



#### Unidade 2: Iniciação à programação em Python

#### Lição 2: O ciclo FOR

Nesta segunda lição da Unidade 2 vamos descobrir como repetir um procedimento ou um conjunto de instruções utilizando um ciclo FOR.

#### Objetivos:

- Explorar e implementar o ciclo FOR
- Utilizar o ciclo FOR em exemplos simples

Por vezes, num programa, é útil e/ou necessário repetir uma ou várias instruções um certo número finito de vezes. Se o número de repetições do processo é conhecido, então utilizamos um ciclo limitado **FOR**.

A sintaxe de um ciclo **FOR** é a seguinte:

#### ALGORITMO

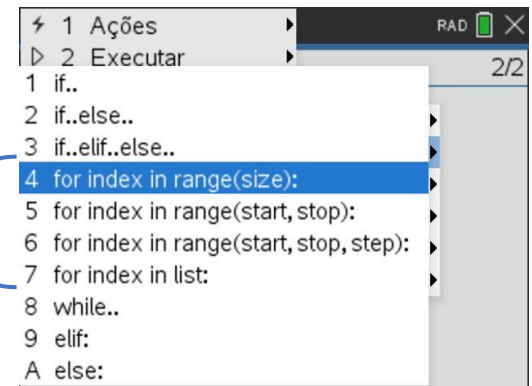
Para a variável entre o valor mínimo e o valor máximo:

Instrução1  
Instrução2  
...

#### LINGUAGEM PYTHON

```
for variável in range( )
```

```
Instrução1  
Instrução2  
...
```



A função `range()` permite, através dos seus argumentos, definir enumeradas formas de se implementar o ciclo **FOR**. Conforme imagem acima, este ciclo pode ser executado das seguintes formas:

- `for i in range(n)`: a variável `i` toma os valores inteiros de 0 até `n-1`, isto é, toma `n` valores.
- `for i in range(início, fim)`: a variável `i` toma os valores inteiros de `início` até `fim-1`, onde `início` e `fim` são os argumentos da função.
- `for i in range(início, fim, passo)`: a variável `i` toma os valores inteiros desde `início` até `fim-1`, obtidos por um incremento de `passo`, onde `início`, `fim` e `passo` são os argumentos da função.
- `for i in lista`: a variável `i` toma os valores da lista, começando pelo primeiro até ao último.

Não existe nenhuma instrução para indicar o fim de ciclo. Faz-se pela indentação, ou seja, o deslocamento para a direita de uma ou mais linhas é que assinala o fim do ciclo.

#### EXEMPLO:

Ao estudar seqüências e regularidades numéricas, foi colocado a um aluno do ensino básico o seguinte desafio:

Elabora um programa que crie uma lista com cinco primeiros quadrados perfeitos não negativos.


#### ALGORITMO

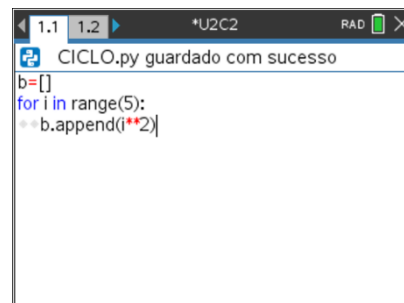
**Entrada:** `n` (número de elementos)  
**Saída:** lista (lista quadrados perfeitos)  
**Procedimentos:**  
Para `i` de 0 até `n-1` :  
    `Lista[i] ← i2`  
Escrever Lista



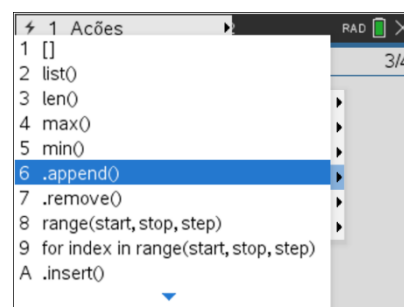
### IMPLEMENTAÇÃO DO ALGORITMO:

Vamos criar um programa para se entender melhor o que é um ciclo, assim como, o que é um processo iterativo.

