

Aula 1 – Tuplas

1. Verificação de coordenadas de estações de metrô

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}")
```

Saída esperada:

text



```
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)
```

2. Criação e acesso a tupla

python



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

python



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

3. Tentativa de modificação de tupla (gera erro)

python



```
coordenadas = (2, 3)
coordenadas[0] = 1 # tentando substituir o elemento de índice 0 que vale 2 por 1
# Isso gera TypeError
```

4. Modificando valor de tupla via conversão para lista

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)
# Saída: (1, 10, 3, 4, 5)
```

5. Percorrendo tupla

python



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

6. Percorrendo tupla com índices

python



```
minha_tupla = (10, 20, 30, 40, 50)
for indice, elemento in enumerate(minha_tupla):
    print(f"Índice: {indice}, Elemento: {elemento}")
```

7. Tuplas aninhadas

python



```
minha_tupla = ((1,2,3), 1, [1,4,'sim'], True)
print(type(minha_tupla))
print(type(minha_tupla[0]))
print(type(minha_tupla[2]))
# Saída:
# <class 'tuple'>
# <class 'tuple'>
# <class 'list'>
```

python



```
tupla_aninhada = ((1, 2, 3), ('a', 'b', 'c'), (True, False, True))
print(tupla_aninhada[0])      # (1, 2, 3)
print(tupla_aninhada[1][1])   # 'b'
print(tupla_aninhada[2][2])   # True
```

8. Descompactando tupla

python



```
minha_tupla = (10, 20, 30)
a, b, c = minha_tupla
print("a:", a)
print("b:", b)
print("c:", c)
```

python



```
minha_outra_tupla = (1, 2.9, 'palavra', True, 'sim', 2)
var1, var2, var3, var4, _, _ = minha_outra_tupla
print(var1, var2, var3, var4)
```

9. Operações com tuplas

python



```
# Concatenar tuplas
nova_tupla = minha_tupla + (4, 5, 6)
# nova_tupla: (10, 20, 30, 4, 5, 6)

# Repetir tupla
repetida = minha_tupla * 2
# repetida: (10, 20, 30, 10, 20, 30)
```

10. Funções de tuplas

python



```
# Comprimento da tupla
tamanho = len(minha_tupla)

# Encontrar índice de um elemento
indice = minha_tupla.index('texto')

# Contar ocorrência de elemento
ocorrencias = minha_tupla.count(2)

# Converter string em tupla
string = "abc"
tupla_a_partir_string = tuple(string)
```

Aula 2 – Conjuntos (Set)

1. Criação de conjunto

python



```
meu_conjunto = {1, 2, 3, 4, 5}
```

2. Unicidade de elementos

python



```
conjunto = {1, 2, 3, 3, 4}
print(conjunto) # {1, 2, 3, 4}
```

3. Adicionar e remover elementos

python



```
conjunto = {1, 2, 3}
conjunto.add(4)    # Adiciona o elemento 4
conjunto.remove(2) # Remove o elemento 2
print(conjunto)    # {1, 3, 4}
```

4. Operações de conjuntos

python



```
conjunto1 = {1, 2, 3}
conjunto2 = {3, 4, 5}
uniao = conjunto1.union(conjunto2)    # União
intersecao = conjunto1.intersection(conjunto2) # Interseção
diferenca = conjunto1.difference(conjunto2)  # Diferença
print(uniao)    # {1, 2, 3, 4, 5}
print(intersecao) # {3}
print(diferenca)  # {1, 2}
```

Usando operadores:

python



```
conjunto1 = {1, 2, 3}
conjunto2 = {3, 4, 5}
uniao = conjunto1 | conjunto2
intersecao = conjunto1 & conjunto2
diferenca = conjunto1 - conjunto2
print(uniao)    # {1, 2, 3, 4, 5}
print(intersecao) # {3}
print(diferenca)  # {1, 2}
```

5. Algoritmo de plotagem de operações de conjuntos (imagem)

python



```
import matplotlib.pyplot as plt
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
union = set1.union(set2)
intersection = set1.intersection(set2)
difference = set1.difference(set2)
fig, ax = plt.subplots(figsize=(6, 4))
ax.set_title("Operações de Conjuntos")
ax.set_xlabel("Elementos")
ax.set_ylabel("Conjuntos")
ax.set_yticks([1, 2, 3])
```

```
ax.set_yticklabels(["Conjunto 1", "Conjunto 2", "Operações"])
ax.scatter(list(set1), [1]*len(set1), color="blue", label="Conjunto 1")
ax.scatter(list(set2), [2]*len(set2), color="red", label="Conjunto 2")
ax.scatter(list(union), [3]*len(union), color="green", label="União", s=100)
ax.scatter(list(intersection), [3]*len(intersection), color="orange",
label="Interseção")
ax.scatter(list(difference), [3]*len(difference), color="purple", label="Diferença")
ax.legend()
plt.show()
```

