Computação 1 - Python
Aula 7 - Teórica: Estrutura de Repetição com teste de parada: while

Estrutura de Repetição *while*

Permite que o programador especifique que o programa deve repetir um conjunto de comandos **enquanto** uma dada condição for verdadeira.

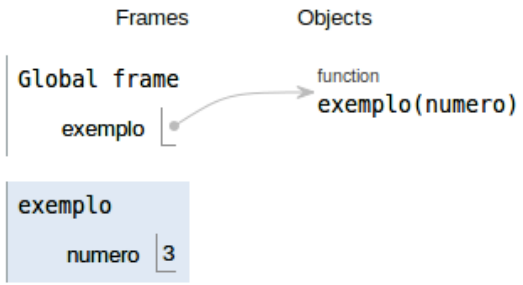
```
while condição:
    conjunto de comandos
```

Exemplo

```
# Função que reduz em 1 o valor do numero passado como
# parâmetro até chegar a zero.
# int → str
def exemplo(numero):
    while numero > 0:
        numero = numero - 1
    return "boom!"
```

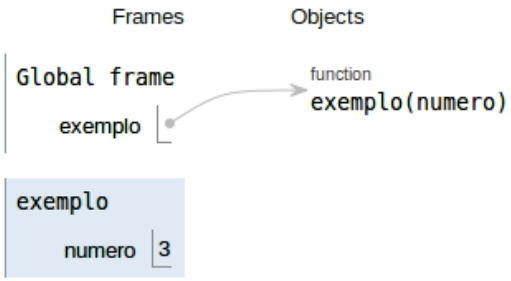
Estrutura de Repetição *while* – Python Tutor

```
1  # função que reduz em 1 o valor do numero passado
2  # parâmetro até chegar a zero.
3  # int -> str
4
5  def exemplo(numero):
6      while numero > 0:
7          numero = numero - 1
8      return "boom !"
9
10 exemplo(3)
```



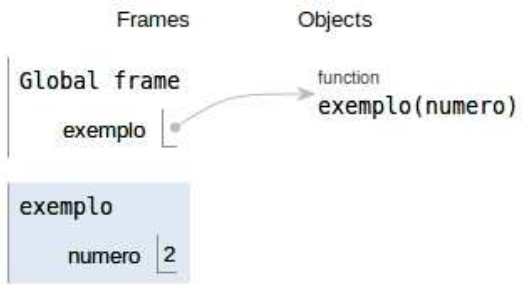
Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero > 0:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(3)
```



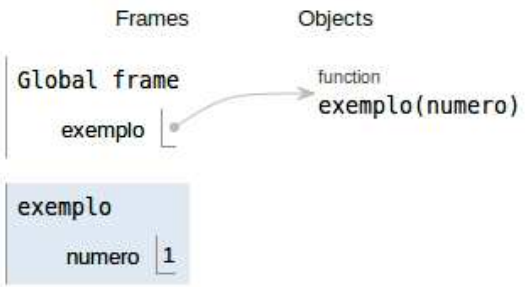
Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero > 0:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(3)
```



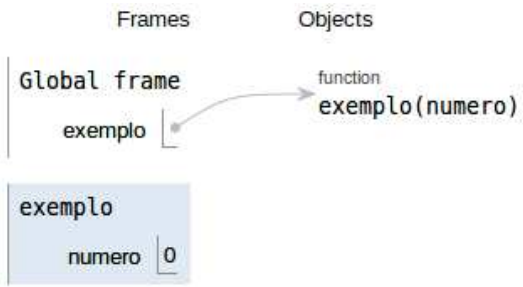
Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero > 0:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(3)
```



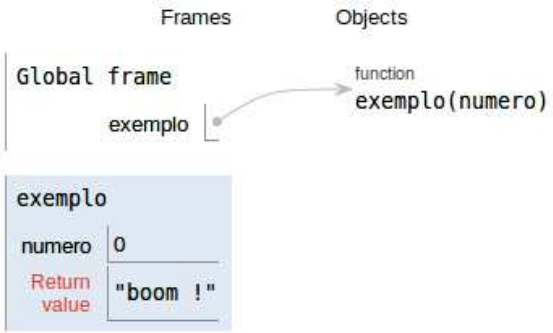
Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero > 0:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(3)
```



