Laboratório de Programação 1

Aula 06

Mário Hozano
professor@hozano.com

Ciência da Computação
UFAL - Arapiraca

Relembrando a aula anterior...

- O que são estruturas condicionais?
- O que são expressões lógicas?
- Quais são os operadores relacionais?
- Para que serve os operadores lógicos?
- Diferencie os condicionais simples, composto, encadeado e aninhado?
- O que é uma função recursiva ?

Roteiro da aula

- Estruturas de Repetição
- Repetições com *while*
- A instrução *break*
- Exemplos com o comando *while*
- Exercícios

Estruturas de Repetição

- A repetição de instruções é comum na execução de algoritmos
- Em algoritmos e programação isto pode ser obtido através das estruturas de repetição
- Em programação cada repetição de um conjunto de comandos é chamada de **iteração**
- As estruturas de repetição podem ser implementadas utilizando o comando *while* (enquanto) e o comando *for* (para)
- As estruturas de repetição também são conhecidas por laços ou *loops*

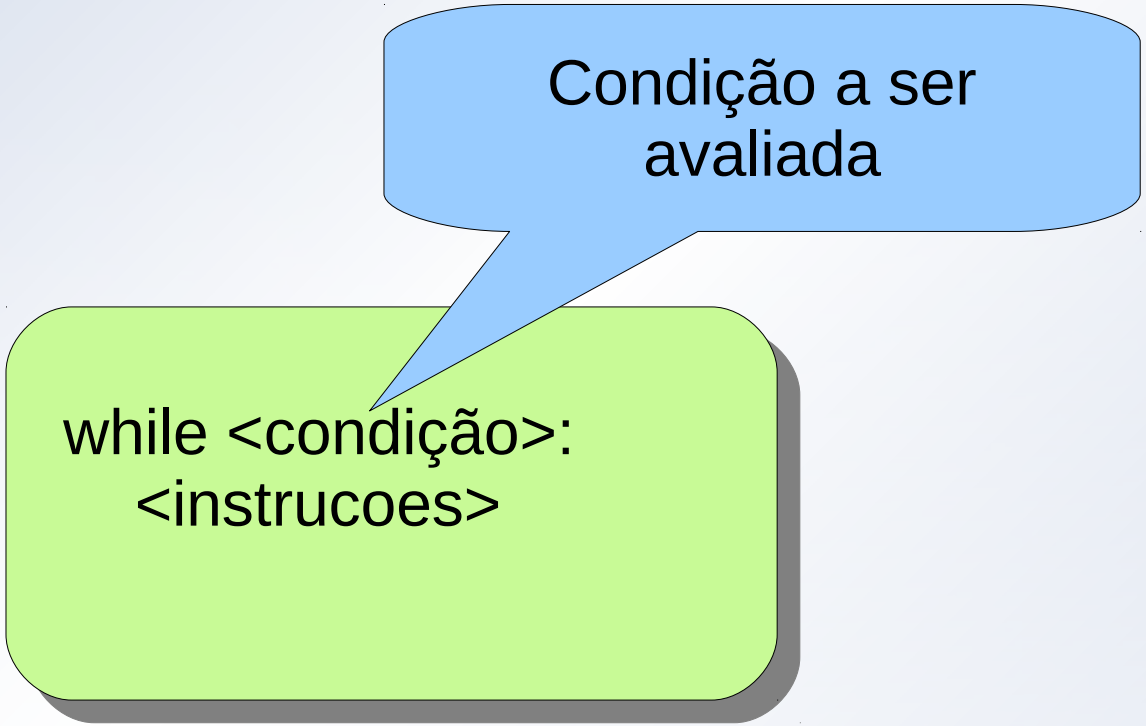
Repetições com w*hile*

- O comando *while* permite repetir um conjunto de instruções a partir de uma expressão lógica avaliada
- Esta expressão indica a condição que determina a iteração ou não das instruções do laço
- Caso a condição tenha valor *True*, é realizada uma iteração do laço
- Caso seja *False,* o bloco de instruções do *while* é ignorado e o interpretador passa para a próxima linha do programa
- No slide seguinte temos a sintaxe do comando *while*

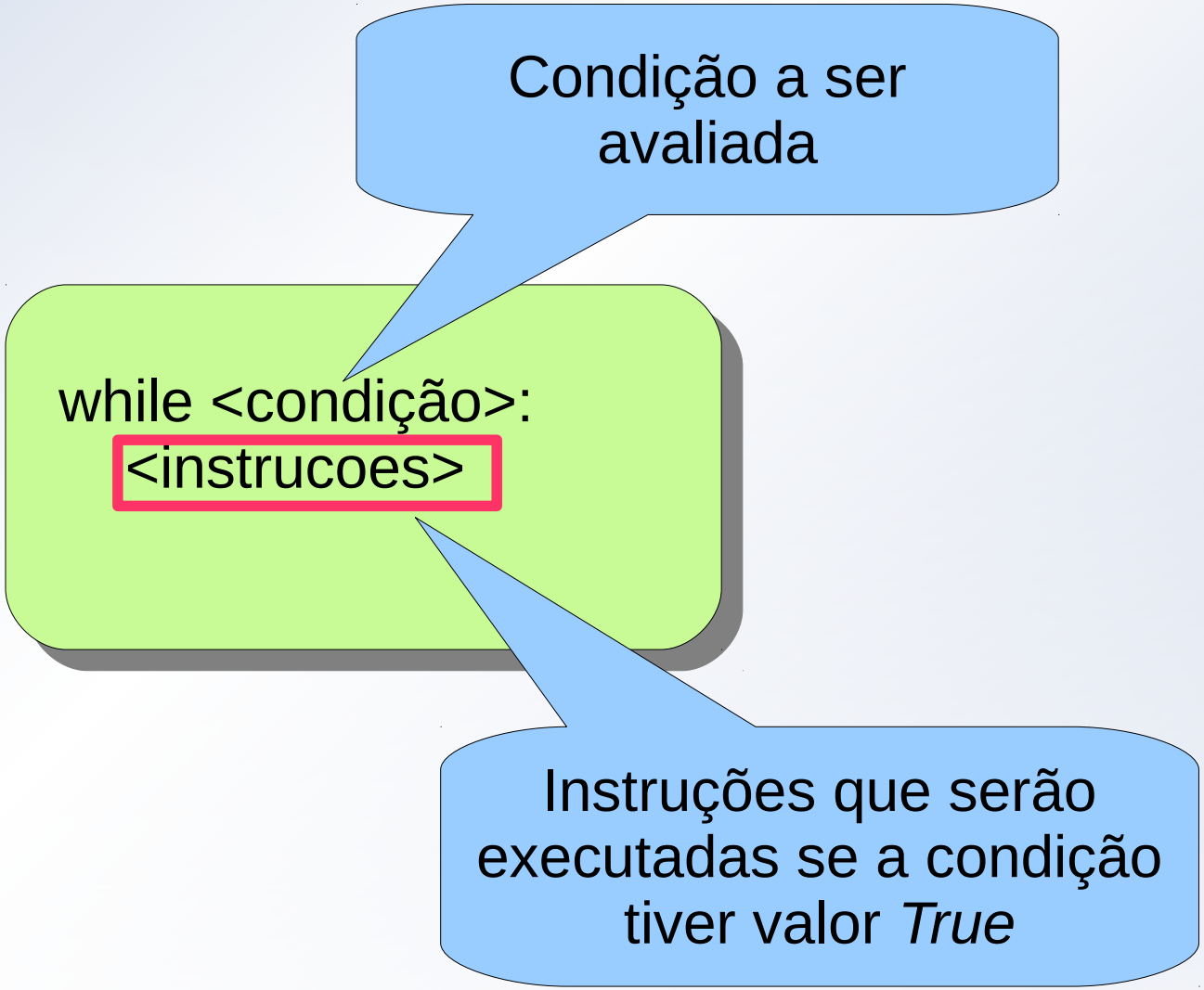
Repetições com w*hile* - Sintaxe

```
while <condição>:
    <instrucoes>
```

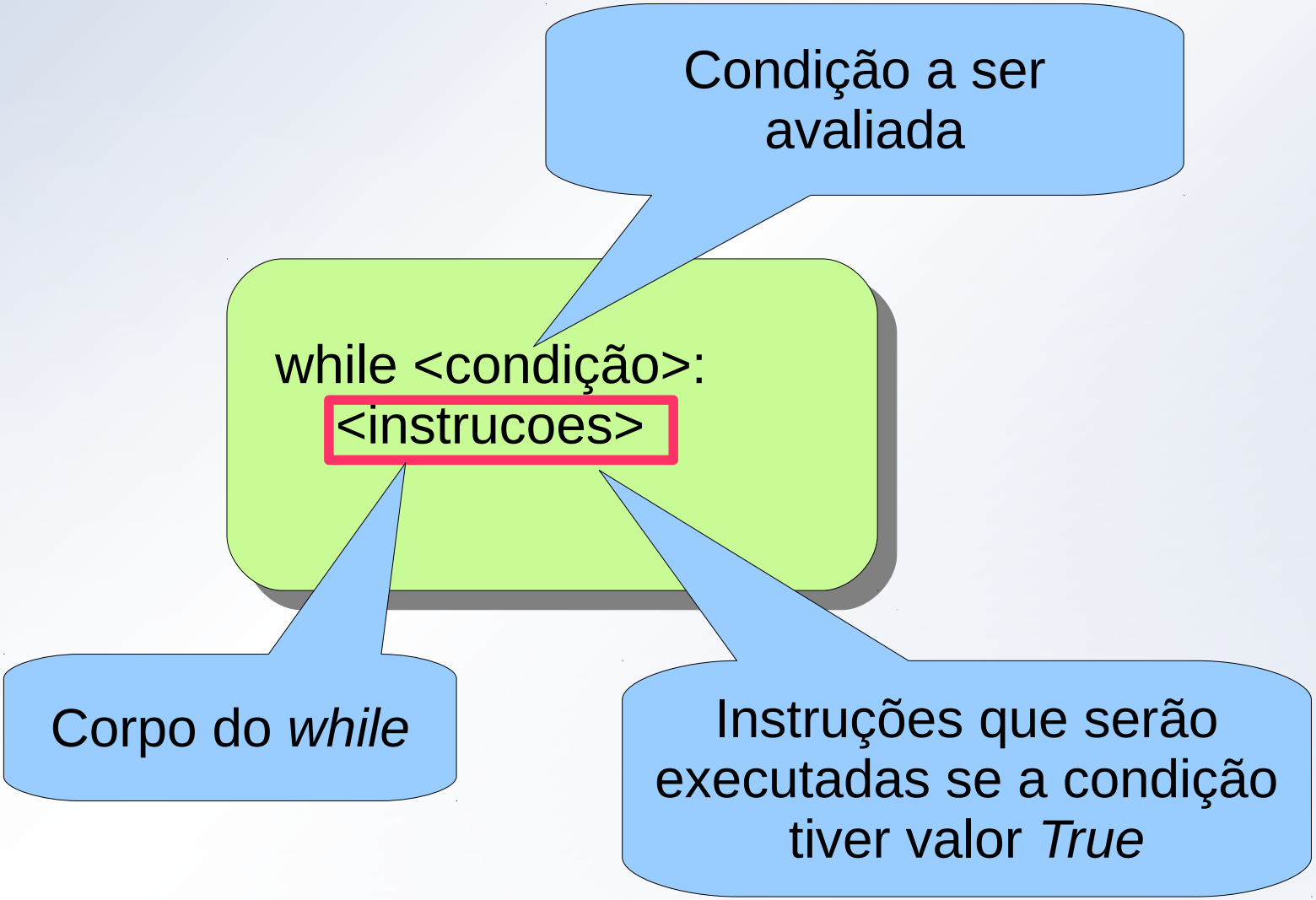
Repetições com w*hile* - Sintaxe



Repetições com w*hile* - Sintaxe



Repetições com w*hile* - Sintaxe



Repetições com while - Exemplo

- A seguir uma função que realiza uma contagem regressiva utilizando *while*

```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)
```

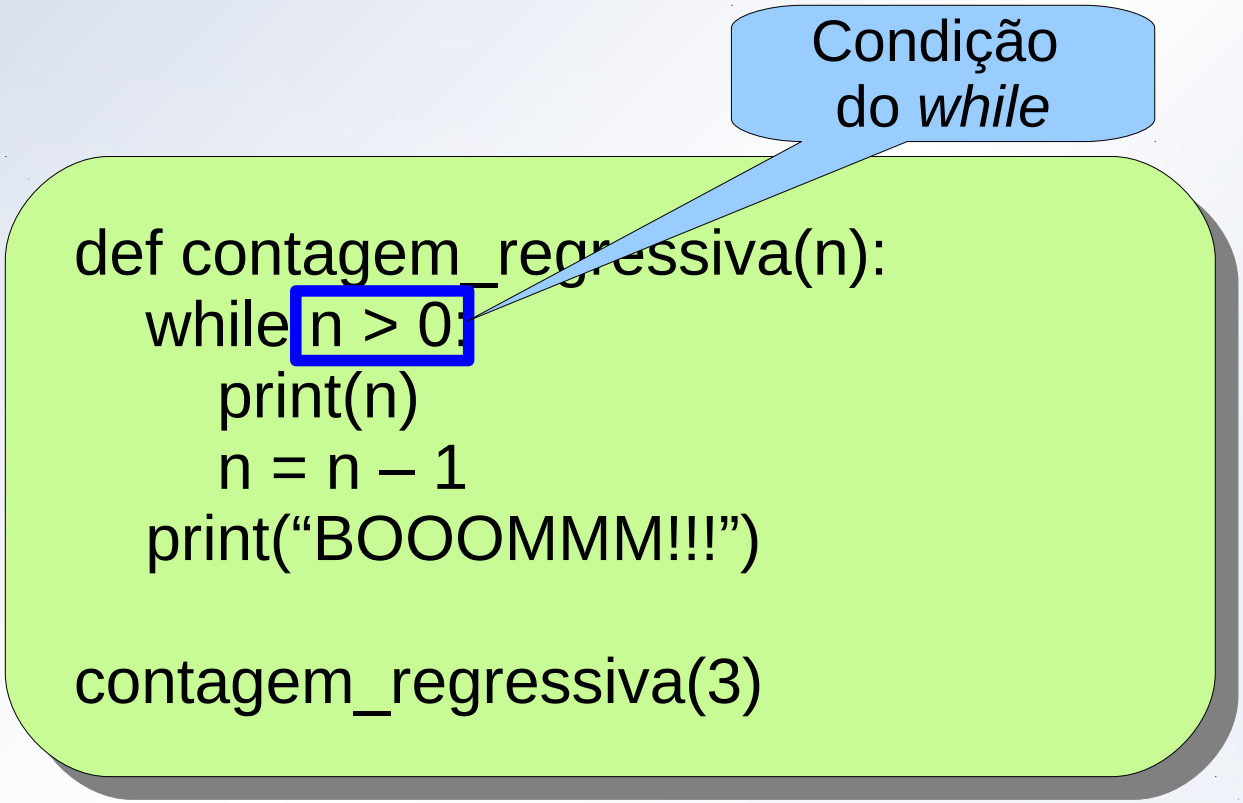
O que acontece se chamarmos o comando abaixo ?
>>> contagem_regressiva(3)

Repetições com while - Exemplo

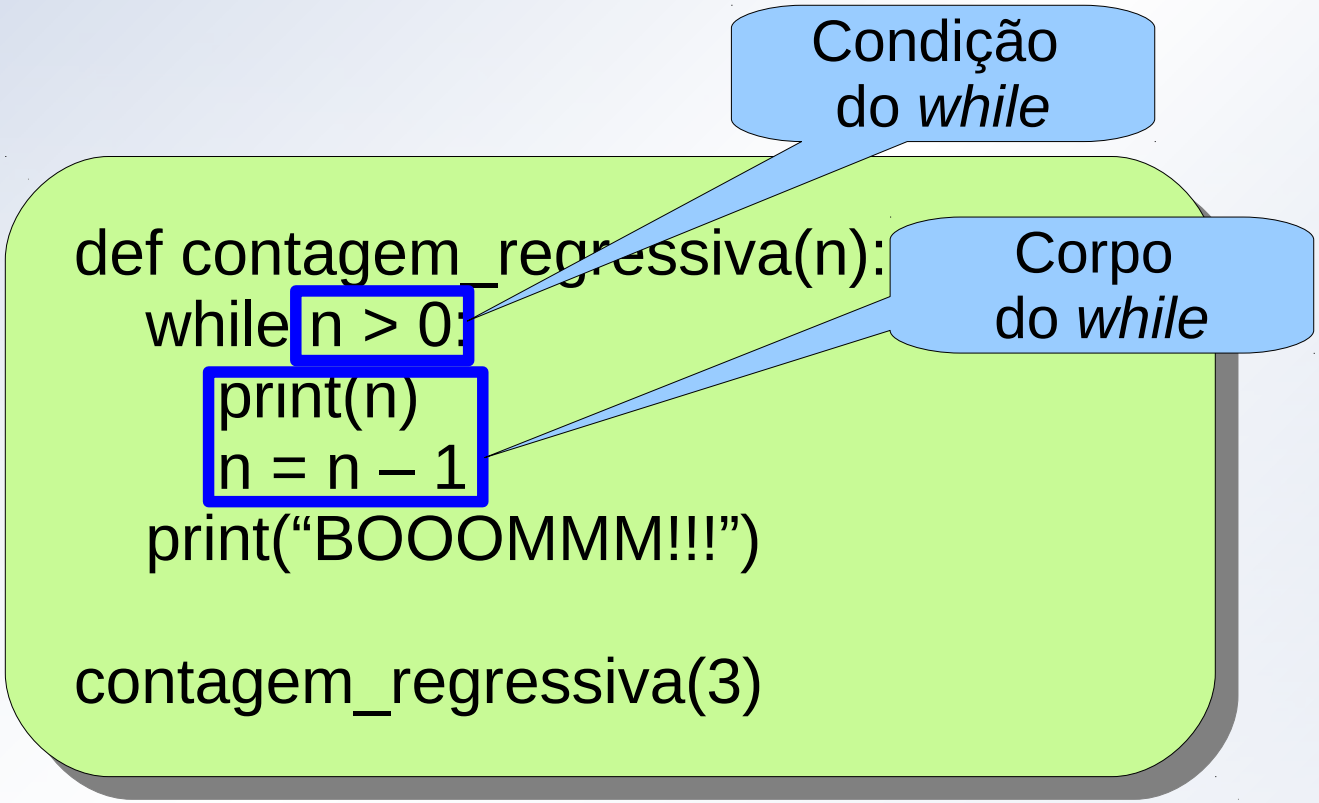
```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

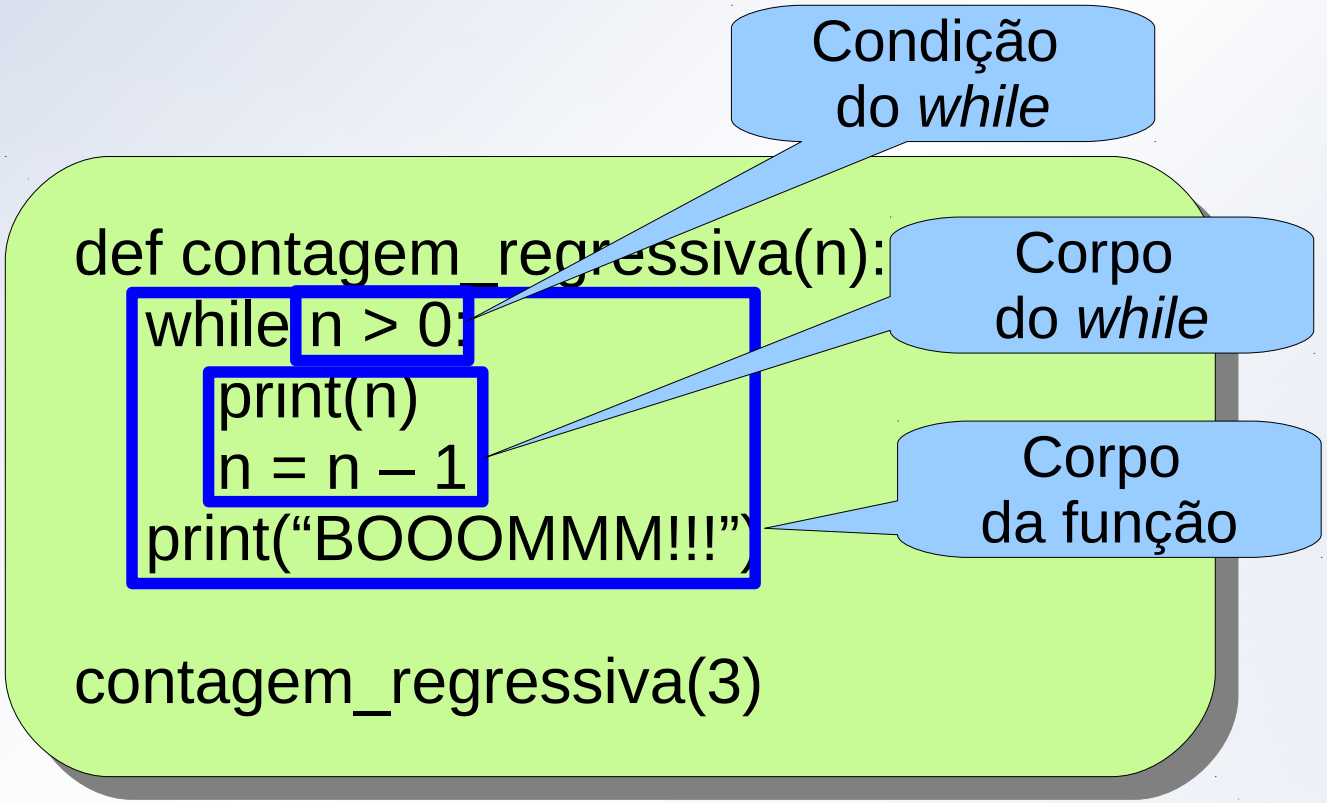
Repetições com while - Exemplo



Repetições com while - Exemplo



Repetições com while - Exemplo



Repetições com while - Exemplo

O interpretador lê o nome da função e passa para a próxima linha após a função