- Inicie um novo programa em Python e designe-o por “CICLO”.
- Crie uma lista vazia escrevendo a instrução `b = [ ]`. Na linguagem Python, os elementos de uma lista são colocados entre `[ ]` (parêntesis retos), separados por vírgulas.
- Depois clique na tecla  e selecione submenu **4: Planos integrados**, depois **2: Controlo**, e por fim a opção **4: for index in range(size):**.
- A função `.append( )` permite o preenchimento de uma lista. Assim, `b.append(i**2)` acrescenta à lista `b` um novo elemento com o quadrado de `i`. Isto é, para cada valor de `i`, o valor de `i2` é acrescentado ao fim da lista.
- A variável `i` varia ente 0 e 4, o que corresponde a um ciclo de 5 valores, e, portanto, a acrescentar à lista `b` cinco elementos.









```
CICLO.py guardado com sucesso
b=[]
for i in range(5):
    b.append(i**2)
```

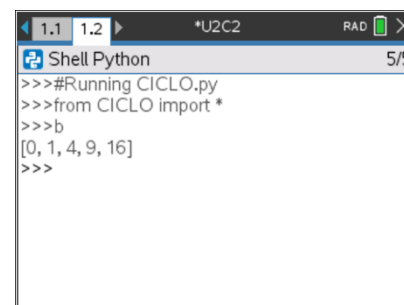


```
1 Acções
2 list()
3 len()
4 max()
5 min()
6 .append()
7 .remove()
8 range(start, stop, step)
9 for index in range(start, stop, step)
A .insert()
```

### OBSERVAÇÃO:

De notar que, o contador do ciclo FOR é, por defeito, inicializado em 0. A função `.append()` aplica-se apenas a listas, sendo a sua sintaxe: `nomelista.append(elemento a acrescentar)`. Pode-se aceder a esta função através do menu, pressionando tecla , de seguida opção **4: Planos integrados**, depois a opção **4: Listas** e, por fim, seleccionar **6: .append()**.

- Pressione simultaneamente as teclas  e  para verificar a sintaxe e guardar o programa.
- Execute o programa, clicando simultaneamente nas teclas  e , abrir-se-á uma nova página com o interpretador de Python (Shell) onde foi executado o programa.
- Por fim, obtenha a lista `b`, escrevendo o seu nome na linha de comando do Shell e pressionando .



```
Shell Python
>>>#Running CICLO.py
>>>from CICLO import *
>>>b
[0, 1, 4, 9, 16]
>>>
```

### OBSERVAÇÃO:

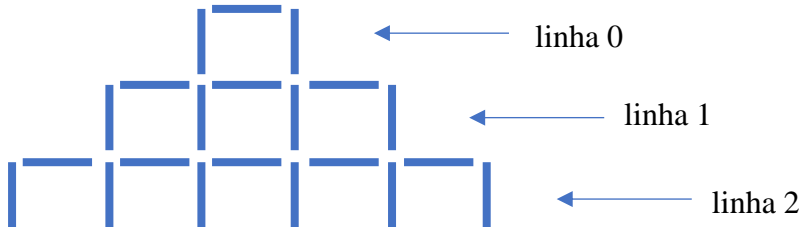
Num ciclo ou numa outra qualquer estrutura de uma função com instruções com recuo, qualquer escrita recuada de um comando faz parte do ciclo ou da função. O fim do ciclo ou da função é definido pela saída do recuo.



### APLICAÇÃO DAS APRENDIZAGENS:

#### Sequências com Construção com Fósforos

Na construção abaixo, a primeira linha, denominada "linha 0", é formada por 3 fósforos (2 na vertical e 1 na horizontal), a segunda linha por 7 fósforos (4 na vertical e 3 na horizontal), a "linha 2" (3ª linha) por 11 fósforos, e assim sucessivamente.



Por quantos fósforos será formada a linha 4?

Construa um programa que, usando um ciclo FOR, permita obter o número de fósforos numa dada linha desta construção.

Execute o programa para obter o número de fósforos que contém a centésima linha.

```
*CONST_FOSF.py 5/5
def nf(n):
    x=3
    for i in range(1,n+1):
        x=x+4
    return x
```

```
CONST_FOSF.py 3/5
def nf(n):
    x=3
    for i in range(1,n+1):
        x=x+4

Shell Python 3/5
>>>#Running CONST_FOSF.py
>>>from CONST_FOSF import *
>>>nf(4)
19
```

### SUGESTÃO:

Pode aceder a variáveis ou funções definidas num programa, antes ou depois de o executar no interpretador, pressionando a tecla `var`.