Estrutura de Repetição *while* – Python Tutor

```
 1 # função que reduz em 1 o valor do numero passado
 2 # parâmetro até chegar a zero.
 3 # int -> str
 4
 5 def exemplo(numero):
 6     while numero > 0:
 7         numero = numero - 1
 8     return "boom !"
 9
10 exemplo(3)
```



Estrutura de Repetição *while*

```
while condição:
    conjunto de comandos
```

- A condição é uma expressão ou dado do tipo booleano (**True** ou **False**), tal como os testes usados com o comando IF.
- Estrutura também conhecida como **laço de repetição** ou **"loop"**: o bloco de comandos é sequencialmente repetido tantas vezes quanto o teste da condição for verdadeiro.
- Somente quando a condição se torna falsa a próxima instrução após o bloco de comandos associado ao **while** é executada (fim do laço).

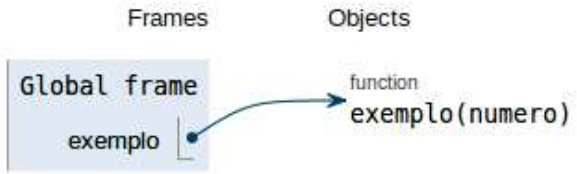
Estrutura de Repetição *while*

```
while condição:
    conjunto de comandos
```

- Se a condição da estrutura while já for falsa desde o início, o bloco de comandos associado a ela nunca é executado.
- Deve haver algum processo dentro do bloco de comandos que torne a condição falsa e a repetição seja encerrada, ou um erro GRAVE ocorrerá: sua função ficará rodando para sempre!!

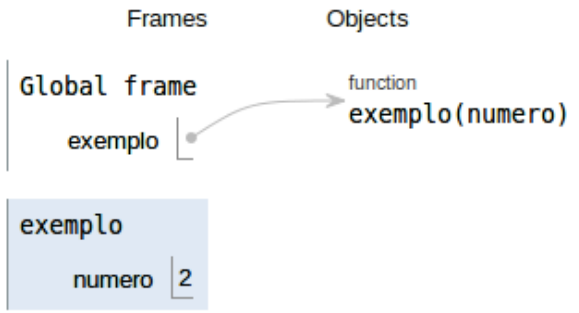
Estrutura de Repetição *while* – Python Tutor

```
1  # função que reduz em 1 o valor do numero passado
2  # parâmetro até chegar a zero.
3  # int -> str
4
5  def exemplo(numero):
6      while numero < 5:
7          numero = numero - 1
8      return "boom !"
9
10 exemplo(2)
```



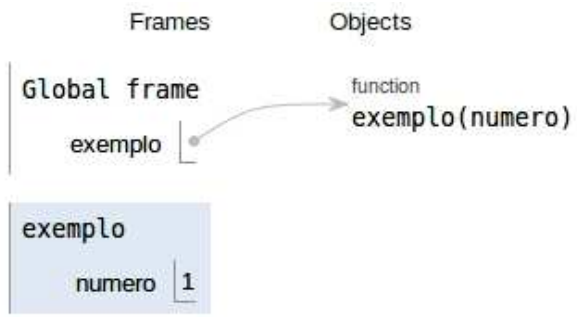
Estrutura de Repetição *while* – Python Tutor

```
1  # função que reduz em 1 o valor do numero passado
2  # parâmetro até chegar a zero.
3  # int -> str
4
5  def exemplo(numero):
6      while numero < 5:
7          numero = numero - 1
8      return "boom !"
9
10 exemplo(2)
```



