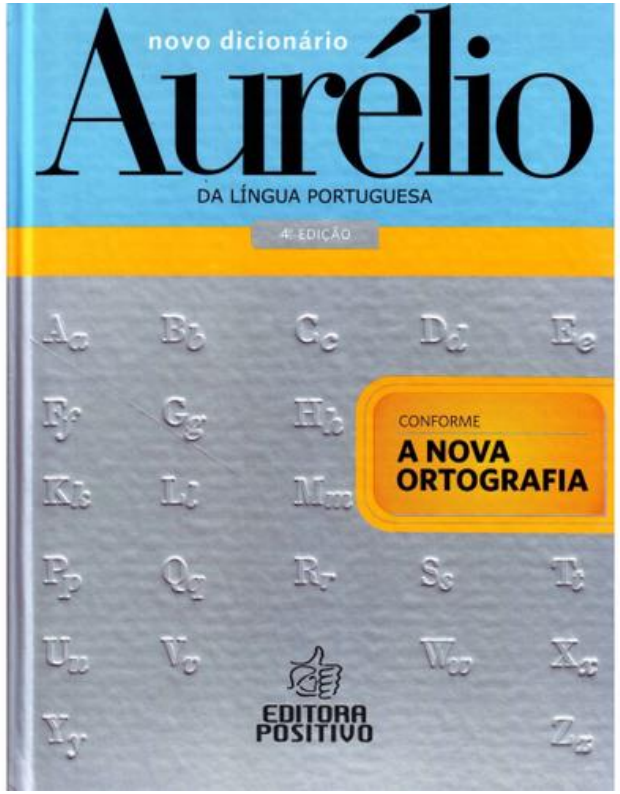
Fernanda Fretes

Lógica de Programação e Algoritmos

Lógica de Programação e Algoritmos

Fernanda Fretes

- Dicionários



Lógica de Programação e Algoritmos

• Em um dicionário, nós armazenamos um dado (valor) que queremos localizar e recuperar posteriormente através de uma chave

Assim, o principal propósito de um dicionário é associar um **valor (value) a uma chave (key)**

Lógica de Programação e Algoritmos

Nesse exemplo temos um dicionário onde a chave é a sigla de um Estado brasileiro, e o valor é o nome completo desse Estado

Key	Value
AC	Acre
AL	Alagoas
AM	Amazonas
AP	Amapá
BA	Bahia
...	...
RS	Rio Grande do Sul
SC	Santa Catarina
SE	Sergipe
SP	São Paulo
TO	Tocantins

Lógica de Programação e Algoritmos

Para fazer isso em python, podemos colocar a chave “:” e o valor dele

```
dic_estados = { "MG":"Minas Gerais",
                "PR": "Paraná",
                "BA": "Bahia",
                "RN": "Rio Grande do Norte",
                "AM": "Amzonas"}

print(dic_estados)
```

```
dic_produtos = {1215:"Lápis", 3221:"Caneta", 2329:"Borracha", 1092:"Caderno", 7633:"Cola"}
print(dic_produtos)
```

Lógica de Programação e Algoritmos

Os dicionários aceitam todos os tipos de dados como valor, inclusive listas e outros dicionários.

```
dic_notas_alunos = {"João":[30, 12, 21], "Maria": [20, 30, 29], "José": [20, 23, 19]}
print(dic_notas_alunos)
```

```
dic_notas_alunos2 = {"João": {"nota1": 30, "nota2": 12, "nota3": 21},
                     "Maria": {"nota1": 20, "nota2": 30, "nota3": 29},
                     "José": {"nota1": 20, "nota2": 23, "nota3": 19},
                    }
print(dic_notas_alunos2)
```

Lógica de Programação e Algoritmos

Para recuperar um valor armazenado no dicionário, devemos usar sua chave.

Fernanda Fretes

```
20
21  #acessando um item do dicionario
22  print(dic_estados["PR"])
23
24  print(dic_produtos[2329])
25
26  # Acessando o valor associado à chave "MG" dentro de um comando print
27  print(f"Eu nasci em {dic_estados["MG"]}.")
```

```
dicionario
Paraná
Borracha
Eu nasci em Minas Gerais.
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Para adicionar um novo valor ao dicionario, devemos colocar a chave entre [] recebendo o valor entre {}

```
dic_estados["SP"] = "São Paulo"
```

```
{'MG': 'Minas Gerais', 'PR': 'Paraná', 'BA': 'Bahia', 'RN': 'Rio Grande do Norte', 'AM': 'Amzonas', 'SP': 'São Paulo'}
```

Lógica de Programação e Algoritmos

Caso, seja necessário **modificar** o valor de uma chave, podemos fazer similar a lista, mas passando a chave do registro como o localizador do item.

```
dic_estados["AM"] = "Amazonas"
print(dic_estados)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Exibindo um item do dicionário pela sua chave.

```
print(est.get("PR"))
```

Lógica de Programação e Algoritmos

Limpar um dicionário:

Se eu quiser **limpar** todo o dicionário, basta que eu use o comando **clear**().

```
dic_notas_alunos2.clear()
print(dic_notas_alunos2)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Copiar dicionários

Para fazer a cópia desse dicionário para um outra variável, podemos usar o **copy**.

```
est = dic_estados.copy()
print(est)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Exibir as chaves

Exibindo todas as chaves de um dicionário.

```
print(est.keys())
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Apagar um registro pela chave

Para remover um item do dicionário, usamos o comando POP, e entre parênteses deixamos a chave do registro que queremos apagar.

```
est.pop("MG")
print(est)
```

E podemos usar o del também.

```
del dic_estados["RN"]
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Atualizar dicionário.

Caso, eu tenha dois dicionários e eu precise junta-los em um único dicionário, posso usar o comando **update.**

```
    est.update(dic_estados_centro_oeste)
    print(est)
```

```
dicionario
C:\Users\htafr\PycharmProjects\LOP\.venv\Scripts\python.exe C:\Users\htafr\PycharmProjects\LOP\dicionario.py
{'MG': 'Minas Gerais', 'PR': 'Paraná', 'BA': 'Bahia', 'RN': 'Rio Grande do Norte', 'AM': 'Amzonas', 'MS': 'Mato Gross

Process finished with exit code 0
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Lógica de Programação e Algoritmos

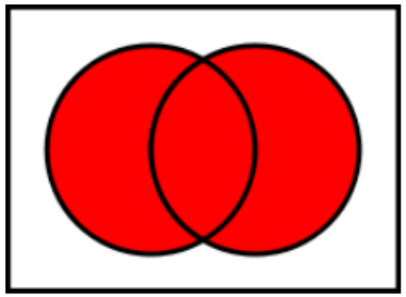
- Conjuntos são coleções de elementos únicos
- Principais características:

  - Os elementos não são armazenados em uma ordem específica
  - Conjuntos não contém elementos repetidos
  - Conjuntos não suportam indexação como as listas e tuplas

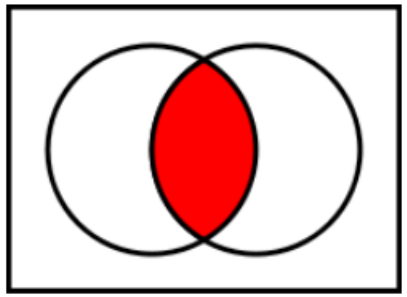
Lógica de Programação e Algoritmos

Fernanda Fretes

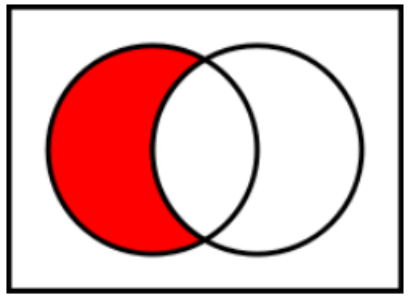
Conjuntos fornecem métodos para as operações mais conhecidas de teoria dos conjuntos, como união, interseção e diferença, além de outras.



União



Interseção



Diferença

Lógica de Programação e Algoritmos

Fernanda Fretes

Os conjuntos são caracterizados com a criação usando {}.