```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

Repetições com while - Exemplo

```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

A função é chamada com argumento igual a 3

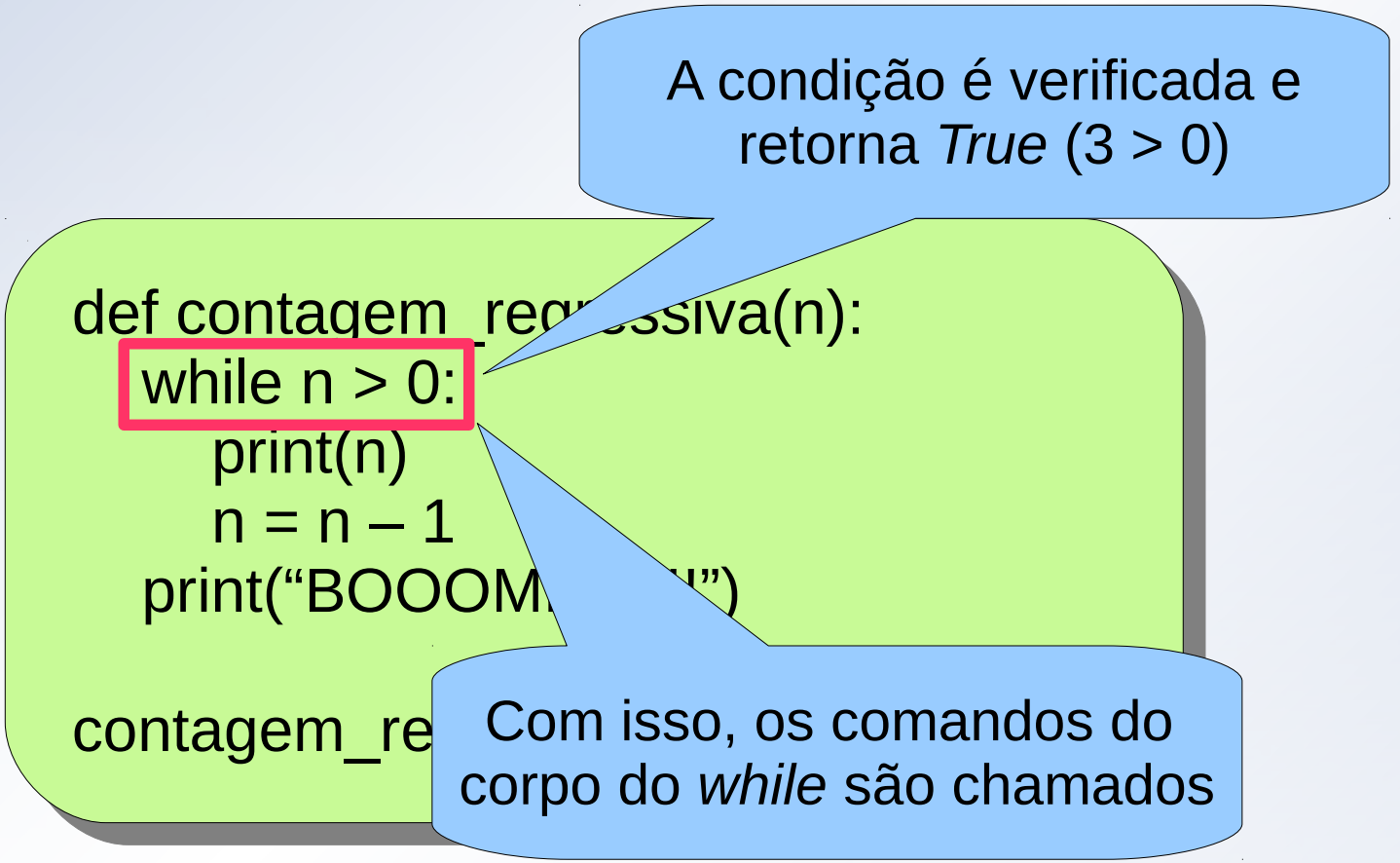
Repetições com while - Exemplo

A função é iniciada com n = 3

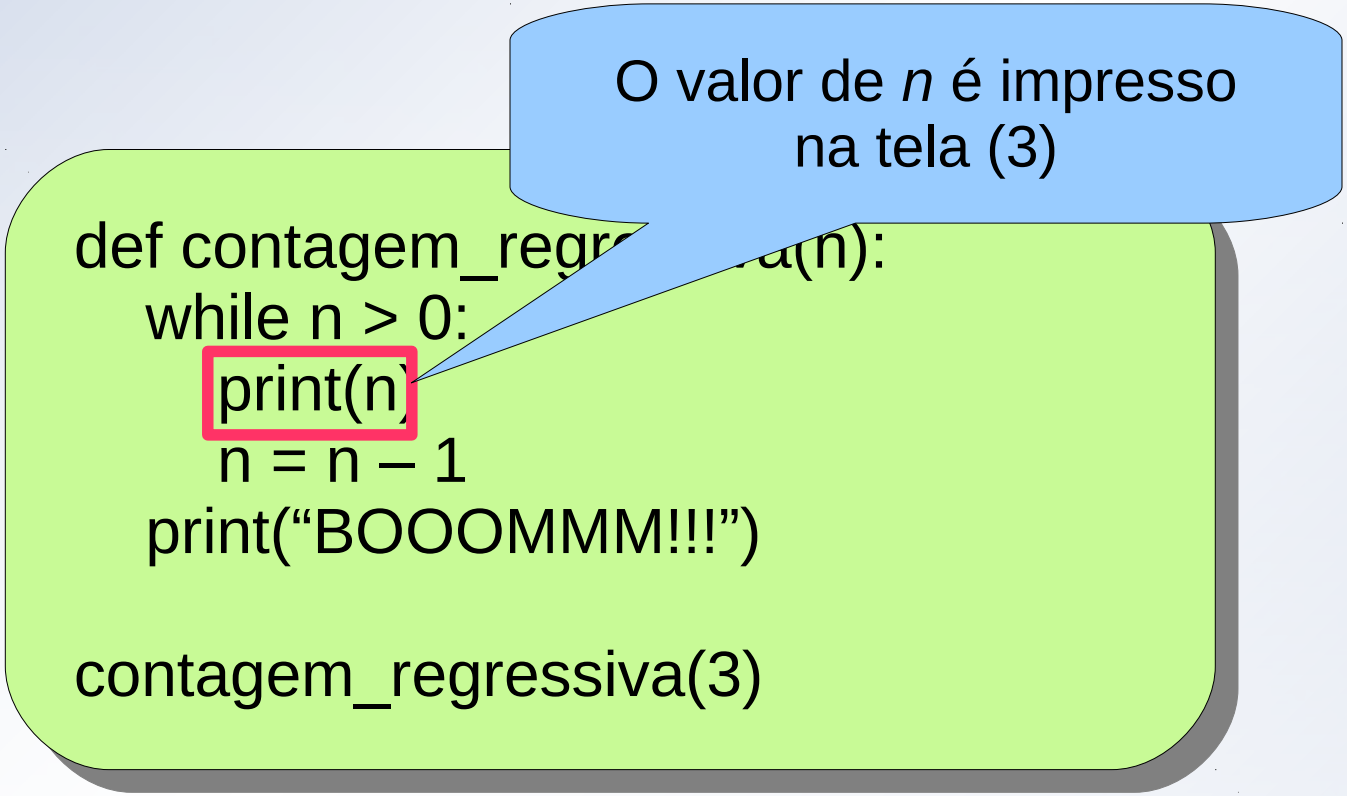
```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

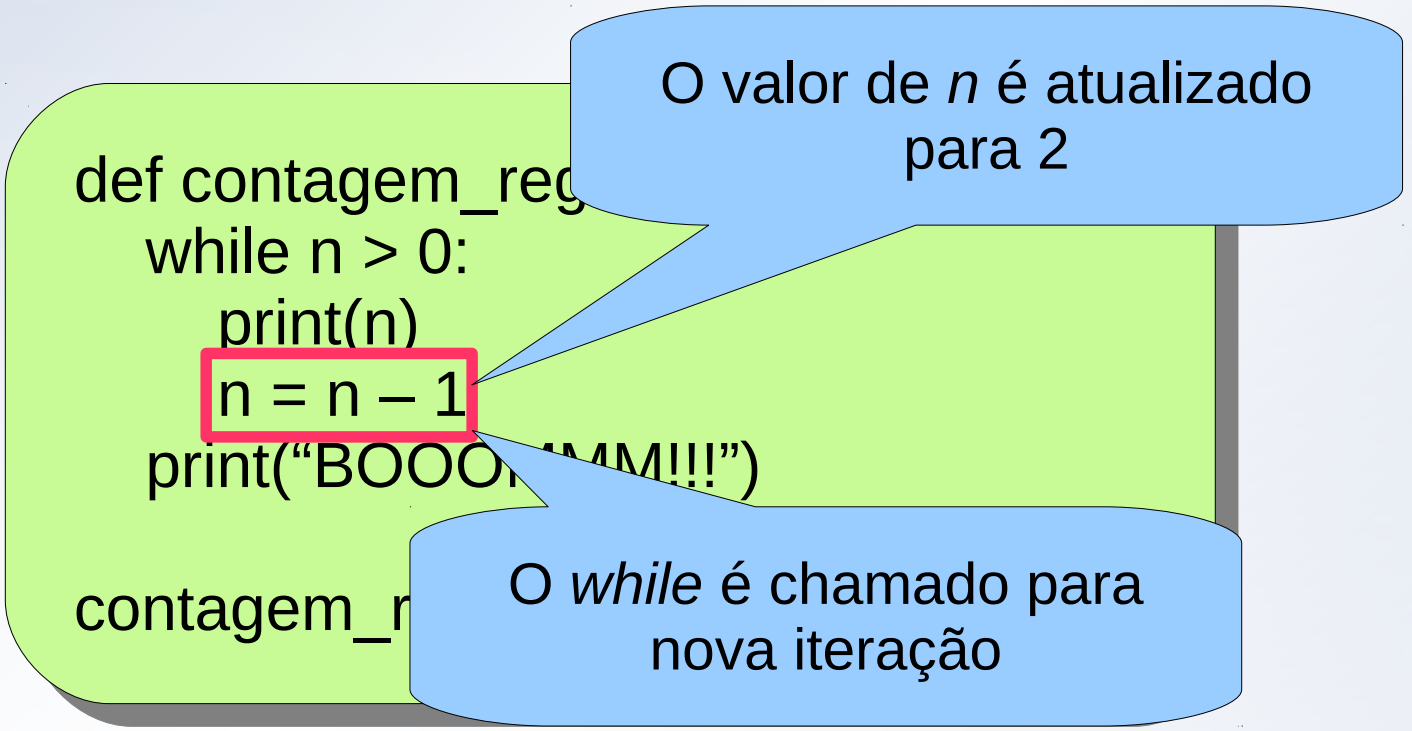
Repetições com while - Exemplo



Repetições com while - Exemplo



Repetições com while - Exemplo



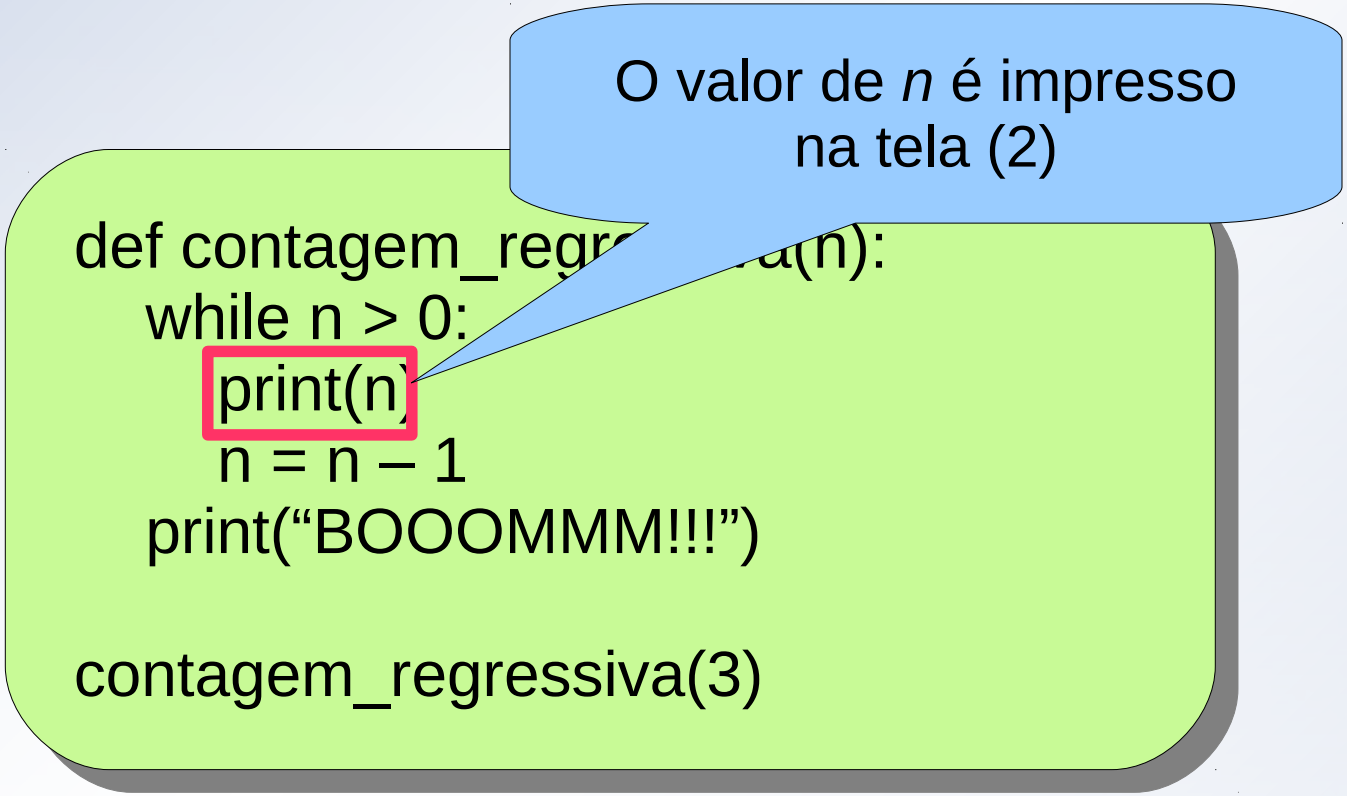
Repetições com while - Exemplo

A condição é novamente verificada e retorna *True* (2 > 0)

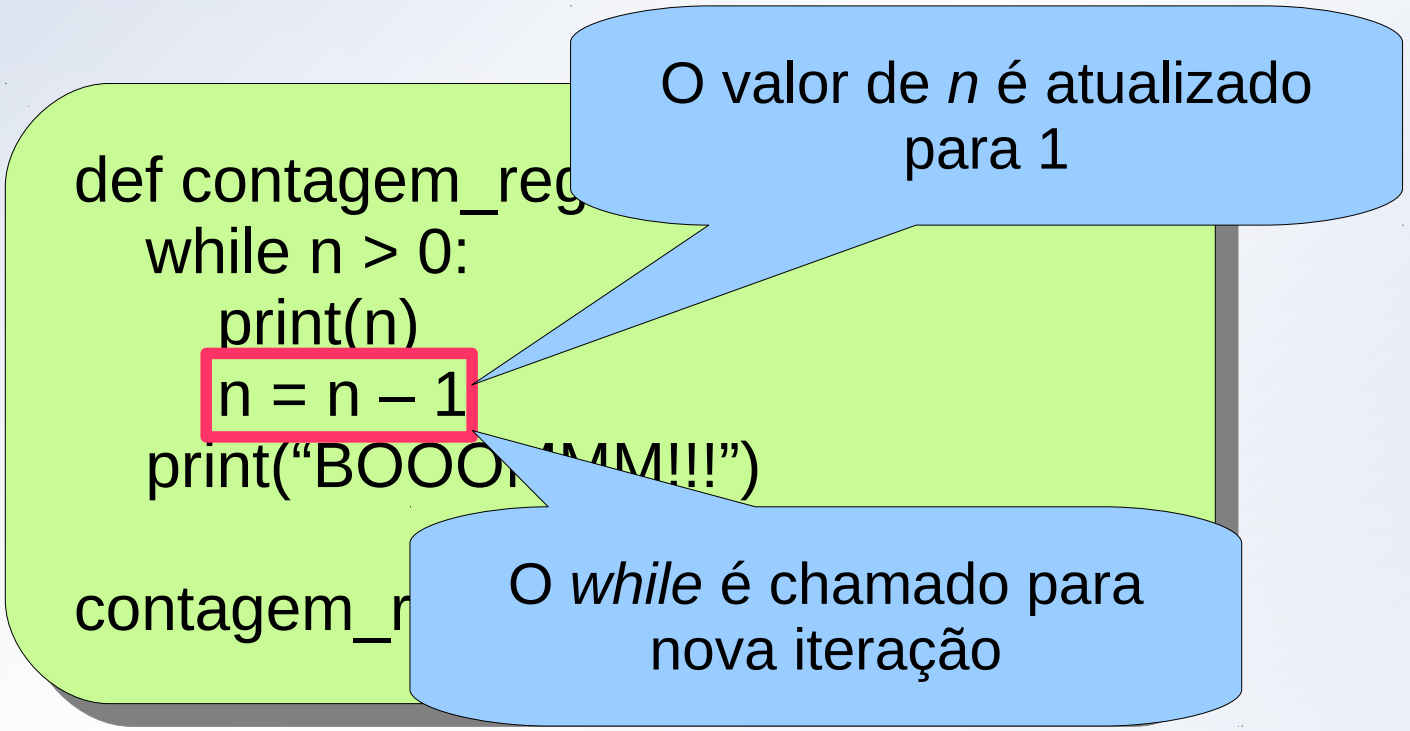
```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

Repetições com while - Exemplo



Repetições com while - Exemplo



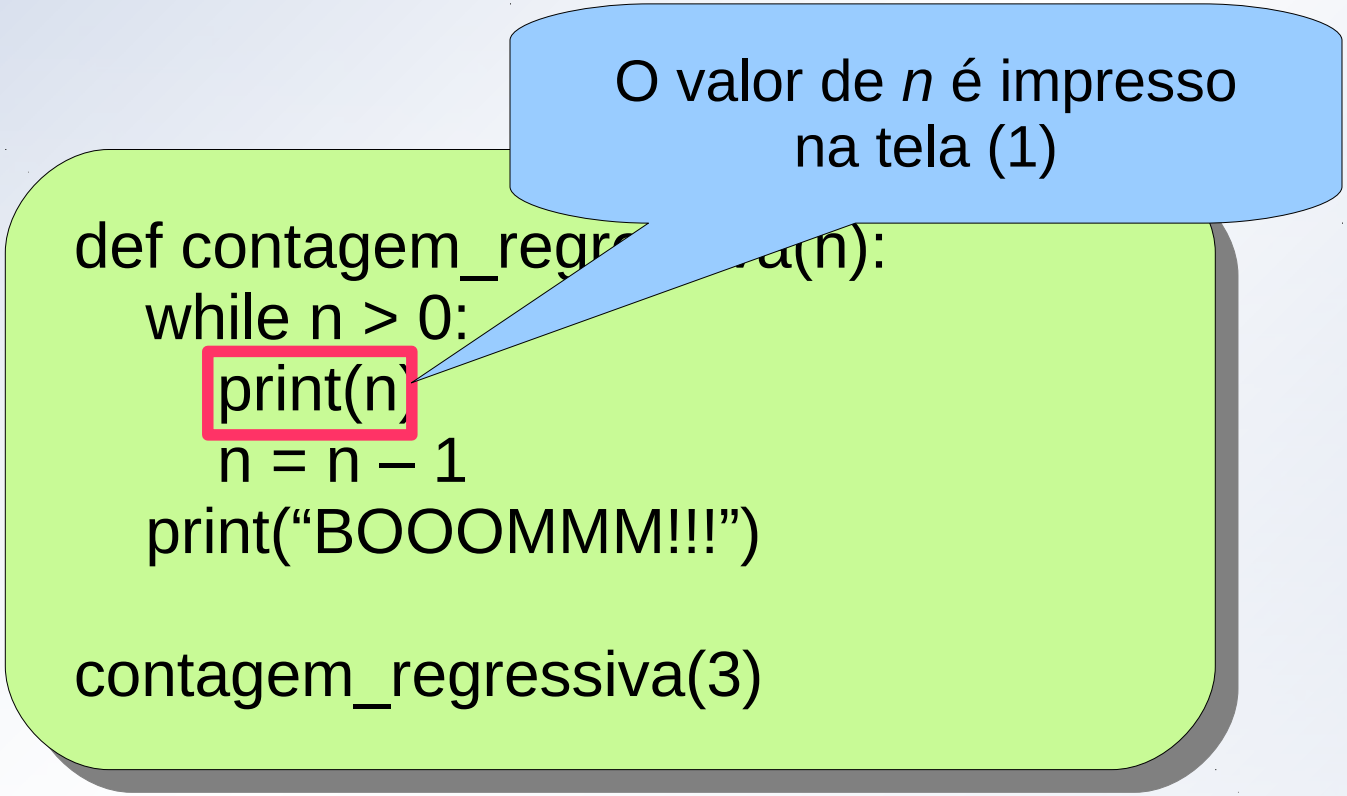
Repetições com while - Exemplo

A condição é novamente verificada e retorna *True* (1 > 0)

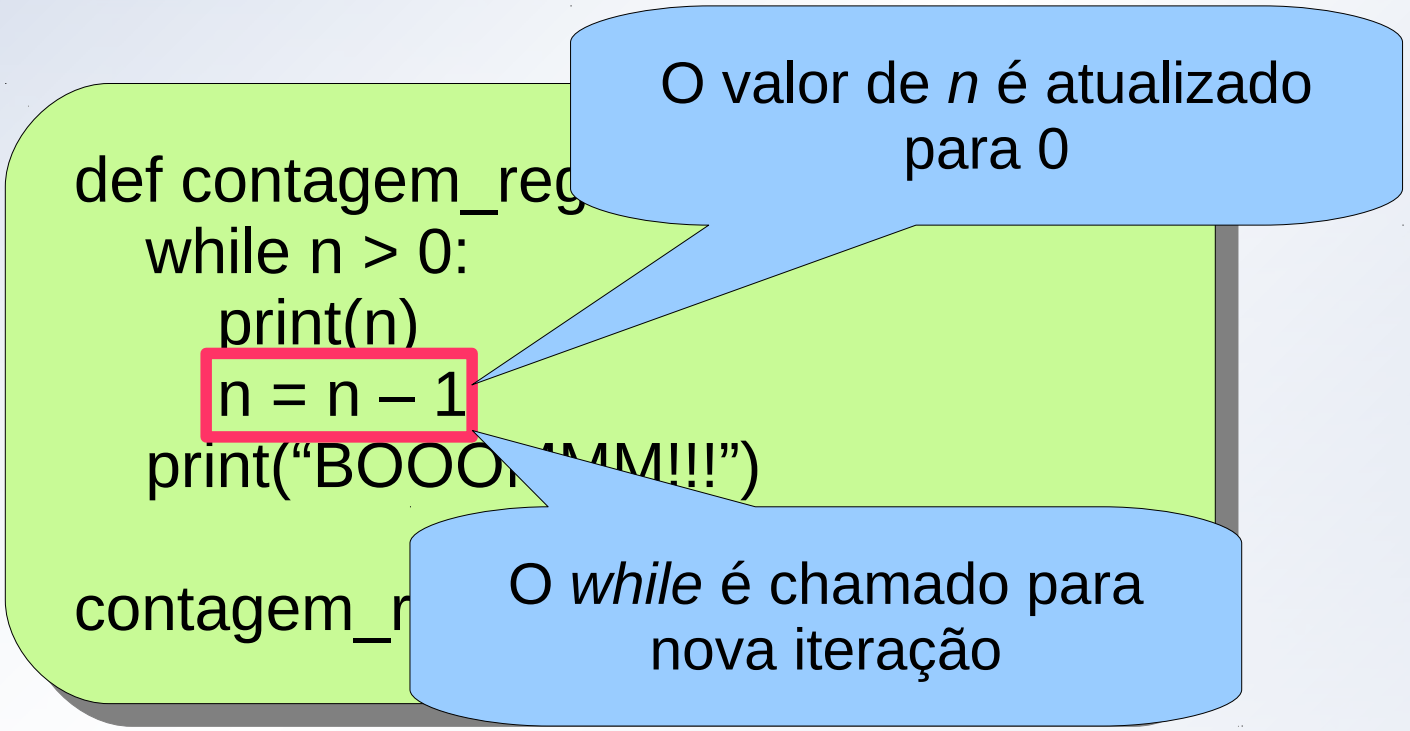
```
def contagem_regressiva(n):
    while n > 0:
        print(n)
        n = n – 1
    print(“BOOOMMM!!!”)

contagem_regressiva(3)
```

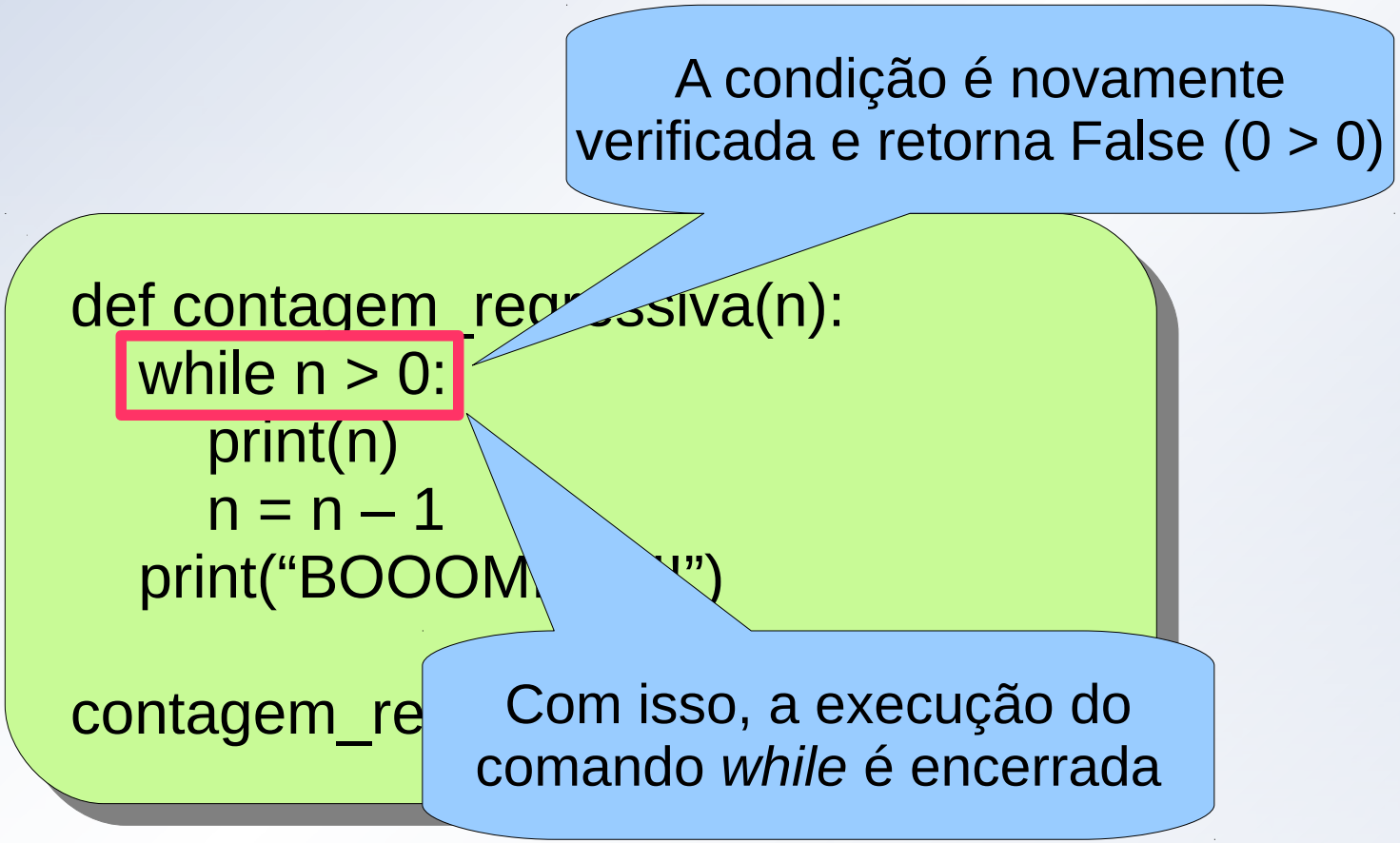
Repetições com while - Exemplo



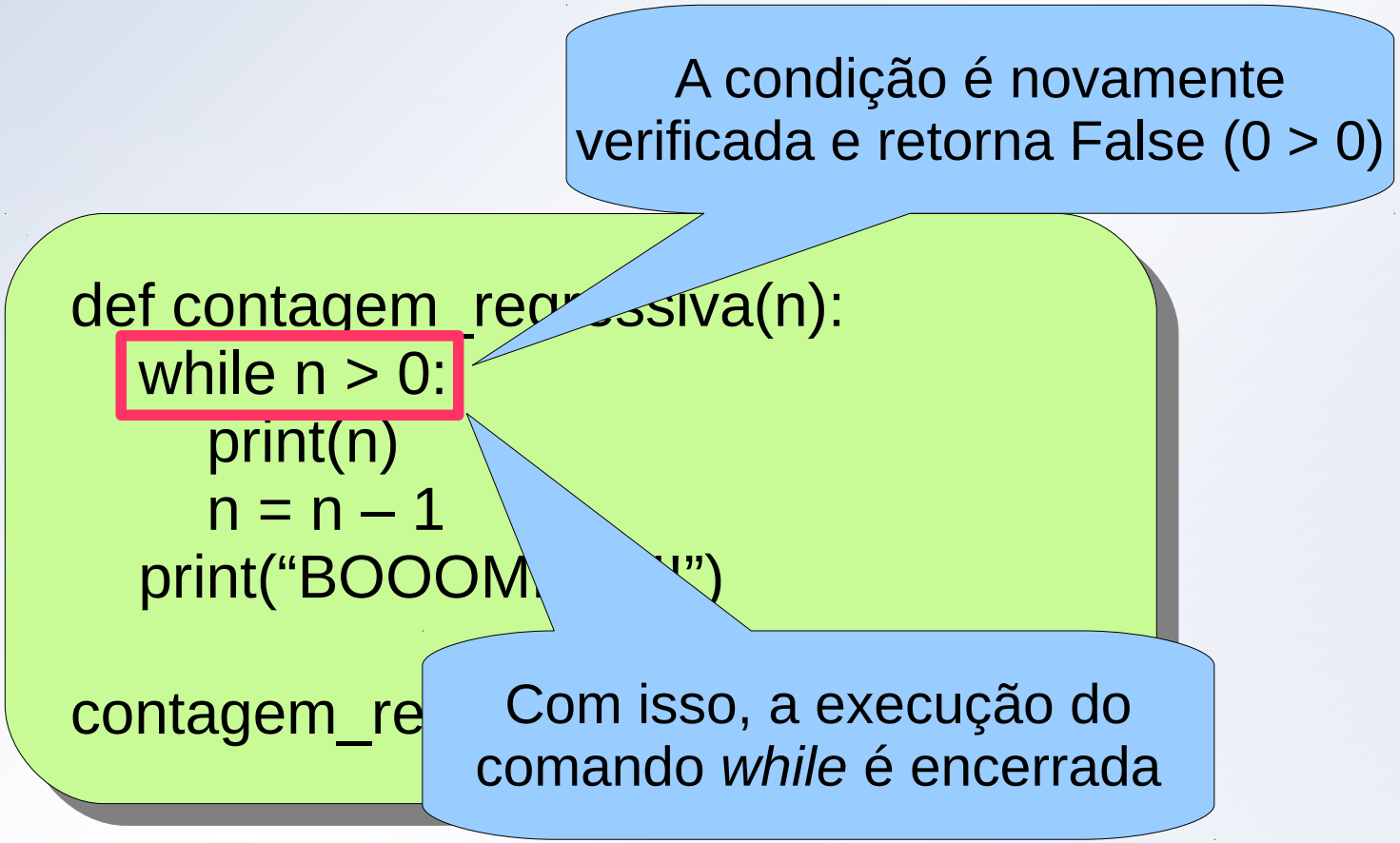
Repetições com while - Exemplo



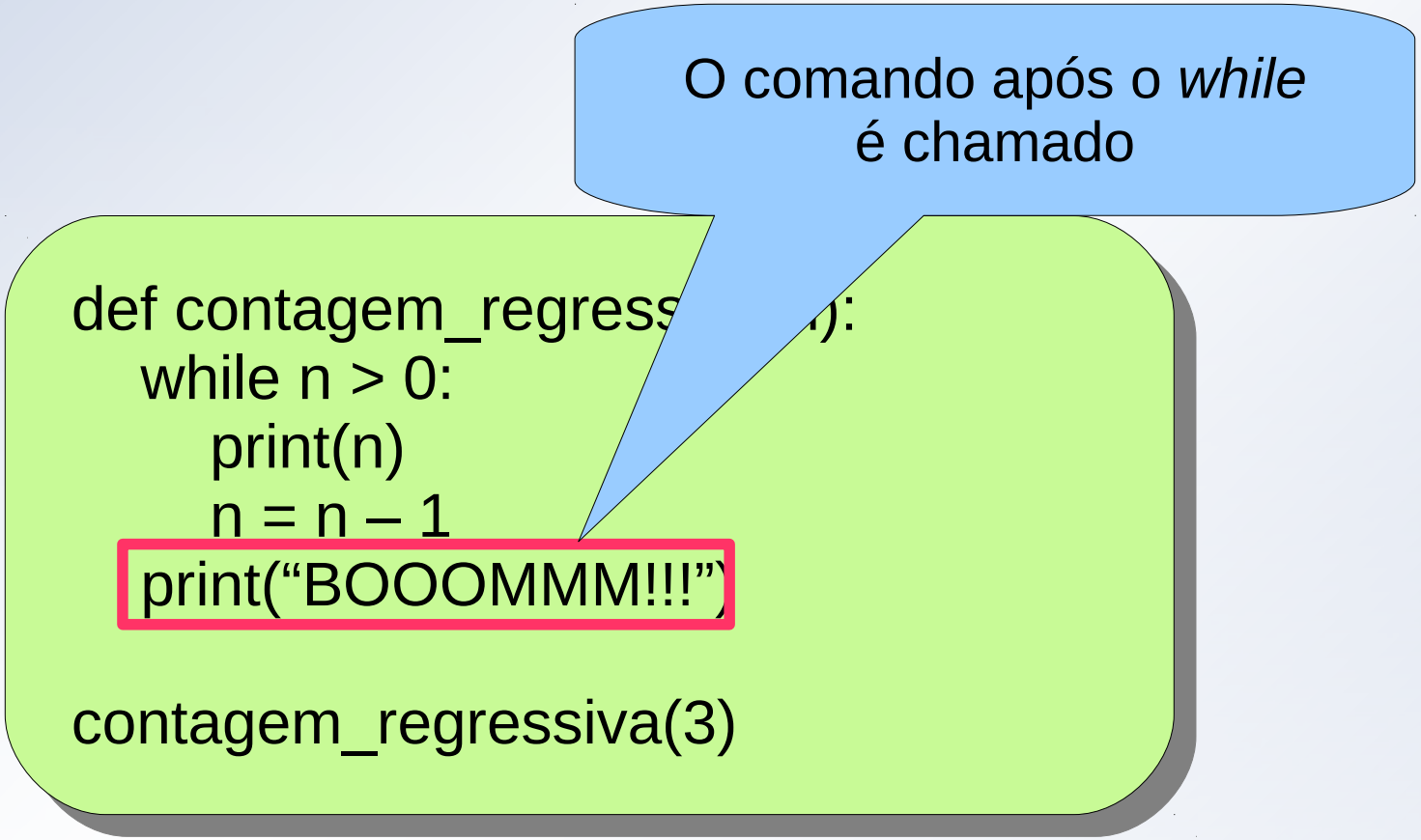
Repetições com while - Exemplo



Repetições com while - Exemplo



Repetições com while - Exemplo



Repetições com while - break

- Em algumas situações não se sabe quando as iterações da estrutura de repetição devem ser finalizadas
- Em tais situações, as condições do *while* dependem de instruções que estão no corpo do bloco
- Nestes casos, Python fornece a instrução *break* que ao ser chamada interrompe as repetições do *while*
- O Slide seguinte apresenta um programa em que as repetições do while são realizadas até que o usuário entre com o valor zero (0)

Repetições com while - break

```
while True:
    num = input(“Digite um número ou 0 para sair: “)
    if num == 0:
        break
    else:
        print(“O número digitado foi:”, num)
```

Repetições com while - break

Quando o usuário digitar 0, o *break* será chamado encerrando o *while*

```
while True:
    num = input("Digite um número ou 0 para sair: ")
    if num == 0:
        break
    else:
        print("O número digitado foi:", num)
```

Repetições com while - break

Quando o usuário digitar 0, o *break* será chamado encerrando o *while*

```
while True:
    num = input("Digite um número ou 0 para sair: ")
    if num == 0:
        break
    else:
        print("O número digitado foi:", num)
```

Quando qualquer outro número for digitado, este será exibido na tela

Repetições com while - break

```
while True:
    num = input(“Digite um número ou 0 para sair: “)
    if num == 0:
        break
    else:
        print(“O número digitado foi:”, num)
```

Como o código acima pode ser melhorado ?

Exercícios

1. Escreva um programa que calcule o fatorial usando *while.*
2. Escreva um programa que verifica os fatores de um número dado.
3. Re-escreva o programa de funções matemáticas permitindo que o usuário possa repetir operações até indicar o número 0 para sair.
4. Crie um programa que exiba na tela a tabuada de 1 a 9.
5. Escreva um programa que leia números inteiros e imprima "par" ou "impar", se os números forem, respectivamente pares ou ímpares. O programa deve parar ao ler um número menor que 0 (zero).