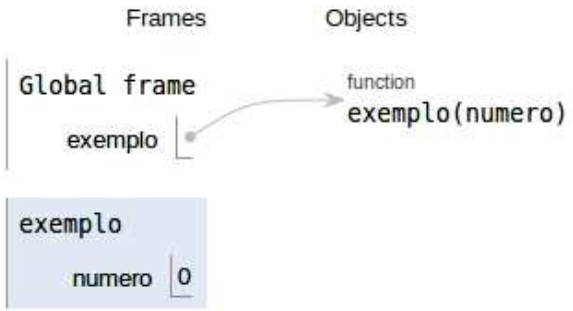
Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero < 5:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(2)
```



Estrutura de Repetição *while* – Python Tutor

```
 1  # função que reduz em 1 o valor do numero passado
 2  # parâmetro até chegar a zero.
 3  # int -> str
 4
 5  def exemplo(numero):
 6      while numero < 5:
 7          numero = numero - 1
 8      return "boom !"
 9
10  exemplo(2)
```



Estrutura de Repetição *while*

Exemplo

```
# Função que conta quantas vezes se pode reduzir em 1 o valor do número
# passado como parâmetro até chegar a zero.
# int → str
def exemplo1(numero):
    contador = 0 # variável contadora
    while numero > 0:
        numero = numero - 1
        contador = contador + 1
    return "O programa rodou " + str(contador) + "vezes."
```

Estrutura de Repetição *while*

Faça uma função que determina a soma de todos os números pares desde 100 até 200.

Estrutura de Repetição *while*

Faça uma função que determina a soma de todos os números pares desde 100 até 200.

```
# Função que calcula a soma dos números pares de 100 a 200
# sem entrada → int
def somaPares():
    soma = 0 # variável acumuladora
    contador = 100 # o contador não precisa começar de zero
    while contador < 200:
        soma = soma + contador
        contador = contador + 2 # o contador não precisa ir de 1 em 1
    return soma
```

Estrutura de Repetição *while*

A função abaixo apresenta algum problema?

```
# sem entrada → int
def exemplo3():
    x = 10
    while x > 8:
        x = x+ 2
    return x
```

Estrutura de Repetição *while*

A função abaixo apresenta algum problema?

```
# sem entrada → int
def exemplo3():
    x = 10
    while x > 8:
        x = x+ 2
    return x
```

- Sendo X igual a 10, o teste $X > 8$ é inicialmente verdadeiro.
- Enquanto a condição for verdadeira, apenas o comando $X = X + 2$ será executado. Porém incrementar a variável X não altera a validade da condição $X > 8$.
- Logo, a repetição segue indefinidamente! (Loop infinito)

Estrutura de Repetição *while*

O que faz a seguinte função ?

```
# int → int
def soma(numero):
    soma = 0
    contador = 0
    while contador < numero:
        soma = soma + contador
        contador = contador + 1
    return soma
```

Estrutura de Repetição *while*

Faça uma função que gere números aleatórios entre 1 e 10 e calcule a soma destes números até que seja gerado o número 5.
Use a função *randint(inicio,fim)* do módulo **random** para gerar um número aleatório, onde os valores de (início,fim) representam o intervalo desejado para os números a serem gerados.
Exemplo: randint(1,10) $\rightarrow$ gera um número aleatório entre 1 e 10, inclusive.

Estrutura de Repetição *while*

Faça uma função que gere números aleatórios entre 1 e 10 e calcule a soma destes números até que seja gerado o número 5.
Use a função *randint(inicio,fim)* do módulo **random** para gerar um número aleatório, onde os valores de (início,fim) representam o intervalo desejado para os números a serem gerados.
Exemplo: randint(1,10) $\rightarrow$ gera um número aleatório entre 1 e 10, inclusive.

