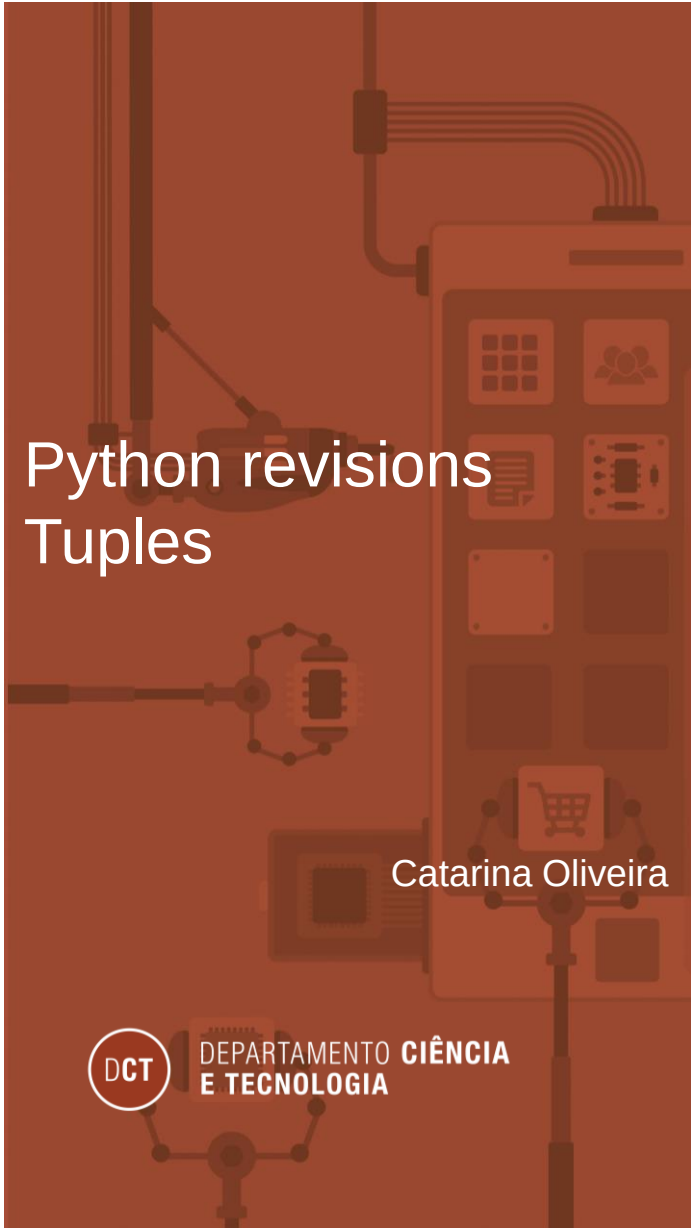

UNIVERSIDADE PORTUCALENSE

CONTENT

1. Conditions
2. Cycles
3. lists
4. input and output
5. Functions
6. tuples
7. Combination of lists and tuples

Conditions

```
if x < y:
    print("menor")
elif x > y:
    print("maior")
else:
    print("igual")
```

Cycles

```
n = 6
current_sum = 0
i = 0
while i<= n:
    current_sum += i
    i += 1
print(current_sum)
```

```
for x in range(5): # 0, 1, 2, 3, 4
    print(x)

for x in range(3,10): # 3, 4, 5, 6, 7, 8, 9
    print(x)

for x in range(3,10,2): #3, 5, 7, 9
    print(x)
```

lists

Example

considering the list

```
xs = [12, 10, 32, 3, 66, 17, 42, 99, 20]
```

Determine the mean, largest, and smallest element

```
soma = 0
maior = 0
menor = 99999
for x in xs:
    soma += x
    if x > maior:
        maior = x
    if x < menor:
        menor = x
print("Média:", soma / len(xs))
print("Maior:", maior)
print("Menor:", menor)
```

Input and Output

```
nome = input("Introduzir o nome: ")
idade = int(input("Introduzir a idade: "))
print("O nome é {0} e a idade {1}".format(nome, str(idade)))
```

Functions

```
def soma(a, b):
    resultado = a + b
    return resultado
```

```
def soma(a, b):
    return a + b
```

```
soma(3,5)
```

tuples

- Ordered sequences of elements: (e1, e2, ..., en)
- Access to elements by indexes
- Immutable
- Operations with tuples :

```
xs = ("Pedro", 12)
print(len(xs))           # [Tamanho]         2
print(xs + ("João", 14)) # [Concatenação]    ('Pedro', 12, 'João', 14)
print(2 * xs)            # Repetição         ('Pedro', 12, 'Pedro', 12)
print(12 in xs)          # Pertença          True
print(xs[0])             # "Pedro"
xs[1] = 14               # TypeError: 'tuple' object does not support item assignment
l=list(xs)               # converte tuplo para lista
xs2 = tuple(l)           # converte lista para tuplo
for x in xs:             # Iteração
    print(x)
```

Combination of lists and tuples

- represent an agenda

```
minhaAgenda = [("Maria João", "mj@mail.pt"),
               ("José Manuel", "jm@mail.pt"),
               ("João Pedro", "jp@mail.pt")]
```

- Add entries (name and email) to the calendar

```
def adicionar(agenda, nome, email):
    agenda.append((nome, email))


adicionar(minhaAgenda, "José Carlos", "jc@mail.pt")
print(minhaAgenda)
# [('Maria João', 'mj@mail.pt'),
#  ('José Manuel', 'jm@mail.pt'),
#  ('João Pedro', 'jp@mail.pt'),
#  ('José Carlos', 'jc@mail.pt')]
```

- Search for names in the agenda and return the email

```
def procurar(agenda, texto):
    emails = []
    for (nome, email) in agenda:
        if texto in nome:
            print(nome)
            emails.append(email)
    return emails

print(procurar(minhaAgenda, "João"))
# ['mj@mail.pt', 'jp@mail.pt']
```

UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.