Pensamento Computacional

Aula 7

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Objetivos

- Iteração
  - A declaração **`while`**
  - O commando **`break`**
  - A declaração **`for`**
  - A função **`range`**

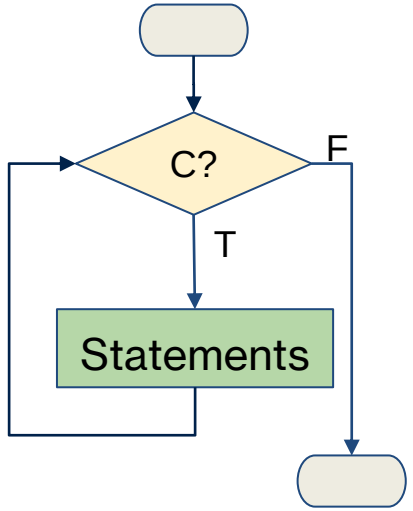
A declaração `while`

(cfr. enquanto...faça...)

- A declaração **`while`** dá indicações ao Python para **executar repetidamente** algum conjunto de declarações **enquanto uma dada condição for verdadeira**.

Sintaxe	Exemplo

`...`
`while condition:`
`    statements`
`...`
	`n = 3`
	`while n > 0:`
	`    print(n)`
	`    n = n-1`
	`print("Go!")`



A declaração `while`
(cfr. enquanto...faça...)

1. **Se a condição for verdadeira**, o bloco de commandos é executado.
2. Posteriormente, a condição é reavaliada e, **se ainda for verdadeira, o bloco de commandos é repetido**.
3. Quando a condição se torna **falsa, a execução avança** até à linha **imediatamente após** o bloco de comandos indentados.

- A condição deve ser uma expressão Booleana.
- Outros tipos de expressões são implicitamente convertidos para `bool`, desta forma, qualquer valor `null` ou vazio significará falso.

O comando `break`
(cfr. interrompa)

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

- O **corpo do ciclo** deve alterar o valor de uma ou mais variáveis de forma a que, eventualmente, a condição se torne falsa e o ciclo termine.

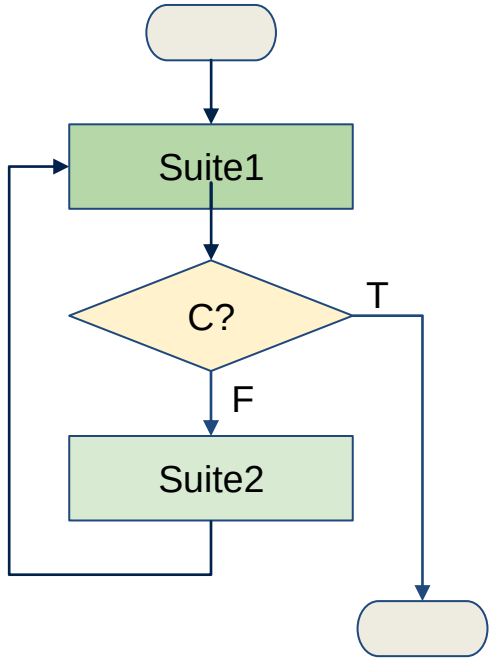
- Caso contrário, o ciclo irá repetir indefinidamente: *ciclo infinito*.

O comando `break`
(cfr. interrompa)

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

- Frequentemente, é necessário decidir se o ciclo deve terminar algures a meio. Nesse caso, usamos o commando **`break`** para saltar o ciclo.

```
while True:
    line = input('Enter text? ')
    if line == 'done':
        break
    print(line)
print('The end')
```



Ciclos com múltiplas saídas

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

- Por vezes, há várias condições que levam ao fim do ciclo e múltiplas localizações para as testar no corpo do ciclo.
- Para tal, usamos múltiplas declarações `if-break`.

```
while C1:
    Suite1
    if C2: break
    Suite2
    if C3: break
    Suite3
...
```

A declaração `for` (cfr. para...faça...)

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

- Outro mecanismo de execução repetida é a declaração **`for`**.
- Esta repete um conjunto de comandos uma vez para cada item numa *coleção* de items, tal como uma lista, uma string ou um tuplo.

Sintaxe	Exemplo

`...`
`for var in collection:`
`    statements`
`...`
	`for n in [3, 1, 9]:`
	`    print(n)`
	`print("End")`

Expressão. Primeira coisa a ser avaliada.

Depois, o 1º item da coleção é atribuído à variável e o bloco de comandos é executado.

Posteriormente, o segundo item da coleção é atribuido a `var`, as declarações são novamente executadas, e por aí adiante até que a coleção tenha sido totalmente percorrida.

A função `range`

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

- A função integrada `range` gera uma sequência de números em progressão aritmética.

```
list(range(4))  →  [0, 1, 2, 3]
```

- É frequentemente utilizada em ciclos **`for`**. Pode ser chamada com 1, 2 ou 3 argumentos:
  - `range(stop)`
  - `range(start, stop)`
  - `range(start, stop, step)`

```
for n in range(0, 4):
    print(n)
```

- Todos os argumentos devem ser **inteiros, positivos ou negativos**.
- Gera inteiro desde/até ao `stop`, **excluindo-o!**