.14, True)
print(minha_tupla)
```

```
(1, 'Olá', 3.14, True)
```

→ Tupla com diferentes tipos de elementos

```
numeros = (10, 20, 30, 40, 50)
print(numeros)
```

```
(10, 20, 30, 40, 50)
```

→ Tupla com números inteiros

```
tupla_vazia = ()
print(tupla_vazia)
```

```
()
```

→ Tupla vazia

```
tupla_simples = (42,)
print(tupla_simples)
```

```
(42,)
```

→ Tupla com um único elemento (necessário adicionar uma vírgula)

```
outra_tupla = 1, 2, 3, 4
print(outra_tupla)
```

```
(1, 2, 3, 4)
```

→ Você também pode criar tuplas sem parênteses, usando apenas vírgulas

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Acessar uma tupla

Para acessar os elementos de uma tupla, é utilizada a **indexação**, da mesma forma que em listas:

Tome nota

Lembre-se de que, devido à sua imutabilidade, as tuplas são mais eficientes em termos de uso de memória e podem ser uma escolha adequada quando você precisa garantir que os dados não sejam alterados após a criação.

```
1  coordenadas = (2, 3)
2
3  x = coordenadas[0]  # x recebe o valor 2
4  y = coordenadas[1]  # y recebe o valor 3
5
6  print(x)
7  print(y)
```

```
2
3
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Substituir um valor em uma tupla

Vamos tentar alterar o elemento da tupla coordenadas = (2, 3) para (1, 3):

```
coordenadas = (2, 3)

coordenadas[0] = 1 # tentando substituir o elemento de índice 0 que vale 2 por 1
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[12], line 3
      1 coordenadas = (2, 3)
----> 3 coordenadas[0] = 1 # tentando substituir o elemento de índice 0 que vale 2 por 1

TypeError: 'tuple' object does not support item assignment
```

Elaborado especialmente para o curso com Jupyter Notebook.

Reflita

Por que isso acontece? Por que deu erro?

Exposição

Substituir um valor em uma tupla

```python
# Criando uma tupla
minha_tupla = (1, 2, 3, 4, 5)

# Convertendo a tupla para uma lista (pois listas são mutáveis)
lista_modificavel = list(minha_tupla)

# Substituindo o segundo elemento (índice 1) por um novo valor
lista_modificavel[1] = 10

# Convertendo a lista de volta para uma tupla
minha_tupla_modificada = tuple(lista_modificavel)

# Exibindo a tupla modificada
print(minha_tupla_modificada)
```

```
(1, 10, 3, 4, 5)
```

Em Python, uma vez que uma tupla é criada, ela é imutável, o que significa que seus elementos não podem ser modificados.

Portanto, você não pode substituir diretamente um valor em uma tupla após sua criação.

Mas **você pode criar uma nova tupla com o valor modificado**, como no exemplo.

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Percorrer uma tupla

Você pode percorrer uma tupla em Python utilizando um **loop**, como o **loop for**.

Este código imprimirá cada elemento da tupla em uma linha separada.

```
1 minha_tupla = (1, 2.9, 'palavra', True)
2
3 for elemento in minha_tupla:
4     print(elemento)
```

```
1
2.9
palavra
True
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Percorrer uma tupla

Se você precisar acessar os índices dos elementos enquanto percorre a tupla, você pode usar a função **enumerate**:

Neste exemplo, o índice conterá o **índice do elemento atual** e o elemento conterá o **valor do elemento na tupla**.

```
1  minha_tupla = (10, 20, 30, 40, 50)
2
3  # Percorrendo a tupla com índices usando a função enumerate
4  for indice, elemento in enumerate(minha_tupla):
5      print(f"Índice: {indice}, Elemento: {elemento}")
```

```
Índice: 0, Elemento: 10
Índice: 1, Elemento: 20
Índice: 2, Elemento: 30
Índice: 3, Elemento: 40
Índice: 4, Elemento: 50
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Tuplas aninhadas

Em Python, você pode criar **tuplas aninhadas**, o que significa que você pode ter uma tupla contendo **outras tuplas** como elementos.

Isso permite representar estruturas de dados mais complexas. Vamos ver um exemplo:

```
minha_tupla = ((1,2,3), 1, [1,4,'sim'], True)

print(type(minha_tupla))
print(type(minha_tupla[0]))
print(type(minha_tupla[2]))
```

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

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Tuplas aninhadas

Neste exemplo, tupla_aninhada é uma tupla que contém três tuplas como elementos. Você pode acessar os elementos da tupla aninhada usando **índices múltiplos**.

```
# Tupla aninhada
tupla_aninhada = ((1, 2, 3), ('a', 'b', 'c'), (True, False, True))

# Acessando elementos da tupla aninhada
print(tupla_aninhada[0])  # Saída: (1, 2, 3)
print(tupla_aninhada[1][1])  # Saída: 'b'
print(tupla_aninhada[2][2])  # Saída: True
```

```
(1, 2, 3)
b
True
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Descompactando uma tupla

Em Python, você pode descompactar uma tupla atribuindo seus elementos a variáveis individuais. Isso é conhecido como **desempacotamento de tupla**. Aqui está um exemplo simples:

```
# Tupla para descompactar
minha_tupla = (10, 20, 30)

# Descompactando a tupla
a, b, c = minha_tupla

# Exibindo os valores descompactados
print("a:", a)
print("b:", b)
print("c:", c)
```

```
a: 10
b: 20
c: 30
```

Neste exemplo, a tupla (10, 20, 30) é descompactada nas variáveis a, b e c.

Cada valor da tupla é atribuído à variável correspondente na ordem em que aparecem na tupla.

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Descompactando uma tupla

O desempacotamento é feito de forma paralela, o que significa que cada variável recebe o valor correspondente na ordem em que aparece na tupla. Outro exemplo:

```
# Definindo a tupla com seis elementos de tipos diferentes
minha_outra_tupla = (1, 2.9, 'palavra', True, 'sim', 2)

# Desempacotando a tupla e atribuindo valores a variáveis
var1, var2, var3, var4, _, _ = minha_outra_tupla

# Imprimindo os valores desempacotados
print(var1, var2, var3, var4)
```

```
1 2.9 palavra True
```

Se houver mais variáveis adicionais no desempacotamento do que elementos na tupla, você pode usar **um sublinhado (_)** como convenção para indicar que você não está interessado em armazenar esses valores.

Elaborado especialmente para o curso com Jupyter Notebook.

Tome nota

O desempacotamento é útil ainda em laços de repetição que iteram sobre uma sequência de tuplas, permitindo desempacotar os elementos diretamente nas variáveis de loop.

Exposição

Operações com tuplas

- Concatenar tuplas:

```
1 nova_tupla = minha_tupla + (4, 5, 6)
2 nova_tupla
```

```
(10, 20, 30, 4, 5, 6)
```

- Repetir uma tupla:

```
1 repetida = minha_tupla * 2  # Repete a tupla duas vezes
2 repetida
```

```
(10, 20, 30, 10, 20, 30)
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Funções de tuplas

As tuplas em Python são versáteis e podem ser utilizadas em diversas situações. A seguir estão algumas funções e operações comuns relacionadas a tuplas:

- Comprimento (tamanho) da tupla:

```
1 tamanho = len(minha_tupla)  # Retorna o número de elementos na tupla
```

- Encontrar índice de um elemento:

```
1 indice = minha_tupla.index('texto')  # Retorna o índice do elemento 'texto'
```

Elaborado especialmente para o curso com Jupyter Notebook.

Exposição

Funções de tuplas

- Contar a ocorrência de um elemento:

```
1 ocorrencias = minha_tupla.count(2)  # Conta quantas vezes o elemento 2 aparece
```

- Converter string em tupla:

```
1 string = "abc"
2 tupla_a_partir_string = tuple(string)
```

Elaborado especialmente para o curso com Jupyter Notebook.

Registro

Vamos fazer um **quiz**

Qual é a principal característica de uma tupla em Python?

- Pode ser modificada após a criação.
- Pode conter apenas números inteiros.
- É uma estrutura de dados mutável.
- É uma sequência ordenada e imutável de elementos.

Registro

Vamos fazer um **quiz**

Qual é a principal característica de uma tupla em Python?

- ✗ **Pode ser modificada após a criação.**
- ✗ **Pode conter apenas números inteiros.**
- ✗ **É uma estrutura de dados mutável.**
- ✓ **É uma sequência ordenada e imutável de elementos.**

FEEDBACK GERAL DA ATIVIDADE

A principal característica das tuplas é a imutabilidade, que as torna úteis para representar conjuntos ordenados de dados.

Registro

Vamos fazer um **quiz**

Como você acessaria o segundo elemento de uma tupla chamada dados?

- dados[2]
- dados(1)
- dados[1]
- dados.elemento(2)

Registro

Vamos fazer um **quiz**

Como você acessaria o segundo elemento de uma tupla chamada dados?

- ✗ dados[2]
- ✗ dados(1)
- ✓ dados[1]
- ✗ dados.elemento(2)

FEEDBACK GERAL DA ATIVIDADE

Para acessar o segundo elemento de uma tupla, utilizamos a indexação correta, que em Python começa do zero.

Vamos fazer um **quiz**

O que acontece se você tentar modificar um elemento específico em uma tupla?

- A tupla é excluída.
- O elemento é removido da tupla.
- O elemento é alterado.
- Gera um erro, pois as tuplas são imutáveis.

Registro

Vamos fazer um **quiz**

O que acontece se você tentar modificar um elemento específico em uma tupla?

- ✗ A tupla é excluída.
- ✗ O elemento é removido da tupla.
- ✗ O elemento é alterado.
- ✓ Gera um erro, pois as tuplas são imutáveis.

FEEDBACK GERAL DA ATIVIDADE

Tentar modificar um elemento em uma tupla resultará em um erro, pois as tuplas são imutáveis em Python.

© Getty Images

O que nós **aprendemos hoje?**

Hoje desenvolvemos:

1. A compreensão do conceito de tuplas em Python;

2. O conhecimento da estrutura básica, das operações e das funções de tuplas;

3. Exercícios de fixação do conteúdo apresentado nesta aula.

Saiba mais

Você sabe diferenciar os conceitos de tupla e lista? Caso haja alguma dúvida, o artigo abaixo pode ajudá-lo!

ORESTES, Y. Tupla no Python: o que é, como criar e manipular e suas diferenças com as Listas. *Alura*, 2 mar. 2023. Disponível em: https://www.alura.com.br/artigos/conhecendo-as-tuplas-no-python. Acesso em: 27 mar. 2024.

Quer aprender mais sobre tupla? Acesse o curso avançado para explorar esse conceito:

ALURA. *Python Collections parte 1*: listas e tuplas. 03 Tuplas, objetos e anemia. Disponível em: https://cursos.alura.com.br/course/python-collections-listas-e-tuplas/task/52940. Acesso em: 27 mar. 2024.

Referências da aula

MENEZES, N. N. C. *Introdução à programação com Python*: algoritmos e lógica de programação para iniciantes. São Paulo: Novatec, 2019.

Educação Profissional Paulista

Técnico em Ciência de Dados

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO