





FT 08			
Curso: UFCD 10793			
UFCD/Módulo/Temática: UFCD 10793 - Fundamentos de Python			
Ação: 10793_1/AT			
Formador/a: Sandra Liliana Meira de Oliveira			
Data:			
Nome do Formando/a:			

Objetivo: Criação e Manipulação de Sets

Set é uma coleção sem elementos duplicados. Por isso é bastante utilizada para remover elementos duplicados de listas já existentes (desde que a ordem não importe). A verificação de existência de elemento é otimizada em sets.

Sets também podem ser vistos como conjuntos matemáticos. As operações que se aplicam a conjuntos matemáticos também podem ser utilizados em sets

Operador Matemático	Operador Python	Exemplo	
∪ (União)		a b	
∩ (Interseção)	&	a	& b
Diferença	_	a - b	
Diferença Simétrica	٨	a ^ b	

Reproduz os seguintes exemplos. O objetivo é instanciar sets e aplicar as operações específicas deste tipo de dados

```
xs = {1, 2, 3}
aux = 1 in xs # True, implementação mais eficiente
print(aux)
a = set('abracadabra') # a = {'a', 'r', 'b', 'c', 'd'}
print(a)
b = set('alacazam') # b = {'c', 'l', 'm', 'z', 'a'}
print(b)
c = a - b # c = {'r', 'b', 'd'}, letras em a mas não em b
print(c)
c = a | b # c = {'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'} letras em a ou b ou ambos
print(c)
c = a & b # c = {'a', 'c'}, letras tanto em a quanto em b
print(c)
c = a ^ b # c = {'l', 'm', 'b', 'z', 'r', 'd'}, letras em a ou b, mas não em ambos
print(c)
```











```
thisset = {"apple", "banana", "cherry"}
print(thisset)
thisset = {"apple", "banana", "cherry", "apple"}
print(thisset)
#True e 1 é considerado o mesmo valor
thisset = {"apple", "banana", "cherry", True, 1, 2}
print(thisset)
#obter o comprimento do set
print(len(thisset))
#Tipos de dados string, int e booleanos:
set1 = {"apple", "banana", "cherry"}
set2 = \{1, 5, 7, 9, 3\}
set3 = {True, False, False}
#Um conjunto com strings, inteiros e valores booleanos:
set1 = {"abc", 34, True, 40, "male"}
#type()
myset = {"apple", "banana", "cherry"}
print(type(myset))
```













```
#0 construtor set()
thisset = set(("apple", "banana", "cherry")) # note the double round-brackets
print(thisset)
#métodos e operações com conjuntos
# create a set of integer type
student_id = {112, 114, 116, 118, 115}
print('Student ID:', student_id)
# create a set of string type
vowel letters = {'a', 'e', 'i', 'o', 'u'}
print('Vowel Letters:', vowel letters)
# create a set of mixed data types
mixed_set = {'Hello', 101, -2, 'Bye'}
print('Set of mixed data types:', mixed_set)
# create an empty set
empty set = set()
# create an empty dictionary
empty_dictionary = { }
# check data type of empty set
print('Data type of empty set:', type(empty set))
numbers = \{2, 4, 6, 6, 2, 8\}
print(numbers) # {8, 2, 4, 6}
```

```
#adicionar e atualizar elementos

numbers = {21, 34, 54, 12}

print('Initial Set:',numbers)

# using add() method
#Conjuntos são mutáveis. No entanto, como não estão ordenados, a indexação não tem significado.
#Não podemos aceder ou alterar um elemento de um conjunto usando indexação ou divisão. O tipo de dados definido não é compatível.

numbers.add(32)

print('Updated Set:', numbers)
```











```
#método update()
companies = {'Lacoste', 'Ralph Lauren'}
tech_companies = ['apple', 'google', 'apple']
companies.update(tech companies)
print(companies)
#Remover um elemento de um conjunto
#Usamos o discard()método para remover o elemento especificado de um conjunto. Por exemplo,
languages = {'Swift', 'Java', 'Python'}
print('Initial Set:',languages)
removedValue = languages.discard('Java')
print('Set after remove():', languages)
fruits = {"Apple", "Peach", "Mango"}
# for loop to access each fruits
for fruit in fruits:
   print(fruit)
even_numbers = \{2,4,6,8\}
print('Set:',even_numbers)
print('Total Elements:', len(even_numbers))
```

```
# find number of elements
print('Total Elements:', len(even_numbers))

#uniao de conjuntos

# first set
A = {1, 3, 5}

# second set
B = {0, 2, 4}

# perform union operation using |
print('Union using |:', A | B)

# perform union operation using union()
print('Union using union():', A.union(B))
```











```
#interseção de conjuntos
# first set
A = {1, 3, 5}

# second set
B = {1, 2, 3}

# perform intersection operation using &
print('Intersection using &:', A & B)

# perform intersection operation using intersection()
print('Intersection using intersection():', A.intersection(B))

#diferença de dois conjuntos - A sem B

# first set
A = {2, 3, 5}

# second set
B = {1, 2, 6}

# perform difference operation using &
print('Difference using &:', A - B)

# perform difference operation using difference()
print('Difference using difference():', A.difference(B))
```

```
#Diferença simétrica de dois conjuntos - A sem os elementos comuns a B
# first set
A = {2, 3, 5}
# second set
B = {1, 2, 6}
# perform difference operation using &
print('using ^:', A ^ B)
# using symmetric_difference()
print('using symmetric_difference():', A.symmetric_difference(B))
#verificar se dois conjuntos são iguais
# first set
A = {1, 3, 5}
# second set
B = {3, 5, 1}
# perform difference operation using &
if A == B:
    print('Set A and Set B are equal')
else:
    print('Set A and Set B are not equal')
```





