**Detalhes de Tuplas**

***Mário Leite***

...

**Tupla** é uma das três estruturas básicas de dados (juntamente com *Lista* e *Dicionário*) oferecidas pelo Python; mas, diferentemente dessas outras duas, as Tuplas possuem algumas características únicas.

Elas definem um tipo de estrutura de dados que, assim como as *Listas*, podem armazenar múltiplos tipos de dados; entretanto, elas possuem algumas diferenças importantes em relação às *Listas* e aos *Dicionários* como, por exemplo:

* São imutáveis, o que significa que não podem ser alteradas após a sua criação, e nem podem ser atribuíveis: **TupLanche[4] = “Refrigerantes” => inválido**.
* São criadas e iniciadas usando parênteses, por exemplo: **minhaTupla = (1, 2, 3, 4, "Beto")**
* são geralmente usadas para armazenar dados que não serão alterados.
* São ordenadas, ou seja, os elementos mantêm a ordem na qual foram inseridos.

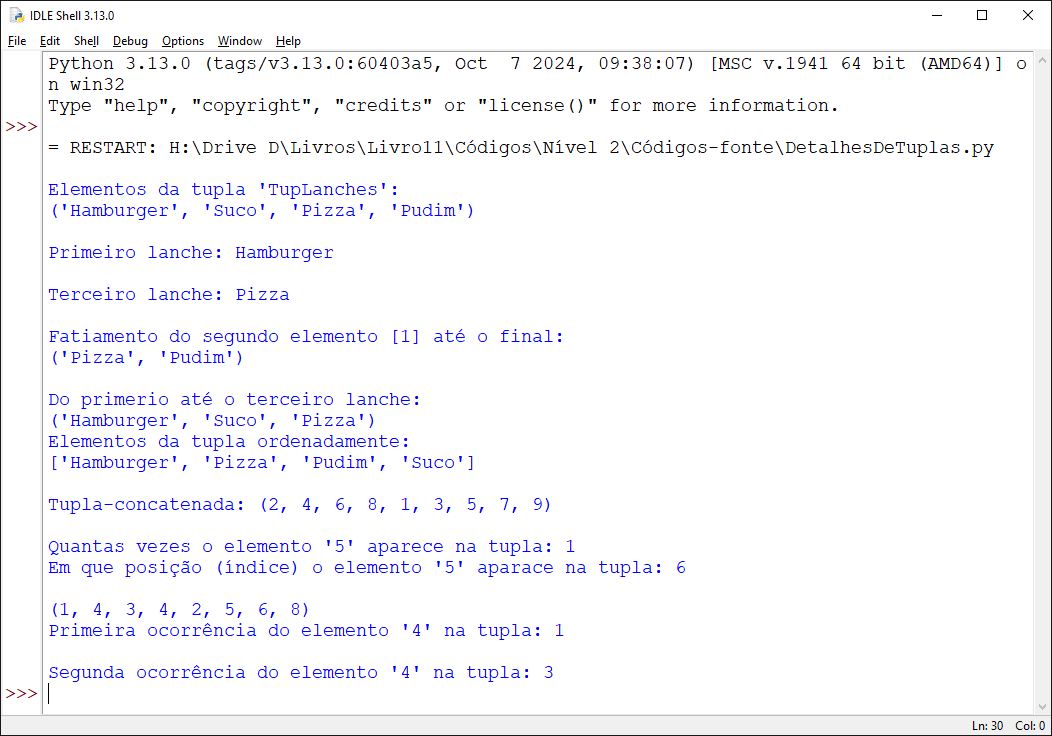
Embora as Tuplas tenham as “desvantagens” de serem imutáveis e não atribuíveis, elas oferecem algumas vantagens sobre as outras estruturas, em dois itens básicos:

* **Imutabilidade**:
* **Segurança e integridade dos dados**: Como são imutáveis, então são ideais quando é necessário garantir que os dados não serão alterados acidentalmente; isto é útil por exemplo, para representar coordenadas geográficas, configurações constantes, ou qualquer conjunto de dados que não deva ser alterado.
* **Desempenho**: Oferecem um desempenho ligeiramente melhor em termos de velocidade, pois o Python pode otimizar o uso de tuplas em operações que envolvem armazenamento e busca de dados. A imutabilidade permite que sejam otimizadas internas que não são possíveis em estruturas mutáveis.
* **Uso de memória:**
* Tuplas tendem a ser mais eficientes em termos de uso de memória quando comparadas às *Listas*; isto ocorre porque as *Listas* precisam alocar mais memória para permitir alterações, enquanto que as Tuplas podem ser armazenadas de maneira mais compacta.
* Em termos de performance de tempo para iteração ou armazenamento, as Tuplas podem ser mais rápidas para acessar ou iterar, dado que sua imutabilidade pode ser mais facilmente otimizada.

Na prática, as Tuplas são úteis quando é necessária uma estrutura de dados simples, eficiente e imutável, especialmente para representar dados constantes e em cenários que exigem segurança ou maior desempenho.

A **figura 1** mostra a saída do programa **“DetalhesDeTuplas”**, que implementa uma solução simples para mostrar alguns detalhes, funcionalidades e operações com Tuplas.

---------------------------------------------------------------------------------------------------------------------------------



**Figura 1 - Saída do programa “DetalhesDeTuplas”**

'''

**DetalhesDeTuplas.py**

-----------------------------------------------------------------------

Mostra alguns detalhes de uma Tupla.

-----------------------------------------------------------------------

'''

**print**()

TupLanches = ("Hamburger", "Suco", "Pizza", "Pudim") *#inicia uma tupla*

*#Exibe os elementos da tupla* ***TupLanches***

**print**("Elementos da tupla 'TupLanches':")

**print**(TupLanches)

**print**()

*#Exibe o primeiro lance*

**print**("Primeiro lanche: ", end="")

**print**(TupLanches[0])

**print**()

*#Exibe o terceiro lanche*

**print**("Terceiro lanche: ", end="")

**print**(TupLanches[2])

**print**()

*#Fatiamento do segundo elemento [****1****] até o final*

**print**("Fatiamento do segundo elemento [1] até o final:")

**print**(TupLanches[2:])

**print**()

*#Mostra desde o início [****0****] até o terceiro [****2****] elemento*

**print**("Do primerio até o terceiro lanche:")

**print**(TupLanches[:3])

*#Exibindo os elementos da tupla ordenadamente*

**print**("Elementos da tupla ordenadamente:")

**print**(**sorted**(TupLanches))

**print**()

*#Concatenando tuplas*

Tup1 = (2,4,6,8)

Tup2 = (1,3,5,7,9)

TupConcat = Tup1 + Tup2

**print**(f"Tupla-concatenada: **{TupConcat}**") *#ocorre uma junção das tuplas*

**print**()

*#Quantas vezes um elemento aparece dentro de uma tupla:*

**print**("Quantas vezes o elemento '5' aparece na tupla: ", end="")

**print**(TupConcat.**count**(5)) *#quantas vezes o elemento [5] aparece em* ***TupConcat***

*#Em que posição aparece determinado elemento de uma tupla*

**print**("Em que posição (índice) o elemento '5' aparace na tupla: ", end="")

**print**(TupConcat.index(5))

**print**()

*#E se um elemento ocorrer mais de uma vez numa tupla!!??*

Tup3 = (1,4,3,4,2,5,6,8)

**print**(Tup3)

**print**("Primeira ocorrência do elemento '4' na tupla: ", end="")

**print**(Tup3.index(4)) *#primeira ocorrência de* ***4****: posição* ***1***

**print**()

*'''E o outro 4, como encontrar!!??*

*#Como já se sabe que a primeira ocorrência foi na posição 1, repete-se a*

*Instrução começando na posição 2'''*

**print**("Segunda ocorrência do elemento '4' na tupla: ", end="")

**print**(Tup3.index(4,2))

**#Fim do programa "DetalhesDeTuplas" -----------------------------------**