

```
object to mirror  
mirror_mod.mirror_object
```

```
operation == "MIRROR_X":  
    mirror_mod.use_x = True  
    mirror_mod.use_y = False  
    mirror_mod.use_z = False  
operation == "MIRROR_Y":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = True  
    mirror_mod.use_z = False  
operation == "MIRROR_Z":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = False  
    mirror_mod.use_z = True
```

```
@selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier  
mirror_ob.select = 0  
= bpy.context.select  
data.objects[one  
print("please select e
```

```
--- OPERATOR CLASSES ---
```

```
types.Operator):  
    X mirror to  
    object.mirror_mirror_x  
    mirror X"
```

Laços de repetição For e While

Prof. Demétrius de Castro
demdecastro@gmail.com
83 9 87730383

COMO FOI O
EXERCÍCIO?



Laços de Repetição

O que é estrutura de repetição em Python?

Em algumas situações é comum que uma mesma instrução (ou conjunto delas) precise ser executada várias vezes seguidas. Nesses casos, normalmente utilizamos um loop (ou laço de repetição) que permite executar o mesmo bloco de código enquanto uma condição é atendida. Em Python, os loops são codificados com as estruturas de repetição for e while.

Laços de Repetição

Vamos imaginar a seguinte situação:

Precisamos criar um código que faça uma contagem de 0 a 5, como podemos fazer?

Usando if's?

Vamos fazer.

Laços de Repetição

```
inicio = 0
fim = 5
if inicio <= fim:
    print (inicio)
    inicio = inicio + 1
if inicio <= fim:
    print (inicio)
    inicio = inicio + 1
if inicio <= fim:
    print (inicio)
    inicio = inicio + 1
if inicio <= fim:
    print (inicio)
    inicio = inicio + 1
if inicio <= fim:
    print (inicio)
    inicio = inicio + 1
```

Laços de Repetição

Não parece ser uma solução muito elegante e completamente inviável, tendo em vista que essa contagem pode ser com valores na casa de milhões, bilhões, ou maior.

Escrevendo desta forma, você teria que escrever milhões de linha de código, repetido, para uma solução bastante simples.

Usando os laços de repetição, este mesmo código pode ser escrito em apenas duas linhas de código, indente do tamanho da contagem.

Laços de Repetição

O laço for nos permite percorrer os itens de uma coleção e, para cada um deles, executar o bloco de código declarado no loop.

Estrutura do FOR:

for variável **in** condição de parada:

bloco de código a ser executado

A condição de parada pode ser um número, string, lista, entre outros.

Laços de Repetição

Vamos fazer o mesmo exemplo da contagem de 0 a 5 utilizando o laço de repetição for.

```
for i in range(6):  
    print(i)
```

Explicando o código:

for – Chamada do comando

i – variável

in – atribuição de sobre o que será aplicado o for

range – função nativa do Python que realiza uma contagem

(6) – intervalo a ser contado*

Laços de Repetição

A função `range()`, possui uma variedade de uso.

Por padrão, ela inicia a contagem com o valor zero, mas podemos alterar esse valor de início no momento da sua declaração.

Ex.:

Contar de 1 a 5

```
for i in range(1, 6):  
    print(i)
```

Contar de 5 a 10

```
for i in range(5, 11):  
    print(i)
```

Laços de Repetição

Também podemos pedir que o usuário informe qual o valor de parada da contagem.

Ex.:

```
n = int(input("Digite o valor: "))  
for i in range(n+1):  
    print(i)
```

PYTHON

Laços de Repetição

A função `range()`, também permite que a contagem de valores com “saltos”.

Ex.:

Contar os valores pares de 0 a 10

```
for i in range(0, 10, 2):  
    print(i)
```

PYTHON

Laços de Repetição

Faça a contagem de valor impares de 0 a 20.

Laços de Repetição

Até agora vimos que podemos fazer várias formas de contagem, mas perceberam que os valores sempre são impressos em linhas diferentes? E se quisermos imprimir na mesma linha?

Neste caso, usamos o comando “end”.

Ex.:

```
for i in range(10):  
    print(i, end="")
```

Laços de Repetição

Faça

Escreva um código que receba um valor do usuário e imprima a sua quantidade de linhas e colunas, de modo crescente.

Ex.:

Caso o usuário digite o valor 4, retorne:

0

01

012

0123

01234

Laços de Repetição

While

Funciona quase da mesma forma que o comando for, a diferença é que precisamos “setar” os valores de controle e parada antes do comando while. Assim como precisamos fazer a incrementação da variável de controle

Ex.:

```
controle = 0
```

```
parada = 5
```

```
while controle <= parada:
```

```
    print(controle)
```

```
    controle = controle + 1
```

Laços de Repetição

Faça

Usando o comando While, escreva um código que receba uma string do usuário e a imprima 5 vezes na tela.