Laboratório de Programação 1

Aula 07

Mário Hozano
professor@hozano.com

Ciência da Computação
UFAL - Arapiraca

Relembrando a aula anterior...

- O que são estruturas de repetição?
- Quais comandos em Python promovem repetição?
- O que é iteração?
- Qual a sintaxe do comando *while* ?
- Para que serve a instrução *break* ?

Roteiro da aula

- Repetições com o comando *for*
- Sequências do *for* – Listas
- Sequências do *for* - *Strings*
- Gerando listas de inteiros
- Gerando listas com entradas
- Exemplo de comparação entre o *while* e o *for*

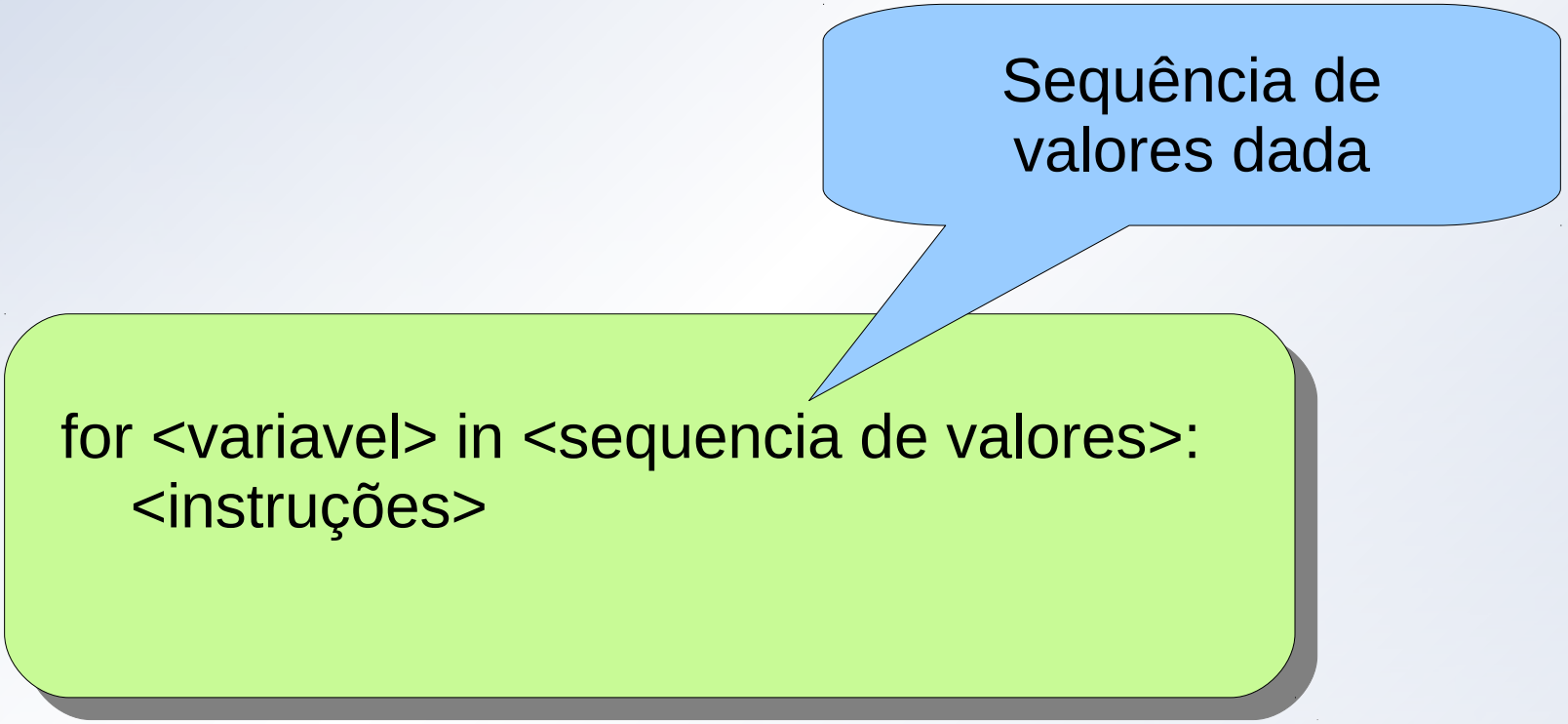
Repetições com *for*

- Diferentemente do *while*, o comando *for* permite repetir um conjunto de instruções independentemente de uma condição lógica
- Com o *for*, é necessário saber quantas iterações devem ser realizadas
- O corpo da variável será executado uma vez para cada valor na sequência de valores
- A cada iteração a variável receberá, em ordem, cada um dos valores dados na sequência de valores
- No slide seguinte temos a sintaxe do comando *for*

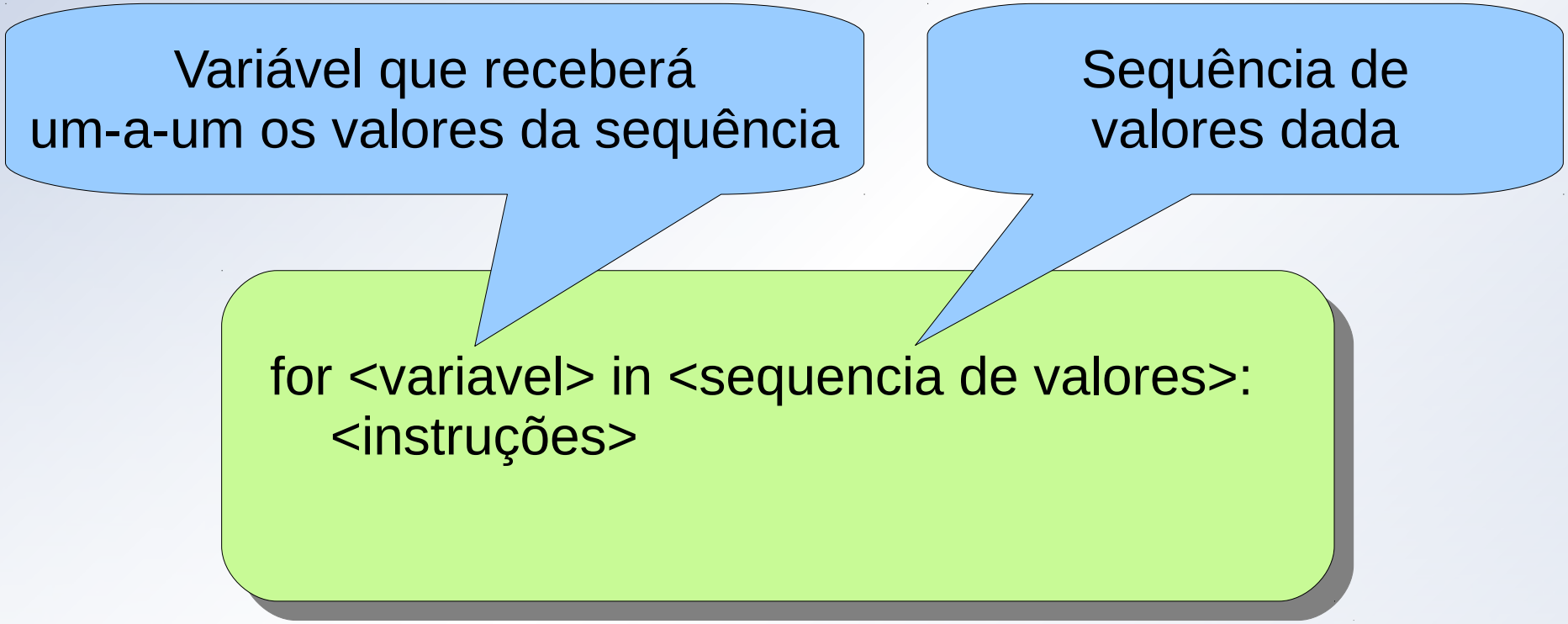
Repetições com *for*

```
for <variavel> in <sequencia de valores>:
    <instruções>
```

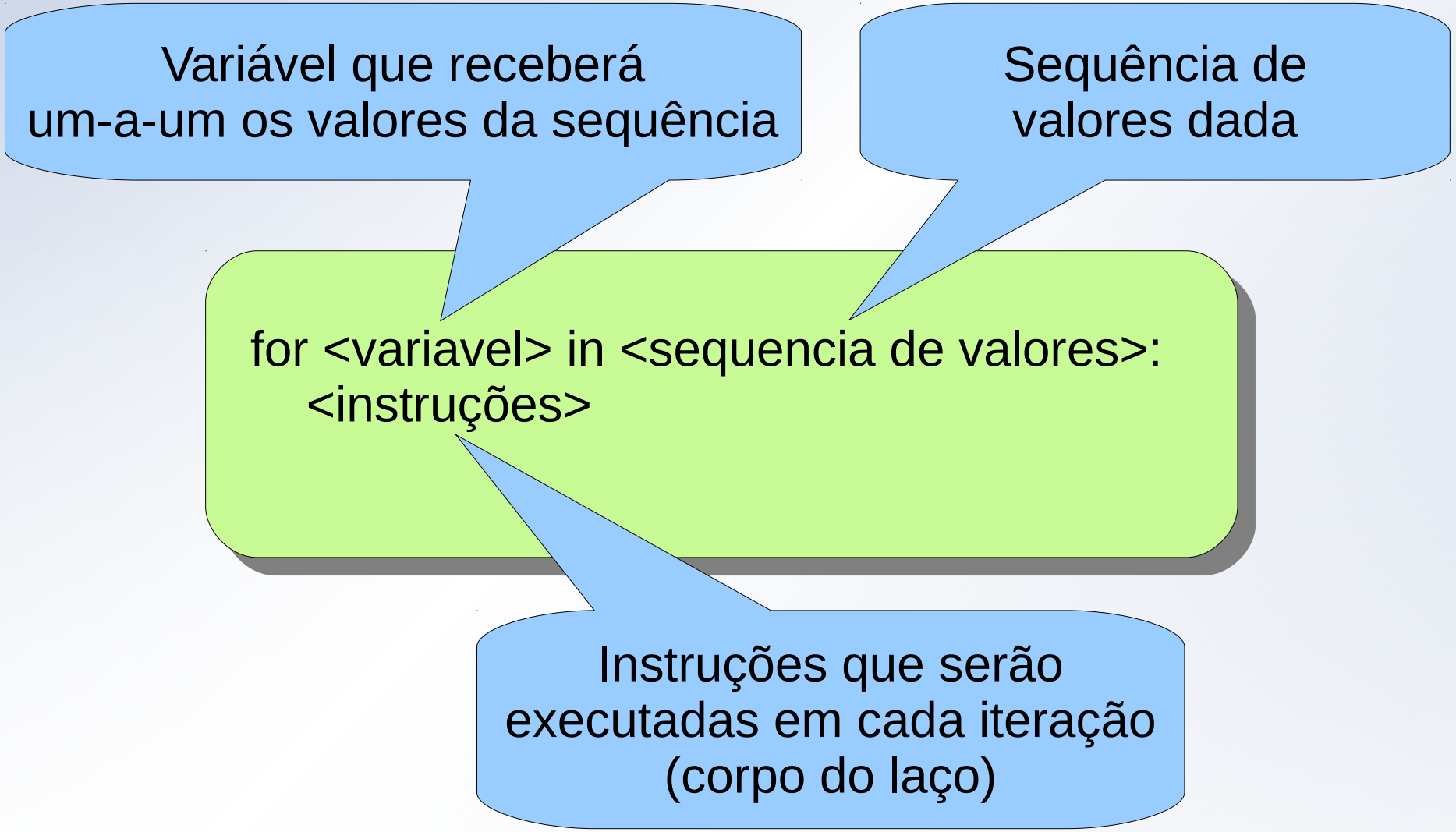
Repetições com *for*



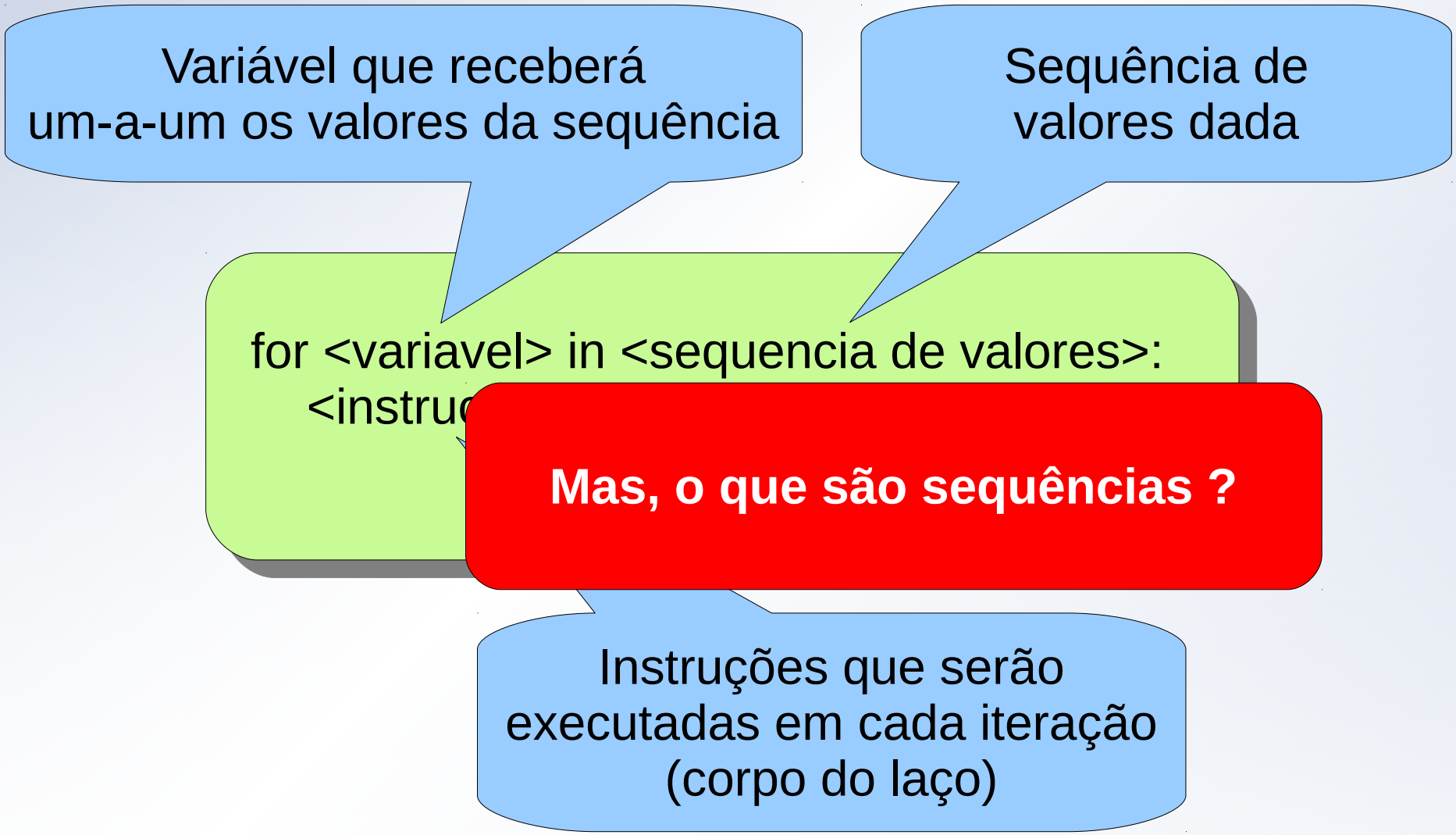
Repetições com *for*



Repetições com *for*



Repetições com *for*



Repetições com *for* - Sequências

- Uma lista é uma sequência de valores
- Uma lista pode conter uma coleção de valores e são apresentados com colchetes como delimitadores e vírgulas como separadores
- Em muitas linguagens estruturas como *array* e vetor possuem comportamentos parecidos com os das listas

```
>>> lista1 = [1,2,3,4,5,6,7]
>>> lista2 = ["a", "b", “c”, “d”]
>>> lista3 = [30, "UFAL", True]
```

- Mais a frente o estudo de listas será aprofundado :)

Repetições com *for* - Sequências

- Uma *string* também é uma sequência
- Neste caso, uma *string* é uma sequência de letras
- Assim, os comandos *for* também podem iterar sobre as letras de uma *string*

```
>>> list("casa")
['c', 'a', 's', 'a']
>>> list("UFAL")
['U', 'F', 'A', 'L']
```

- Mais a frente o estudo de *strings* será aprofundado :)

Repetições com *for* - Exemplo

- O *for* itera sobre sequências (como listas e *strings*)

```
for num in [1,2,3,4,5]:
    print(num)
```

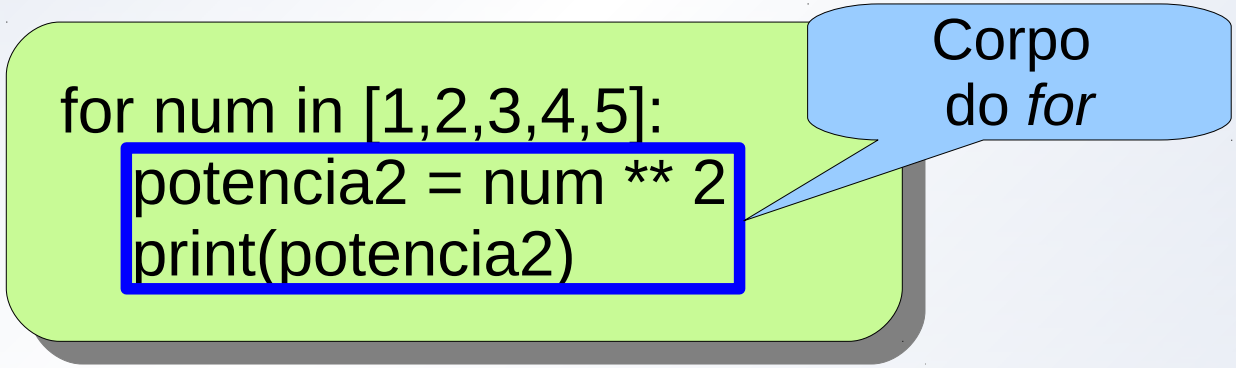
```
for palavra in [“UFAL”, “ASA”, “Arapiraca”]:
    print(palavra)
```

```
for letra in “UFAL”:
    print(letra)
```

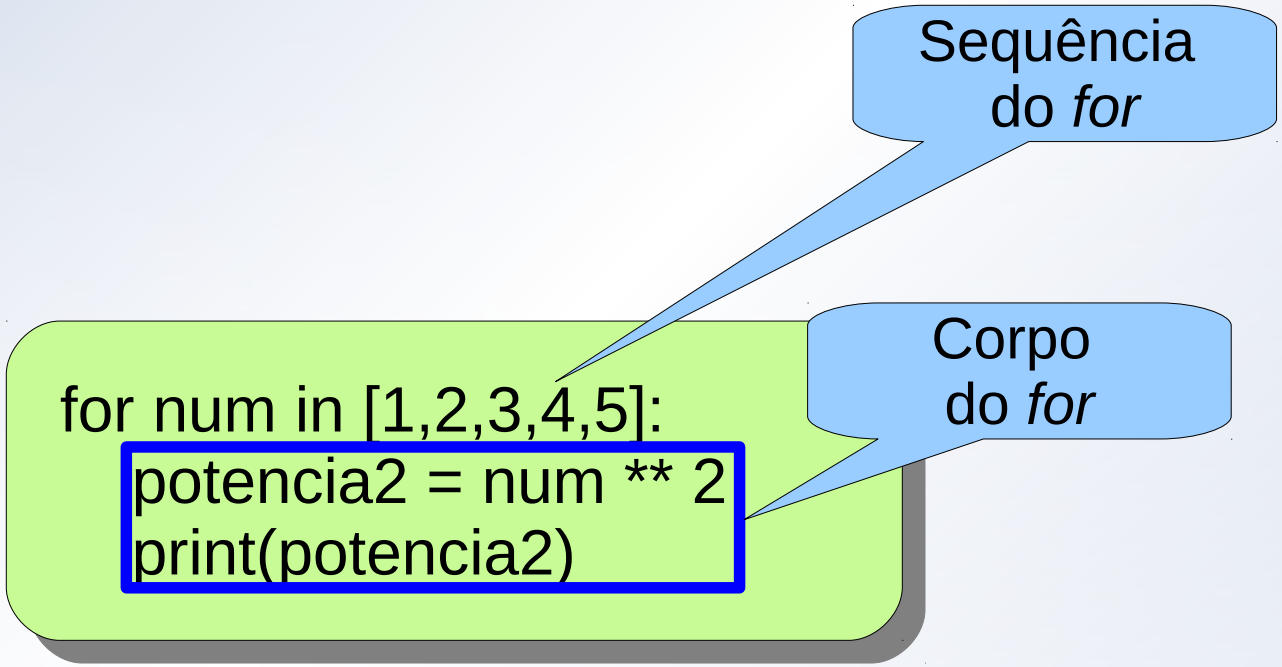
Repetições com *for* - Exemplo

```
for num in [1,2,3,4,5]:
    potencia2 = num ** 2
    print(potencia2)
```

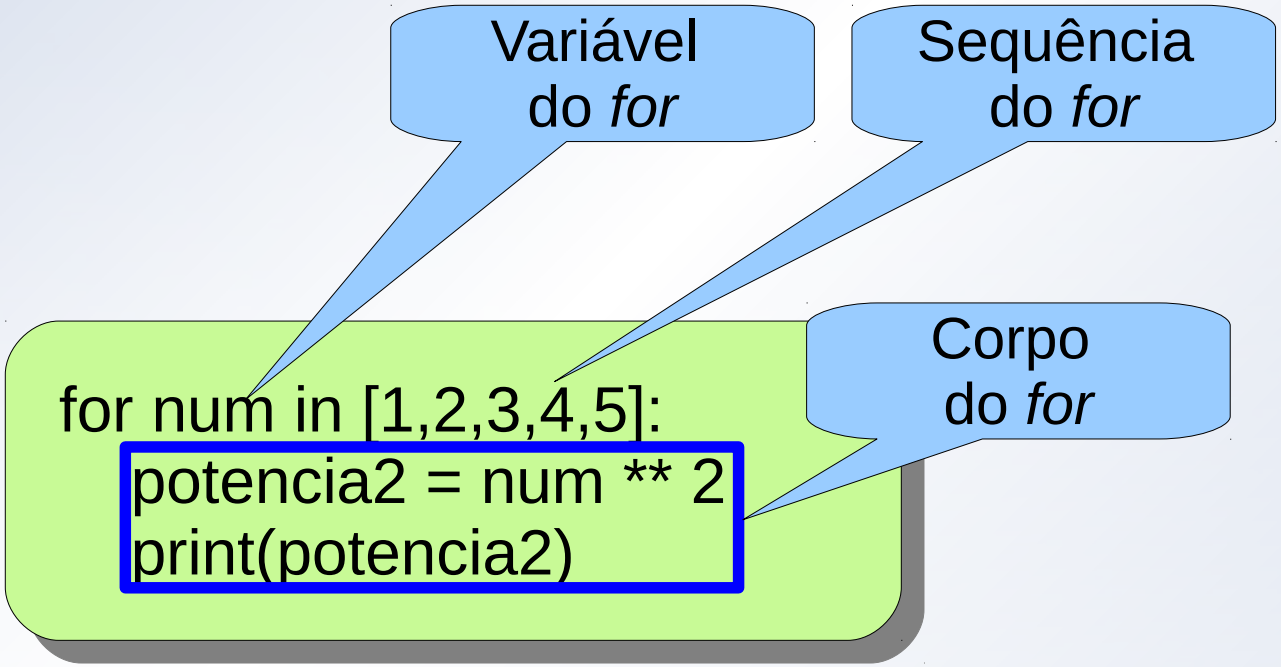
Repetições com *for* - Exemplo



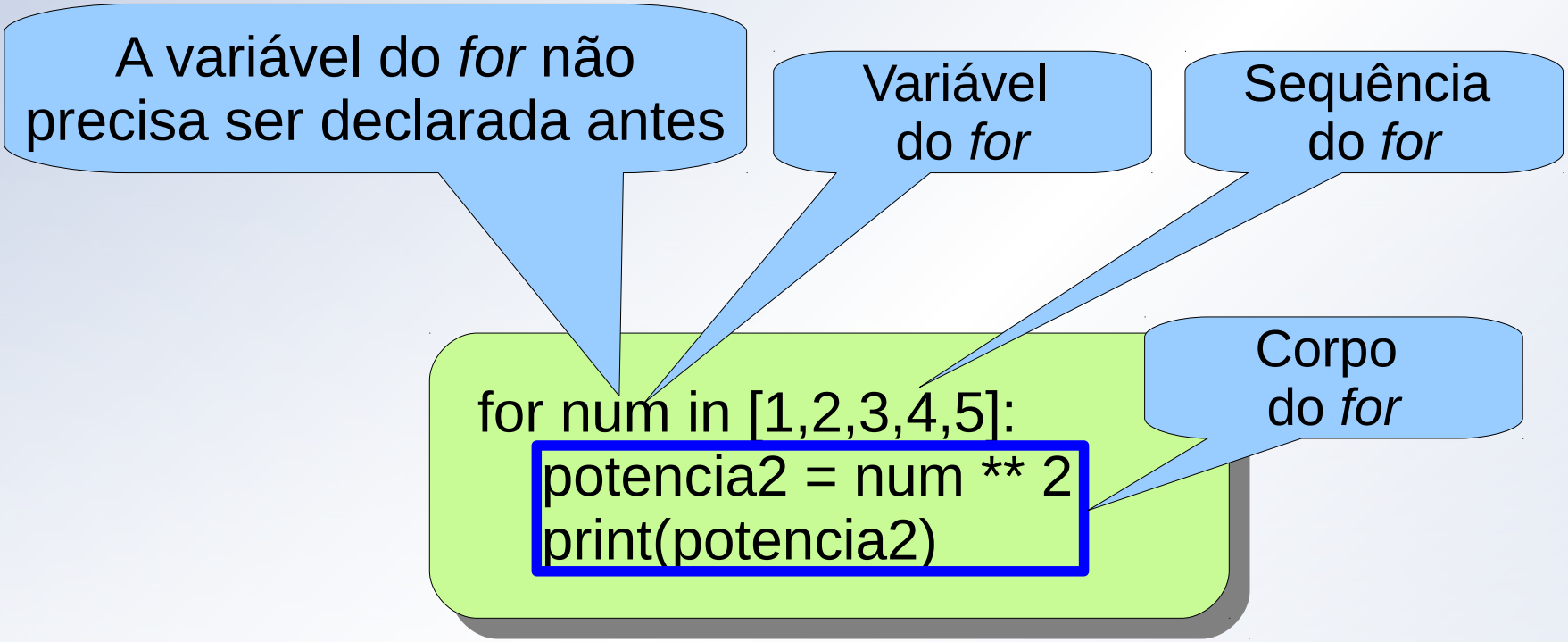
Repetições com *for* - Exemplo



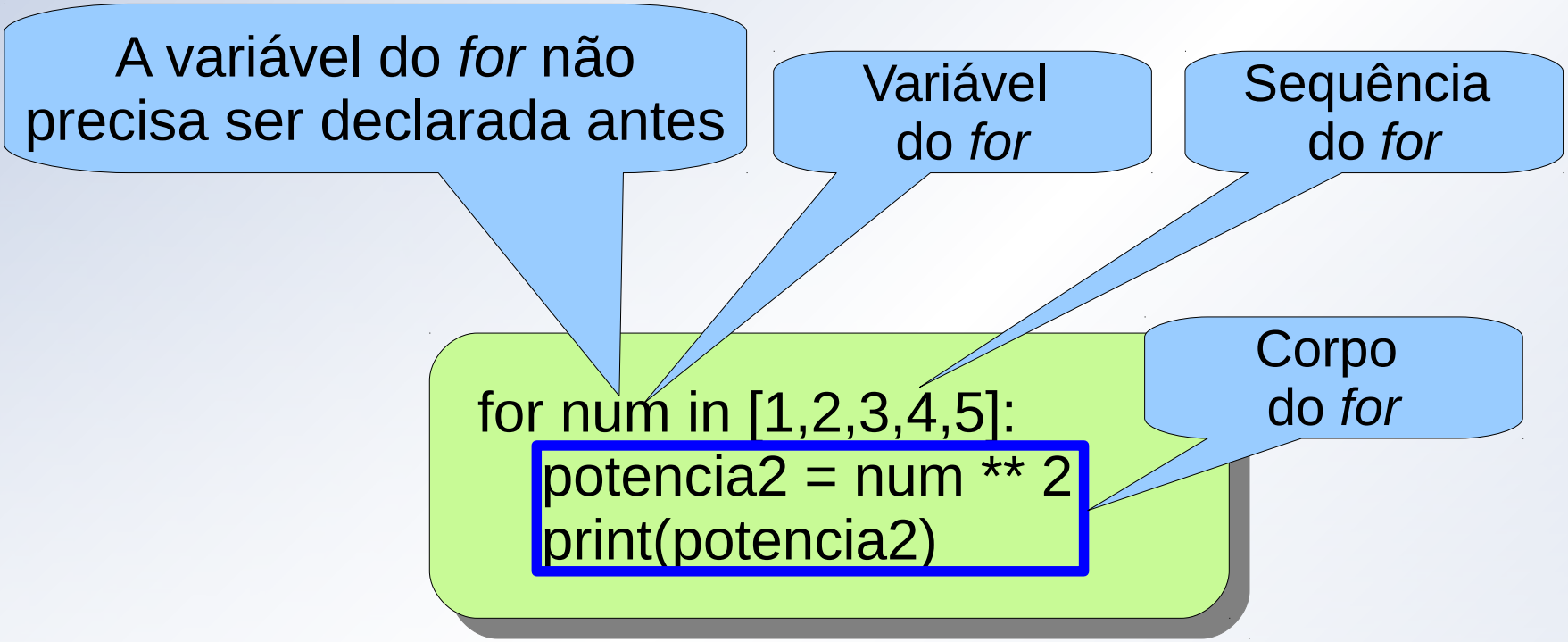
Repetições com *for* - Exemplo



Repetições com *for* - Exemplo

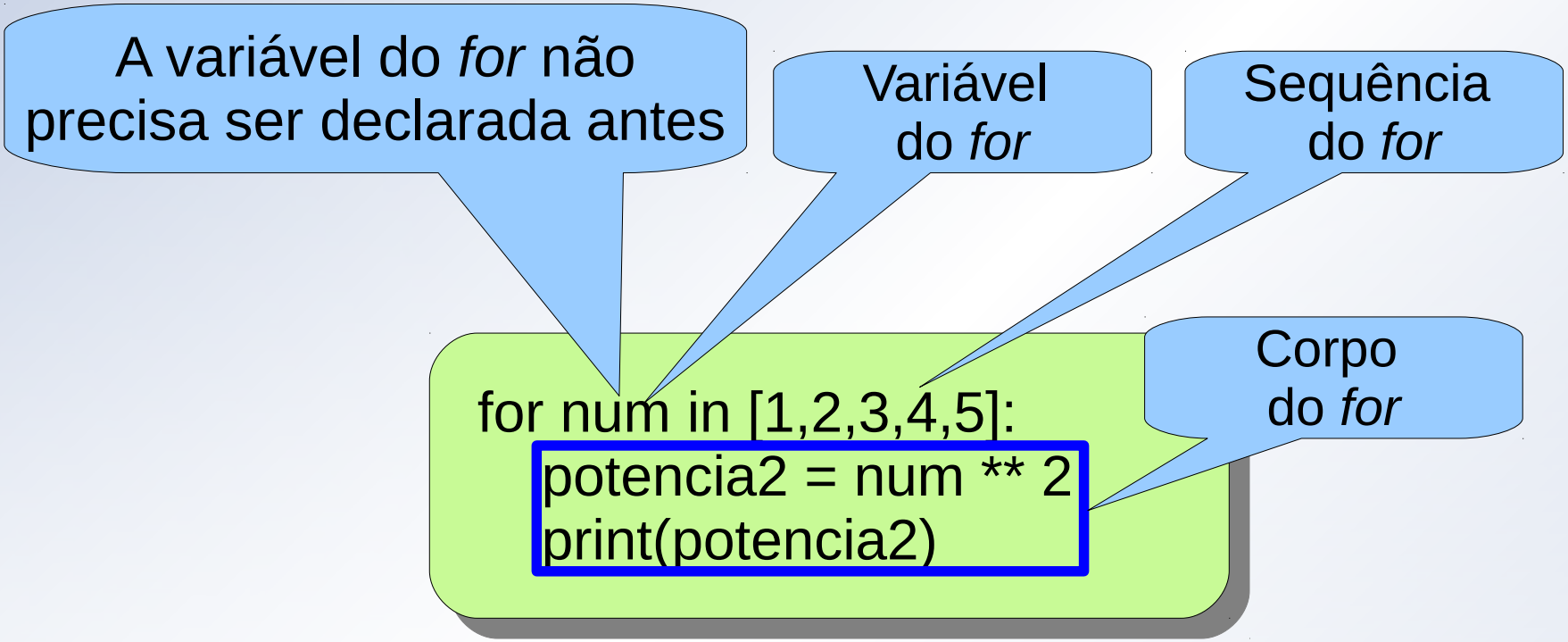


Repetições com *for* - Exemplo



O que faz o código acima ?

Repetições com *for* - Exemplo



O que faz o código acima ?

Como iterar uma sequência de 1000 valores ?

Repetições com *for* – Gerando Listas de Inteiros

- Criar listas de inteiros com muitos valores pode ser muito incômodo
- Python fornece a função *range()* para a criação de listas de números inteiros
- Muitos programas utilizam a função *range()* juntamente com o comando *for*
- A função *range()* pode ser chamada com 1, 2 ou 3 argumentos como mostrado no slide seguinte

Repetições com *for* – Gerando Listas de Inteiros

- *range(n)* – Retorna uma lista de inteiros de *0* a *n-1*

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

- *range(m,n)* – Retorna uma lista de inteiros de *m* a *n-1*

```
>>> range(2,10)
[2, 3, 4, 5, 6, 7, 8, 9]
```

- *range(m,n,k)* – Retorna uma lista de inteiros de *m* a *n-1* com passo *k*

```
>>> range(2,10,2)
[2, 4, 6, 8]
```

Repetições com *for* – Listas a partir de entradas

- Em algumas situações é necessário fazer listas com dados indicados pelo usuário

```
num1 = input("Num1: ")
num2 = input("Num2: ")
num3 = input("Num3: ")

numeros = [num1, num2, num3]

for num in numeros:
    print(num**2)
```

O que faz o código acima ?

Repetições com *for* – Listas a partir de entradas

- Utilizando a função *append()* para inserir valores em listas

```
numeros = []

while True:
    numero = input("Digite um número ou 0 para sair: ")
    if not numero:
        break
    numeros.append(numero)

for num in numeros:
    print(num**2)
```

O que faz o código acima ?

Repetições com *for* – Exemplo com *while* e *for*

- Imprimindo inteiros de 0 a 1000 com *while*

```
num = 0
while (num < 1001):
    print(num)
    num = num + 1
```

- Imprimindo inteiros de 0 a 1000 com *for*

```
for num in range(1001):
    print(num)
```

Exercícios

1. Escreva um programa que calcule o fatorial dos 10 primeiros inteiros positivos.
2. Escreva um programa que verifica os fatores de um número dado utilizando o *for*.
3. Re-escreva o programa de funções matemáticas permitindo que o usuário possa repetir operações 10 vezes.
4. Crie um programa que exiba na tela a tabuada de 1 a 9 utilizando o *for*.
5. Escreva um programa que indique os 10 primeiros números primos.