Este código plota visualmente os elementos dos conjuntos e suas

operações

- Criação, acesso, manipulação, concatenação, filtragem, ordenação, e ajuste de elementos em tuplas.
- Exemplos de algoritmos sugeridos:

python



```
# 1. Criar tupla de 1 a 5
numeros = (1, 2, 3, 4, 5)
print(numeros)
numeros_sem_parenteses = 1, 2, 3, 4, 5
print(numeros_sem_parenteses)

# 2. Acessar segundo elemento
cores = ('vermelho', 'verde', 'azul')
print(cores[1])

# 3. Tupla com tipos diferentes
mistura = (1, 2.5, 'texto')
print(mistura)

# 4. Filtrar frutas que começam com 'l'
frutas = ('maçã', 'banana', 'laranja', 'uva')
frutas_sel = tuple(f for f in frutas if f.startswith('l'))
print(frutas_sel)

# 5. Converter tupla para lista, adicionar, voltar para tupla
nomes = ('Ana', 'João')
lista_nomes = list(nomes)
lista_nomes.append('Maria')
nomes = tuple(lista_nomes)
print(nomes)

# 6. Média das pontuações
pontuacoes = (85, 90, 78, 92, 88)
media = sum(pontuacoes) / len(pontuacoes)
print(media)

# 7. Concatenar tuplas
tupla1 = (1, 2)
tupla2 = (3, 4)
tupla_concat = tupla1 + tupla2
print(tupla_concat)

# 8. Ordenar tupla
alunos = ('Ana', 'João', 'Maria', 'Carlos')
alunos_ord = tuple(sorted(alunos))
print(alunos_ord)

# 9. Imprimir números pares
num_tupla = tuple(range(1, 11))
pares = tuple(n for n in num_tupla if n % 2 == 0)
print(pares)

# 10. Ajustar notas
notas = (7, 8, 6, 9, 5)
notas_ajustadas = tuple(n + 1 for n in notas)
print(notas_ajustadas)
```

Aula 3 – Registro de Alunos (Tuplas e Funções)

python



```
registros_alunos = [
    ("Alice", 18, 8.5, ["Matemática", "História"]),
    ("Bob", 17, 7.2, ["Inglês", "Ciências"]),
    ("Charlie", 16, 6.8, ["Matemática", "Inglês"])
]

def imprimir_alunos_aprovados(registros):
    for nome, idade, media, disciplinas in registros:
        if media >= 7.0:
            print(nome)

def encontrar_aluno_disciplina(registros, disciplina):
    return [nome for nome, idade, media, disciplinas in registros if disciplina in
disciplinas]

# Uso das funções
imprimir_alunos_aprovados(registros_alunos)
print(encontrar_aluno_disciplina(registros_alunos, "Matemática"))
```

Aula 4 – Exercícios Práticos com Conjuntos

Exercícios propostos:

python



```
# 1. Criar conjunto de 1 a 5
numeros = {1, 2, 3, 4, 5}
print(numeros)

# 2. Verificar se 'i' está em vogais
vogais = {'a', 'e', 'i', 'o', 'u'}
print('i' in vogais)

# 3. União de dois conjuntos
conjunto1 = {1, 2, 3}
conjunto2 = {3, 4, 5}
uniao = conjunto1 | conjunto2
print(uniao)

# 4. Adicionar elemento
cores_primarias = {'vermelho', 'azul', 'amarelo'}
cores_primarias.add('verde')
print(cores_primarias)

# 5. Remover elemento
alunos_matriculados = {'Ana', 'João', 'Maria'}
alunos_matriculados.remove('João')
print(alunos_matriculados)

# 6. Dobrar elementos de um conjunto
pares = {2, 4, 6, 8, 10}
pares_multiplicados = {x * 2 for x in pares}
print(pares_multiplicados)
```



```

# 7. Interseção de conjuntos de frutas
frutas = {'maçã', 'banana', 'abacaxi'}
frutas_exoticas = {'abacaxi', 'kiwi'}
print(frutas & frutas_exoticas)

# 8. Filtrar animais domésticos
animais = {'cachorro', 'gato', 'pássaro'}
animais_domesticos = {a for a in animais if a in {'cachorro', 'gato'}}
print(animais_domesticos)

# 9. Diferença entre conjuntos
conjunto_A = {1, 2, 3}
conjunto_B = {2, 3, 4}
diferenca = conjunto_A - conjunto_B
print(diferenca)

# 10. Filtrar vogais de conjunto de letras
letras = {'a', 'b', 'c', 'd'}
letras_vogais = {l for l in letras if l in 'aeiou'}
print(letras_vogais)

```

Aula 4 – Clube de Esportes (Funções com Sets)

python



```

membros_futebol = set()
membros_basquete = set()
membros_volei = set()

def adicionar_membro(nome, esporte):
    if esporte == 'futebol':
        membros_futebol.add(nome)
    elif esporte == 'basquete':
        membros_basquete.add(nome)
    elif esporte == 'volei':
        membros_volei.add(nome)

def remover_membro(nome):
    removidos = []
    for equipe, membros in [('futebol', membros_futebol), ('basquete',
membros_basquete), ('volei', membros_volei)]:
        if nome in membros:
            membros.remove(nome)
            removidos.append(equipe)
    print(f"{nome} removido dos times: {' '.join(removidos)}")

def listar_times():
    print("Futebol:", membros_futebol)
    print("Basquete:", membros_basquete)
    print("Vôlei:", membros_volei)

def verificar_presenca(nome):
    presente = any(nome in equipe for equipe in [membros_futebol, membros_basquete,
membros_volei])
    print(f"{nome} está em algum time? {presente}")

```

Exemplos de uso:

```
adicionar_membro("Ana", "futebol")  
adicionar_membro("João", "basquete")  
listar_times()  
remover_membro("Ana")  
verificar_presenca("João")
```