

ALGORITMIA E ESTRUTURAS DE DADOS

MÓDULO II – ESTRUTURAS DE CONTROLO

TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB

1. Estruturas de Decisão

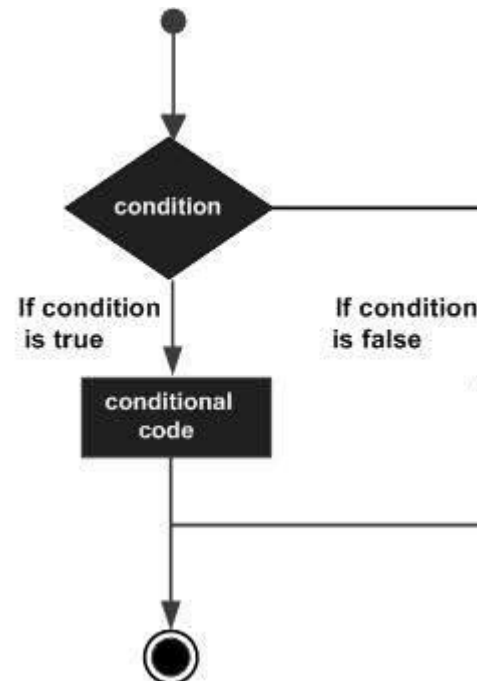
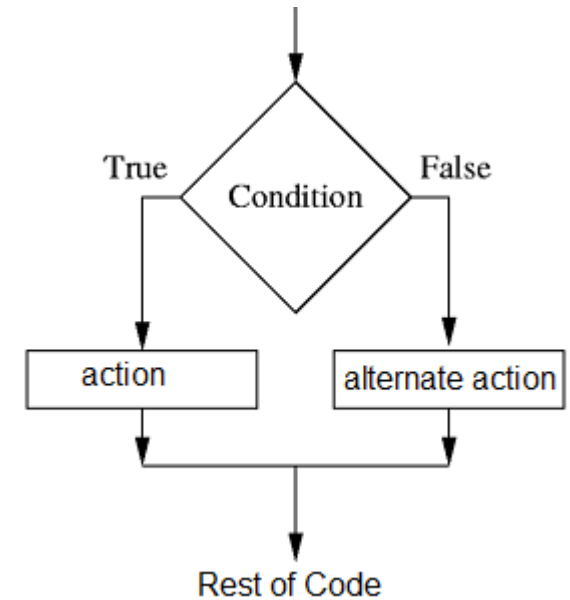
- if
- if – else
- if – elif

2. Estruturas de Repetição

- for
- while

3. Quebras de ciclo

- break
- continue



❖ Estruturas de Decisão

if

Avaliam condições e permitem executar ações (uma ou várias instruções) mediante o resultado da avaliação da condição

Especifica ação a ser executada quando se verifica determinada condição (*if*)

```
FICHA 02 > Exemplo.py > ...  
1  
2 num1 = int(input("Primeiro Número:"))  
3 num2 = int(input("Segundo Numero:"))  
4  
5 # if  
6 if num1 > num2:  
7     print("o maior é {0}" .format(num1))  
8  
9  
10
```

C:\WINDOWS\py.exe
Primeiro Número:10
Segundo Numero:7
o maior é 10
-

❖ Estruturas de Decisão

if - else

Avaliam condições e permitem executar ações (uma ou várias instruções) mediante o resultado da avaliação da condição

Especifica ação a executar quando a condição é verdadeira (*if*) e ação a executar quando a condição falha (*else*)

FICHA 02 > Exemplo.py > ...

```
1
2  num1 = int(input("Primeiro Número:"))
3  num2 = int(input("Segundo Numero:"))
4
5
6
7  if num1 > num2:
8      print("o maior é {0}" .format(num1))
9  else:
10     print("o maior é {0}" .format(num2))
11
12
```

C:\WINDOWS\py.exe

```
Primeiro Número:10
Segundo Numero:20
o maior é 20
```

❖ Estruturas de Decisão

if – elif - else

Avaliam condições e permitem executar ações (uma ou várias instruções) mediante o resultado da avaliação da condição

Especifica ação a executar quando a condição é verdadeira (*if*)
define nova condição quando a anterior falha (*elif*)

```
1
2 num1 = int(input("Primeiro Número:"))
3 num2 = int(input("Segundo Numero:"))
4
5
6 if num1 > num2:
7     print("o maior é {0}" .format(num1))
8 elif num1 == num2:
9     print("os números {0} são iguais" .format(num1))
10 else:
11     print("o maior é {0}" .format(num2))
12
13
```

C:\WINDOWS\py.exe

Primeiro Número:10
Segundo Numero:10
os números 10 são iguais

❖ Estruturas de Decisão

if – elif - else

Avaliam condições e permitem executar ações (uma ou várias instruções) mediante o resultado da avaliação da condição

Especifica ação a executar quando a condição é verdadeira (*if*)
define nova condição quando a anterior falha (*elif*)

```
1
2 num1 = int(input("Primeiro Número:"))
3 num2 = int(input("Segundo Numero:"))
4
5
6 if num1 > num2:
7     print("o maior é {0}" .format(num1))
8 elif num1 == num2:
9     print("os números {0} são iguais" .format(num1))
10 else:
11     print("o maior é {0}" .format(num2))
12
13
```

C:\WINDOWS\py.exe
Primeiro Número:15
Segundo Numero:25
o maior é 25


❖ Estruturas de Decisão

if – elif - else

Avaliam condições e permitem executar ações (uma ou várias instruções) mediante o resultado da avaliação da condição

Especifica ação a executar quando a condição é verdadeira (*if*)
define nova condição quando a anterior falha (*elif*)

```
num1 = int(input("Primeiro Número:"))  
num2 = int(input("Segundo Numero:"))  
num3 = int(input("Terceiro Numero:"))  
  
if num1 > num2 and num1 > num3:  
    print("o maior é {0}" .format(num1))  
elif num2 > num1 and num2 > num3:  
    print("o maior é {0}" .format(num2))  
else:  
    print("o maior é {0}" .format(num3))
```

 C:\WINDOWS\py.exe

```
Primeiro Número:10  
Segundo Numero:29  
Terceiro Numero:25  
o maior é 29
```

❖ Estruturas de Decisão

if – elif - else

Escrita sintética de estruturas condicionais

```
FICHA 02 > Exemplo.py > ...  
1  
2 num1 = int(input("Primeiro Número:"))  
3 num2 = int(input("Segundo Numero:"))  
4  
5 if num1 > num2:  
6     print ("o maior é {0}" .format(num1))  
7  
8 if num1 > num2: print ("o maior é {0}" .format(num1))  
9  
10
```


❖ Estruturas de Decisão

if – elif - else

Escrita sintética de estruturas condicionais

```
num1 = int(input("Primeiro Número:"))  
num2 = int(input("Segundo Numero:"))  
  
if num1 > num2:  
|   print ("o maior é {0}" .format(num1))  
else:  
|   print ("o maior é {0}" .format(num2))
```

```
print ("o maior é {0}" .format(num1)) if num1 > num2 else print ("o maior é {0}" .format(num2))
```

❖ Estruturas de Decisão

if – elif - else

Escrita sintética de estruturas condicionais

```
num1 = int(input("Primeiro Número:"))
num2 = int(input("Segundo Numero:"))

if num1 > num2:
    print("o maior é {0}" .format(num1))
elif num1 == num2:
    print("os números {0} são iguais" .format(num1))
else:
    print("o maior é {0}" .format(num2))

print ("o maior é {0}" .format(num1)) if num1 > num2 else print("os números {0} são iguais" .format(num1)) if num1 == num2 else print
["o maior é {0}" .format(num2)]
```

input()

C:\WINDOWS\py.exe

Primeiro Número:10
Segundo Numero:25
o maior é 25

❖ Estruturas de Repetição | Iterativas

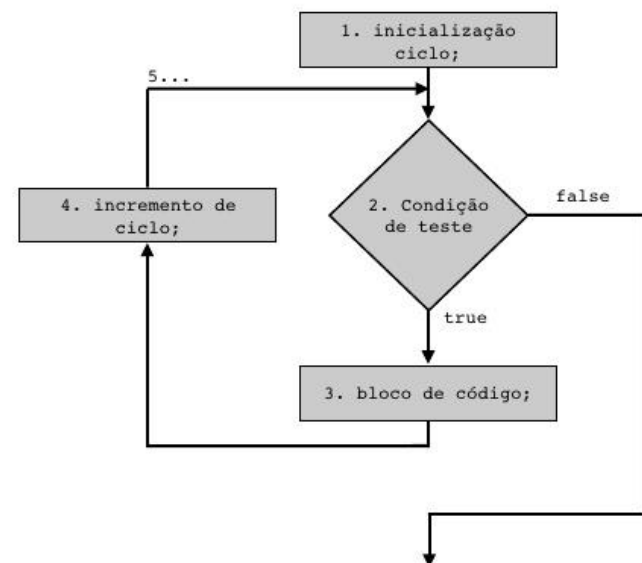
for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

A iteração (repetição) de um ciclo *for* pode ser associada a:

- ❑ Valores numéricos (*range*)
- ❑ Texto (strings)
- ❑ Sequências encadeadas
executar a repetição para
cada valor de um sequência:
array, lista, tupla, etc.

for: como funciona em fluxograma



❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

range: permite especificar o nº de vezes que o ciclo se repete

Repete o ciclo **10 vezes**. Começa com i a **0** e termina em **9**

Variável
contadora
do ciclo

```
Exemplos.py > ...  
1  # Exemplos de estruturas repetitivas (iterativas) for  
2  
3  for i in range (10):  
4      print(i)  
5  
6  
7  input()  
8  
9
```

C:\WINDOWS\py.exe

0
1
2
3
4
5
6
7
8
9
—

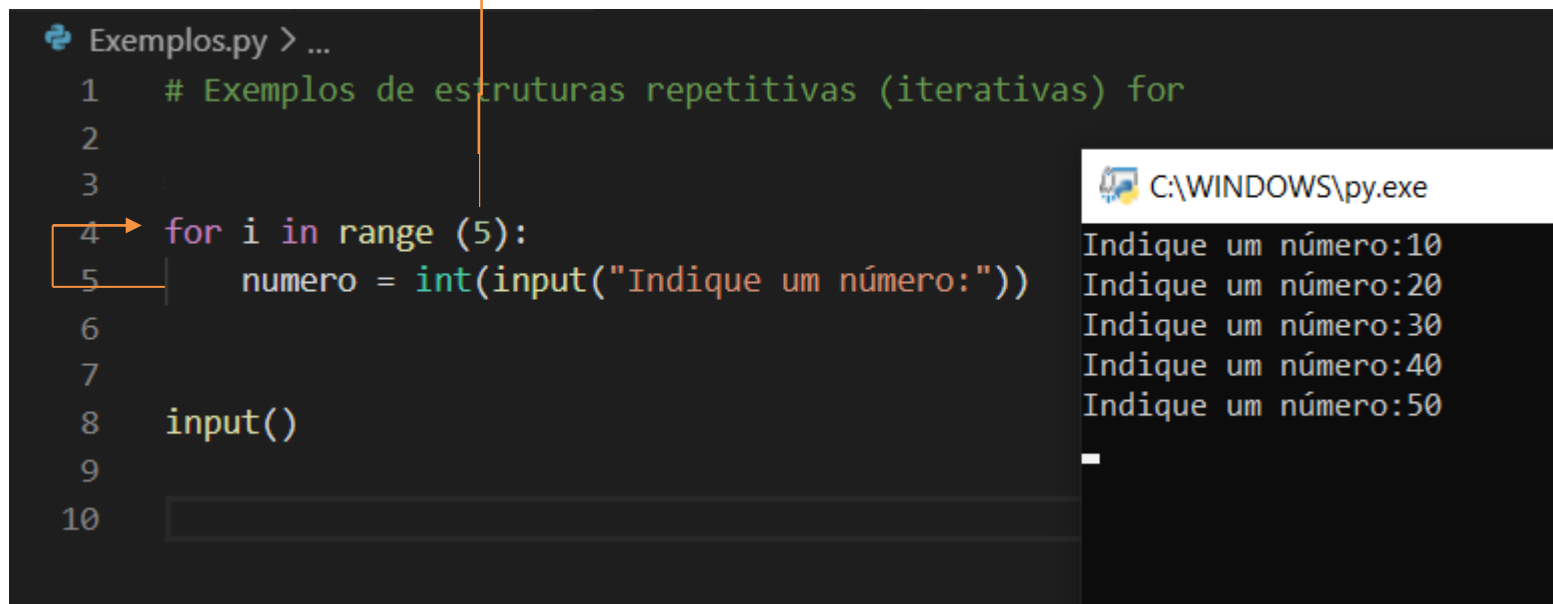
❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

range: permite especificar o nº de vezes que o ciclo se repete

Repete o ciclo **5 vezes**



The screenshot shows a Python IDE with a file named 'Exemplos.py'. The code is as follows:

```
1 # Exemplos de estruturas repetitivas (iterativas) for
2
3
4 for i in range (5):
5     numero = int(input("Indique um número:"))
6
7
8 input()
9
10
```

An orange arrow points from the text 'Repete o ciclo 5 vezes' to the '5' in the range function of the code. Another orange arrow points from the same text to the 'for' loop structure.

On the right side, a terminal window titled 'C:\WINDOWS\py.exe' shows the output of the program:

```
Indique um número:10
Indique um número:20
Indique um número:30
Indique um número:40
Indique um número:50
_
```

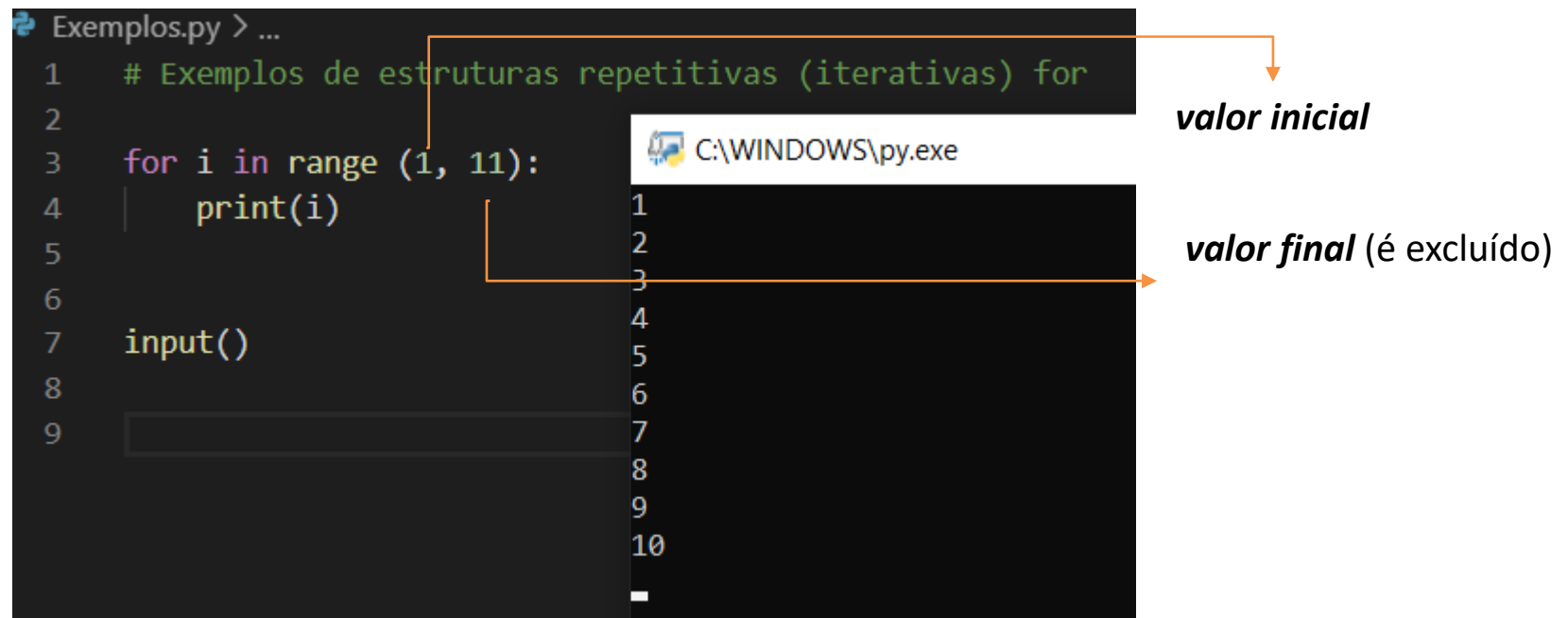
❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

range: permite especificar o nº de vezes que o ciclo se repete

Repete o ciclo **10 vezes**. Começa com **i** a **1** e termina em **10**



The image shows a screenshot of a Python script named 'Exemplos.py' being executed. The script contains the following code:

```
1 # Exemplos de estruturas repetitivas (iterativas) for
2
3 for i in range (1, 11):
4     print(i)
5
6
7 input()
8
9
```

The output of the script is a list of numbers from 1 to 10, printed one per line. An orange arrow points from the text 'valor inicial' to the number 1 in the range function of the code. Another orange arrow points from the text 'valor final (é excluído)' to the number 11 in the range function. A third orange arrow points from the text 'Repete o ciclo 10 vezes' to the output line showing the number 10.

valor inicial

valor final (é excluído)

❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

range: permite especificar o nº de vezes que o ciclo se repete

Valor inicial Valor final (excluído) Step / incremento da variável i em cada iteração

```
Exemplos.py > ...  
1 # Exemplos de estruturas repetitivas  
2  
3 for i in range (1, 20, 2):  
4     print(i)  
5  
6  
7 input()  
8  
9
```

C:\WINDOWS\py.exe
1
3
5
7
9
11
13
15
17
19

❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

range: permite especificar o nº de vezes que o ciclo se repete

Valor inicial Valor final (excluído) Step / incremento da variável i em cada iteração

```
Exemplos.py > ...  
1  # Exemplos de estruturas repeti  
2  
3  for i in range(0, 20, 2):  
4      print(i)  
5  
6  
7  input()  
8  
9
```

C:\WINDOWS\py.exe
0
2
4
6
8
10
12
14
16
18

❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

strings: podemos iterar strings, pois consistem em sequências de caracteres

```
Exemplos.py > ...
1  # Exemplos de estruturas repetit
2
3  nome = "exemplo for"
4  for caracter in nome:
5      print(caracter)
6
7
8  input()
9
10
```

C:\WINDOWS\py.exe

e
x
e
m
p
l
o

f
o
r

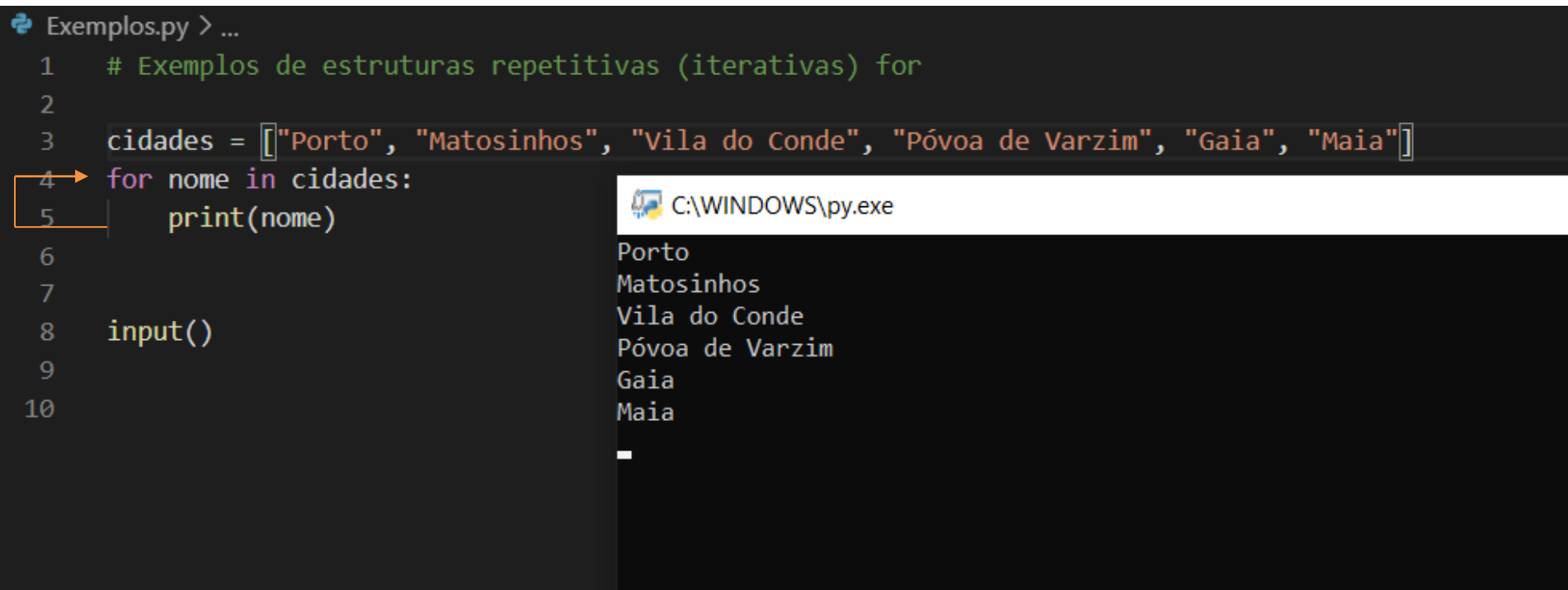
❖ Estruturas de Repetição | Iterativas

for

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

Sequências: podemos iterar estruturas que representam sequências de dados

```
Exemplos.py > ...  
1  # Exemplos de estruturas repetitivas (iterativas) for  
2  
3  cidades = ["Porto", "Matosinhos", "Vila do Conde", "Póvoa de Varzim", "Gaia", "Maia"]  
4  for nome in cidades:  
5      print(nome)  
6  
7  
8  input()  
9  
10
```



The screenshot shows a Python script being executed. The script defines a list of cities and iterates over it using a for loop, printing each city name. The output of the script is displayed in a separate window, showing the names of the cities: Porto, Matosinhos, Vila do Conde, Póvoa de Varzim, Gaia, and Maia. The script also includes an input() function at the end, which is currently not being executed.

Nota: voltaremos a estes casos mais tarde, quando usarmos estruturas de dados deste tipo

❖ Estruturas de Repetição | Iterativas

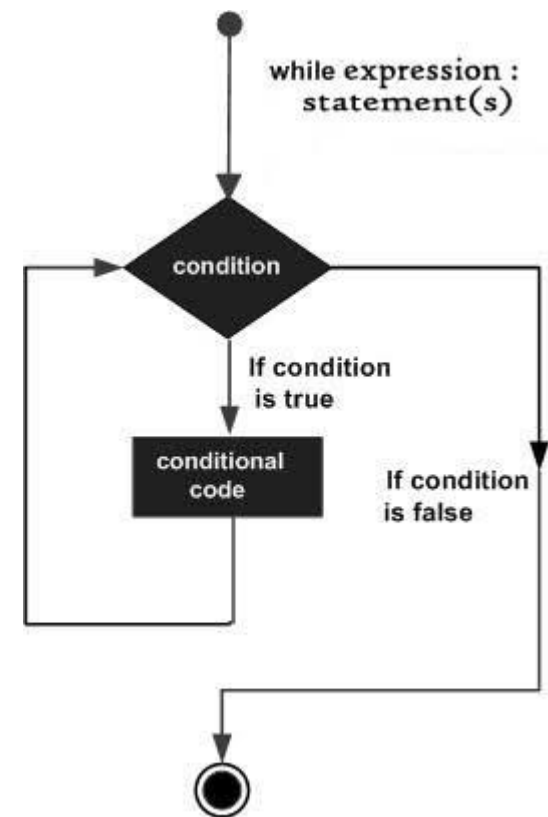
while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

Quando a condição para a ser falsa, a execução segue para a linha de código imediatamente a seguir ao fim do ciclo while

```
while expression:  
    statement(s)
```



❖ Estruturas de Repetição | Iterativas

while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

testa
novamente
a condição

```
Exemplos.py > ...  
1  # Exemplos de estruturas repetitivas (iterativas) while  
2  
3  numero = 1  
4  while numero <10 :  
5      print(numero)  
6      numero+=1  
7  
8  
9  
10 input()  
11  
12
```

C:\WINDOWS\py.exe

1
2
3
4
5
6
7
8
9

❖ Estruturas de Repetição | Iterativas

while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

```
Exemplos.py > ...  
1  # Exemplos de estruturas repetitivas (iterativas) while  
2  
3  
4  numero = int(input("Indique uma valor entre 0 e 20:"))  
5  → while numero <0 or numero >20:  
6      print("o número inserido não é válido! \n")  
7      numero = int(input("Indique uma valor entre 0 e 20:"))  
8  
9  
10 input()  
11  
12
```

C:\WINDOWS\py.exe

Indique uma valor entre 0 e 20:30
o número inserido não é válido!

Indique uma valor entre 0 e 20:-1
o número inserido não é válido!

Indique uma valor entre 0 e 20:18

❖ Estruturas de Repetição | Iterativas

while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

```
Exemplos.py > ...
1  # Exemplos de estruturas repetitivas (iterativas) while
2
3
4  tabuada = int(input("Imprimir a tabuada dos: "))
5  numero = 1
6  while numero < 11 :
7      print(tabuada, "*", numero, "=", tabuada * numero )
8      numero+=1
9
10
11  input()
12
13
```

C:\WINDOWS\py.exe

Imprimir a tabuada dos: 7

7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70

❖ Estruturas de Repetição | Iterativas

while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de entrar no ciclo

```
Exemplos.py > ...  
1  # Exemplos de estruturas repetitivas (iterativas) while  
2  import random  
3  
4  numero = random.randint(0,10)  
5  palpite = int(input("Indique o seu palpite:"))  
6  while numero != palpite:  
7      print("Não acertou, tente novamente :( \n")  
8      palpite = int(input("Indique o seu palpite:"))  
9  
10 print("Parabéns, Acertou!!! :-)" )  
11  
12  
13  
14 input()  
15  
16
```

C:\WINDOWS\py.exe

Indique o seu palpite:5
Não acertou, tente novamente :(

Indique o seu palpite:6
Não acertou, tente novamente :(

Indique o seu palpite:8
Não acertou, tente novamente :(

Indique o seu palpite:7
Não acertou, tente novamente :(

Indique o seu palpite:4
Parabéns, Acertou!!! :-)

❖ Estruturas de Repetição | Iterativas

while

Permite implementar um ciclo para executar um conjunto de instruções repetidamente, mas enquanto uma determinada condição for verdadeira

A condição é testada, repetidamente, antes de entrar no ciclo

Exemplos.py > ...

```
1  # Exemplos de estruturas repetitivas (iterativas) while
2  import random
3
4  numero = random.randint(0,10) # gera numero aleatorio ente 0 e 10
5  palpite = int(input("Indique o seu palpite: "))
6  tentativas = 1                # para contar o nº de tentativas até acertar
7
8  while numero != palpite:
9      print("Não acertou, tente novamente :( \n")
10     palpite = int(input("Indique o seu palpite: "))
11     tentativas+=1
12
13     print("Parabéns, Acertou em {0} tentativas! :-)" .format(tentativas) )
14
15
16
17     input()
18
19
```

C:\WINDOWS\py.exe

Indique o seu palpite: 5
Não acertou, tente novamente :(

Indique o seu palpite: 7
Não acertou, tente novamente :(

Indique o seu palpite: 3
Não acertou, tente novamente :(

Indique o seu palpite: 8
Parabéns, Acertou em 4 tentativas! :-)

❖ Estruturas de Repetição | Iterativas

Quebras de ciclo

Permitem interromper / quebrar ciclos repetitivos

❑ break

Permite quebrar o ciclo e transfere a execução para a primeira instrução imediatamente a seguir ao ciclo

❑ continue

Permite continuar diretamente para a próxima iteração de um ciclo sem executar as instruções seguintes da iteração corrente

❖ Estruturas de Repetição | Iterativas

Quebras de ciclo

Permitem interromper / quebrar ciclos repetitivos

break

Exemplo1.py > ...

```
1  # Exemplos de quebras de ciclos
2
3  tabuada = int(input("Imprimir a tabuada dos: "))
4  numero = 0
5  while numero < 10 :
6      numero+=1
7      if numero == 6:
8          break
9      print(tabuada, "*", numero, "=", tabuada * numero)
10 input()
11
12
```

C:\WINDOWS\py.exe

```
Imprimir a tabuada dos: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
```

❖ Estruturas de Repetição | Iterativas

Quebras de ciclo

Permitem interromper / quebrar ciclos repetitivos

continue

Exemplo1.py > ...

```
1  # Exemplos de quebras de ciclos
2
3  tabuada = int(input("Imprimir a tabuada dos: "))
4  numero = 0
5  while numero < 10 :
6      numero+=1
7      if numero == 6:
8          continue
9      print(tabuada, "*", numero, "=", tabuada * numero)
10 input()
11
12
```

C:\WINDOWS\py.exe

Imprimir a tabuada dos: 7

```
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
```