Computação I - Python
Aula 7 - Teórica: Estrutura de Repetição com Teste de Parada: while

João C. P. da Silva

Carla A. D. M. Delgado

Ana Luisa Duboc

Dept. Ciência da Computação - UFRJ

Estrutura de Repetição *while*

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

```
while <condição>:
    <sequência de comandos>
```

Exemplo

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0$

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0$

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True
  - listax=['x','x']

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True
  - listax=['x','x']
  - numero = 0

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True
  - listax=['x','x']
  - numero = 0
- $0 > 0$

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True
  - listax=['x','x']
  - numero = 0
- $0 > 0 \Rightarrow$ False

Estrutura de Repetição *while*

```
1  def exemplo1(numero):
2   """Funcao que cria uma lista formada por strings 'x'. A quantidade de strings 'x' e
        igual a numero.
3    Parametro de entrada: int
4    Valor de Retorno: list"""
5
6    listax = []
7    while numero > 0:
8      list.append(listax,'x')
9      numero = numero - 1
10   return listax
```

- exemplo1(2)
- listax=[]
- $2 > 0 \Rightarrow$ True
  - listax=['x']
  - numero = 1
- $1 > 0 \Rightarrow$ True
  - listax=['x','x']
  - numero = 0
- $0 > 0 \Rightarrow$ False

return ['x','x']

Estrutura de Repetição *while*

```
while <condição>:
    <sequência 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>:
    <sequência de comandos>
```

- Se a <condição> da estrutura while já for falsa desde o início, o bloco de <sequência de comandos> associado a ela nunca é executado.
- Deve haver algum processo dentro do bloco de <sequência 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*

```
1  def exemplowhile0(numero):
2
3  """Parametro de entrada: int
4  Valor de Retorno: list"""
5
6    listanum = [ ]
7    while numero > 0:
8      listanum[numero-1] = numero
9      numero = numero -1
10   return listanum
```

Qual o problema desta função?

Estrutura de Repetição *while*

```
1  def exemplowhile0(numero):
2
3  """Parametro de entrada: int
4  Valor de Retorno: list"""
5
6    listanum = [ ]
7    while numero > 0:
8      listanum[numero-1] = numero
9      numero = numero -1
10   return listanum
```

Qual o problema desta função?

```
1  def exemplowhile1(numero):
2
3  """Parametro de entrada: int
4  Valor de Retorno: list"""
5
6    listanum = numero * [0]
7    while numero > 0:
8      listanum[numero-1] = numero
9      numero = numero -1
10   return listanum
```

Estrutura de Repetição *while*

```
1  def exemplowhile0(numero):
2
3  """Parametro de entrada: int
4  Valor de Retorno: list"""
5
6    listanum = [ ]
7    while numero > 0:
8      listanum[numero-1] = numero
9      numero = numero -1
10   return listanum
```

Qual o problema desta função?

```
1  def exemplowhile2(numero):
2
3  """Parametro de entrada: int
4  Valor de Retorno: list"""
5
6    listanum = [ ]
7    while numero > 0:
8      list.insert(listanum,0,numero)
9      numero = numero -1
10   return listanum
```

Estrutura de Repetição *while*

Exemplo

```
1  def exemplo2(numero):
2
3  """Funcao que conta quantas vezes se pode reduzir em 1 o valor do numero passado como
       parametro ate chegar a zero.
4  Parametro de entrada: int
5  Valor de retorno: str"""
6
7    contador = 0 # variavel contadora
8    while numero > 0:
9      numero = numero - 1
10     contador = contador + 1
11   return "A funcao 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.

```
1  def somaPares():
2
3  """Funcao que calcula a soma dos numeros pares de 100 a 200
4  Parametro de entrada: nao tem
5  Valor de retorno: int"""
6
7    soma = 0 # variavel acumuladora
8    contador = 100 # o contador nao precisa comecar de zero
9    while contador <= 200:
10     soma = soma + contador
11     contador = contador + 2 # o contador nao precisa ir de 1 em 1
12   return soma
```

Estrutura de Repetição *while*

A função abaixo apresenta algum problema?

```
1  def exemplo3():
2
3  """Parametro de entrada: nao tem
4  Valor de retorno: int"""
5
6    x = 10
7    while x > 8:
8      x = x+ 2
9    return x
```

Estrutura de Repetição *while*

A função abaixo apresenta algum problema?

```
1 def exemplo3():
2
3 """Parametro de entrada: nao tem
4 Valor de retorno: int"""
5
6   x = 10
7   while x > 8:
8     x = x+ 2
9   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 ?

```
1  def soma(numero):
2
3  """Parametro de entrada: int
4  Valor de retorno: int"""
5
6    soma = 0
7    contador = 0
8    while contador < numero:
9      soma = soma + contador
10     contador = contador + 1
11   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.

```
1  from random import randint
2
3  def somaAleatoria():
4
5  """Parametro de entrada: nao tem
6  Valor de retorno: int"""
7
8    soma = 0
9    numero = randint(1,10)
10   while numero != 5:
11     soma = soma + numero
12     numero = randint(1,10)
13   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.

Solução usando listas.

```
1  from random import randint
2
3  def somaAleatoria():
4
5  """Parametro de entrada: nao tem
6  Valor de retorno: int"""
7
8    lista = []
9    numero = randint(1,10)
10   while numero != 5:
11     list.append(lista, numero)
12     numero = randint(1,10)
13   return sum(lista)
```

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.

```
1  from random import randint
2
3  def soma10():
4
5  """Parametro de entrada: nao tem
6  Valor de retorno: int"""
7
8    soma = 0
9    contador = 0
10   while contador < 10:
11     numero = randint(1,5)
12     soma = soma + numero
13     contador = contador + 1
14   return soma
```

Estrutura de Repetição *while*

Para cada um dos itens abaixo, faça uma tabela mostrando os valores que i, j e n assumem depois de cada execução do laço *while*.

```
def ...
    i = 0
    j = 10
    n = 0
    while i < j:
        i = i + 1
        j = j - 1
        n = n + 1
```

```
def ...
    i = 0
    j = 0
    n = 0
    while i < 10:
        i = i + 1
        n = n + i + j
        j = j + 1
```

Estrutura de Repetição *while*

Faça uma função que dada uma lista de tamanho desconhecido contendo as notas de uma turma de alunos, retorne a média dessas notas.

Autores

- João C. P. da Silva ▸ Lattes
- Carla Delgado ▸ Lattes
- Ana Luisa Duboc ▸ Lattes

Colaboradores

- Anamaria Martins Moreira ▸ Lattes
- Fabio Mascarenhas ▸ Lattes
- Leonardo de Oliveira Carvalho ▸ Lattes
- Charles Figueiredo de Barros ▸ Lattes
- Fabrício Firmino de Faria ▸ Lattes

Computação I - Python
Aula 7 - Teórica: Estrutura de Repetição com Teste de Parada: while

João C. P. da Silva

Carla A. D. M. Delgado

Ana Luisa Duboc

Dept. Ciência da Computação - UFRJ