```
conj1 = {1, 2, 3, 4, 5}
conj2 = {"A", "B", "C", "D"}
conj3 = {"ABC", 123, 3.14}
```

Lógica de Programação e Algoritmos

Fernanda Fretes

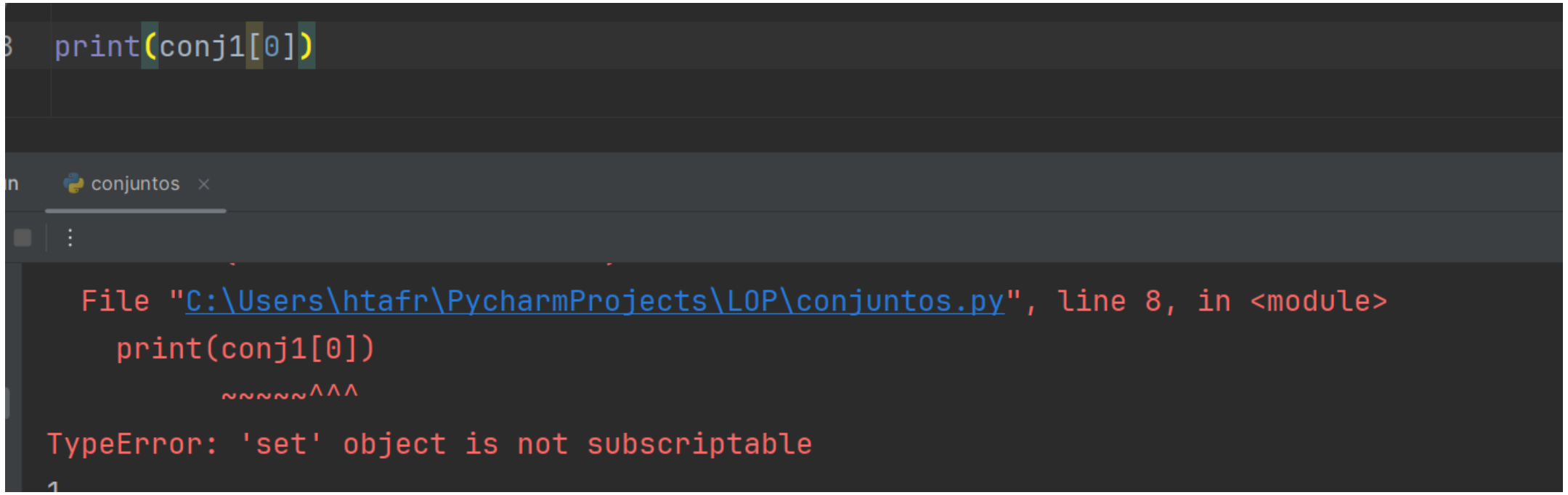
Para acessar os elementos dentro de um conjunto, podemos fazer a leitura deles usando o FOR, por exemplo.

```
for elem in conj1:
    print(elem)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Não posso fazer a pesquisa de um item pelo seu índice, por exemplo, pois, isso gera um erro.



Lógica de Programação e Algoritmos

Fernanda Fretes

Para **adicionar** um item ao conjunto, posso usar o **add**.

```
conj1.add(6)
```

```
{1, 2, 3, 4, 5, 6}
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Para saber se existe um dado no conjunto, posso executar o comando:

```
print(30 in conj)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

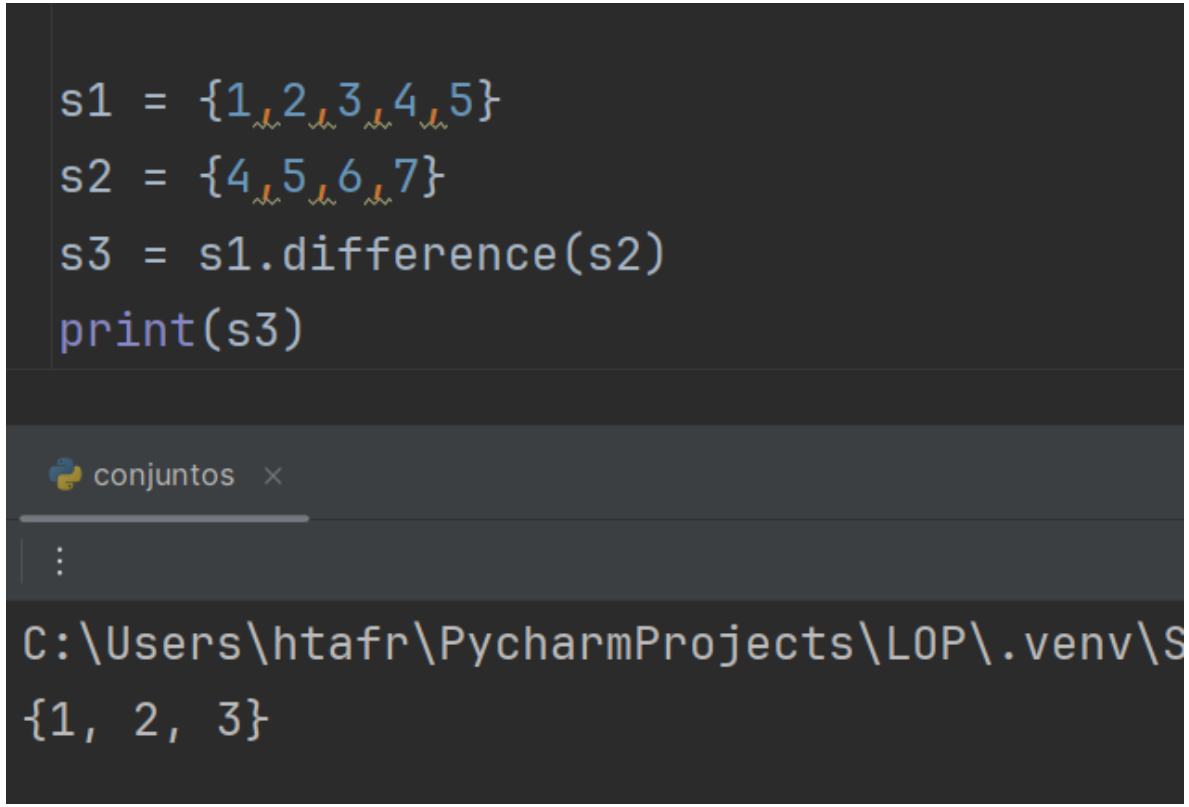
Para **limpar** os dados dentro de um conjunto, devemos usar o **clear**().

```
conj3.clear()
print(conj3)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Agora, caso eu queira saber quais os elementos que existem em um conjunto e não em outro, usamos o difference.



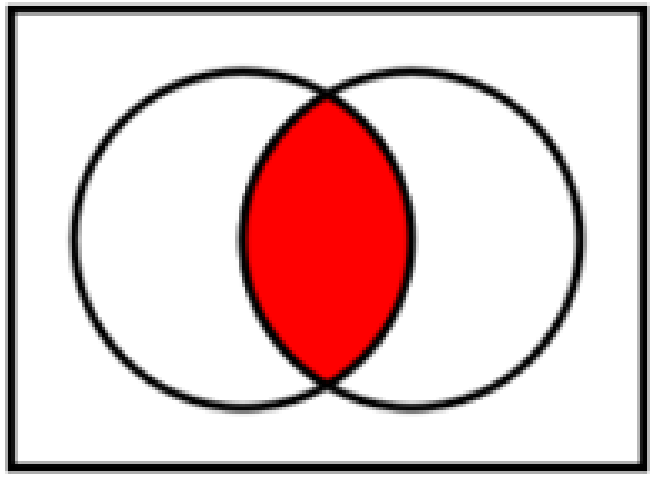
Lógica de Programação e Algoritmos

Podemos saber a interseção, através do comando