```
from random import randint
# sem entrada → int

def somaAleatoria():
    soma = 0
    numero = randint(1,10)
    while numero != 5:
        soma = soma + numero
        numero = randint(1,10)
    return soma
```

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.

```
from random import randint
# sem entrada → int

def soma10():
    soma = 0
    contador = 0
    while contador < 10:
        numero = randint(1,5)
        soma = soma + numero
        contador = contador + 1
    return soma
```

Computação 1 - Python
Aula 7 - Teórica: Estrutura de Repetição com teste de parada: while

Computação 1 - Python
Aula 8 - Teórica: Estrutura de Repetição : for

Estrutura de Repetição *while*

Estrutura que permite a repetição de um conjunto de comandos. Até o momento vimos o *while*:

```
while condição:
    conjunto de comandos
```

- Com *while* podemos implementar qualquer algoritmo que envolva repetição.
- **DICA**: o *while* é mais recomendado quando não se sabe ao certo quantas vez a repetição será feita, pois a condição é um teste booleano qualquer e não necessariamente uma contagem.

Estrutura de Repetição *while*

Lembre: Faça uma função que gere números aleatórios entre 1 e 10 e calcule a soma destes números até que seja gerado o número 5.

```
from random import randint
# sem entrada → int

def somaAleatoria():
    soma = 0
    numero = randint(1,10)
    while numero != 5:
        soma = soma + numero
        numero = randint(1,10)
    return soma
```

O número de repetições dos comandos associados ao laço *while* depende de quando sair o número 5. Podem ser 2 vezes ou 1000 vezes!

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.
Como seria essa função com while?

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.
Como seria essa função com while?

```
from random import randint
# Função que soma 10 números gerados aleatoriamente
# no intervalo de 1 a 5
# sem entrada → int
def soma10():
    contador = 0
    soma = 0
    while contador < 10:
        numero = randint(1,5)
        soma = soma + numero
        contador = contador + 1
    return soma
```

O número de repetições será 10 em qualquer execução do programa, independente dos números aleatórios gerados.

Estrutura de Repetição *for*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.

```
from random import randint
# Função que soma 10 números gerados aleatoriamente
# no intervalo de 1 a 5 - usando for
# sem entrada → int
def soma10usandofor():
    soma = 0
    for contador in range(10):
        numero = randint(1,5)
        soma = soma + numero
    return soma
```

```
for var in range(n):
    comandos
```

OU

```
for var in [0,..., n-1]:
    comandos
```

No programa acima, a variável *contador* vai assumir os valores 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9.

Estrutura de Repetição

IMPORTANTE: diferença de uso entre *while* e *for*:

- **While**: decisão sobre repetir ou não baseia-se em teste booleano. Risco de loop infinito. :-(
- **For**: Contagem automática do número de repetições.

Estrutura de Repetição *for*

- A função range(...) pode ter 1, 2 ou 3 argumentos:
  - range(numero): faz com que a variável do **for** assuma valores de 0 a numero-1

    for x in range(10) → x recebe 0,1,2,...,9
  - range(inf,sup): faz com que a variável do **for** assuma valores de inf a sup-1

    for x in range(3,8) → x recebe 3,4,5,6,7
  - range(inf, sup, inc): faz com que a variável do **for** assuma valores de inf a sup-1 com incremento de inc

    for x in range(3,8,2) → x recebe 3,5,7

Estrutura de Repetição *for*

Faça um programa que determina a soma de todos os números pares desde 100 até 200. (Usando **for** ao invés de **while**)

Estrutura de Repetição *for*

Faça um programa que determina a soma de todos os números pares desde 100 até 200. (Usando **for** ao invés de **while**)

```
# Função que soma todos os números pares
# de 100 até 200
# sem entrada -> int
def somaPares():
      Soma = 0
      for Par in range(100,202,2) :
          Soma = Soma + Par
      return Soma
```

Computação 1 - Python
Aula 8 - Teórica: Estrutura de Repetição : for