```
s1 = {1,2,3,4,5}
s2 = {4,5,6,7}
```

```
s4 = s1.intersection(s2)
print(s4)
```

```
{4, 5}
```



Interseção

Lógica de Programação e Algoritmos

Fernanda Fretes

Podemos saber se um conjunto, e um subconjunto de outro. Ou seja, se todos os elemento de um estão contidos em outro, usando o issubset.

```
s10 = {10,11,12,13,14,15}
s11 = {10,11,12}
print(s11.issubset(s10))
```

```
True
```

Lógica de Programação e Algoritmos

Podemos saber se um conjunto, contém todos os elementos de um outro, usando issuperset.

```
s10 = {10,11,12,13,14,15}
s11 = {10,11,12}
print(s10.issuperset(s11))
```

```
True
```

Lógica de Programação e Algoritmos

Fernanda Fretes

Usamos o remove, para excluir um elemento do conjunto.

```
s10.remove(15)
print(s10)
```

Lógica de Programação e Algoritmos

Usamos o union, para aglomerar os itens de 2 conjuntos.

```
s6 = s10.union(s11)
print(s6)
```

Lógica de Programação e Algoritmos

Fernanda Fretes

1. Através de função, permita que o usuário possa
    1. Crie um dicionário com três pares chave-valor representando códigos e livros.
    2. Adicione uma nova chave-valor ao dicionário existente.
    3. Acesse o valor associado a uma chave específica no dicionário.
    4. Verifique se uma chave existe no dicionário.
    5. Remova uma chave do dicionário.
    6. Itere sobre todas as chaves do dicionário e imprima-as.
    7. Itere sobre todos os valores do dicionário e imprima-os.
    8. Itere sobre todos os itens do dicionário e imprima-os.

Lógica de Programação e Algoritmos

Fernanda Fretes

1. Através de funções, permita que o usuário possa escolher entre as opções:
    1. Adicione elementos ao conjunto.
    2. Remova um elemento específico do conjunto.
    3. Verifique se um elemento está presente no conjunto.
    4. Verifique a interseção de dois conjuntos.
    5. Verifique a união de dois conjuntos.
    6. Verifique se um conjunto é um subconjunto de outro.
    7. Calcule a diferença entre dois conjuntos.
    8. Remova todos os elementos de um conjunto.

Lógica de Programação e Algoritmos

Fernanda Fretes

1.Crie um dicionário representando uma lista de contatos, onde as chaves são os nomes das pessoas e os valores são os números de telefone.
2.Adicione um novo contato ao dicionário.
3.Verifique se um determinado contato está presente na lista.
4.Remova um contato da lista pelo nome.
5.Imprima todos os nomes de contatos presentes na lista.
6.Imprima todos os números de telefone de contatos presentes na lista.
7.Conte quantos contatos existem na lista.
8.Verifique se um determinado número de telefone está presente na lista.
9.Imprima todos os pares chave-valor da lista de contatos.
10.Limpe a lista de contatos, removendo todos os contatos.