Python - Comandos de Repetição

Prof. André Backes

Repetição por Condição

- Um conjunto de comandos de um algoritmo pode ser repetido quando subordinado a uma condição:

enquanto *condição* faça
  comandos;
fim enquanto

Repetição por Condição

- De acordo com a condição, os comandos serão repetidos
  - zero vezes (condição falsa)
  - ou mais vezes (enquanto a condição for verdadeira)

- Essa estrutura normalmente é denominada ***laço*** ou ***loop***.

Repetição por Condição

- Condição
  - qualquer expressão que resulte em um valor do tipo lógico e pode envolver operadores aritméticos, lógicos, relacionais e resultados de funções.
  - Ex:

    x > 5

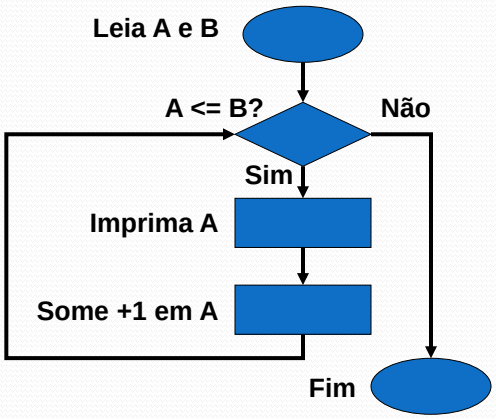
    (N < 60) and (N > 35)

Funcionamento

- A condição da cláusula ***enquanto*** é testada.
  - Se ela for verdadeira os comandos seguintes são executados em sequência como em qualquer algoritmo, até a cláusula ***fim enquanto***.
  - O fluxo nesse ponto é desviado de volta para a cláusula ***enquanto*** e o processo se repete.
  - Se a condição for falsa (ou quando se tornar falsa), o fluxo do algoritmo é desviado para o primeiro comando após a cláusula ***fim enquanto***.

Repetição por Condição

- Relembrando fluxogramas
  - Um processo pode ser repetido até atender ou não uma condição.



Loop Infinito

- Um **loop** ou **laço** infinito ocorre quando cometemos algum erro
  - ao especificar a condição lógica que controla a repetição
  - ou por esquecer de algum comando dentro da repetição

Loop Infinito

Exemplo: loop infinito (condição errônea)

```
X recebe 4;
enquanto (X < 5) faca
    X recebe X - 1;
    Imprima X;
fim enquanto
```

Exemplo: loop infinito (não muda valor)

```
X recebe 4;
enquanto (X < 5) faca
   Imprima X;
fim enquanto
```

Exercício

- Escreva, em pseudo-código, o algoritmo para calcular a média de N números

Exercício

```
Leia n;
media recebe 0;
n1 recebe 0;
Enquanto (n1 < n)
   Leia x;
   media recebe media + x;
   n1 recebe n1 + 1;
Fim enquanto
Imprima media/n;
```

Comando while

- Equivale ao comando "**enquanto**" utilizado nos pseudo-códigos
  - Repete uma sequência de comandos enquanto a condição for verdadeira

- Forma geral:

```
while condição:
    instrução 1
    instrução 2
    ...
    instrução n

continuação do programa
```

Exemplo while

- Imprimindo os números entre A e B

Solução

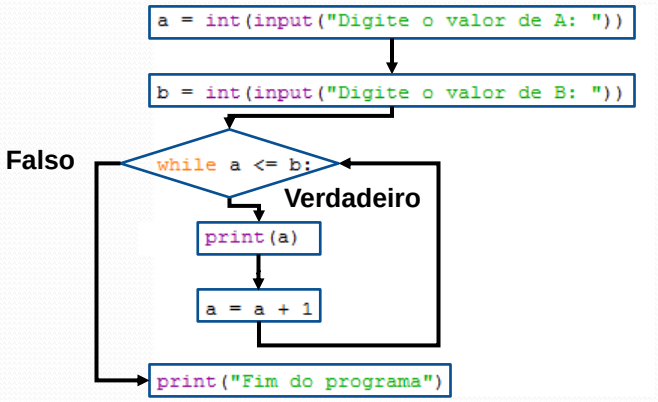
```
a = int(input("Digite o valor de A: "))
b = int(input("Digite o valor de B: "))
while a <= b:
    print(a)
    a = a + 1

print("Fim do programa")
```

Saída

```
Digite o valor de A: 1
Digite o valor de B: 7
1
2
3
4
5
6
7
Fim do programa
```

Exemplo while



Exercício

- Escreva, usando while, o algoritmo para calcular a média de N números

Exercício

Solução

```
N = int(input("Digite o valor de N: "))
media = 0
total = 0
while total < N:
    x = int(input("Digite o valor de x: "))
    media = media + x
    total = total + 1
media = media / N
print("media = ",media)
```

Saída

```
>>>
Digite o valor de N: 5
Digite o valor de x: 3
Digite o valor de x: 6
Digite o valor de x: 8
Digite o valor de x: 2
Digite o valor de x: 1
media =  4.0
```

Comando for

- O loop ou laço **for** é usado para repetir um conjunto de instruções para uma lista de valores
  - O número de iterações da repetição está limitado pelo comprimento da lista de valores

- Forma geral

```
for variável in lista-de-valores:
    instrução 1
    instrução 2
    ...
    instrução n

continuação do programa
```

Comando for

- Exemplo

```
import math

for x in [0,1,2,3,4,5]:
    print("A raiz de",x, "é igual a", math.sqrt(x))
```

- Saída

```
>>>
A raiz de 0 é igual a 0.0
A raiz de 1 é igual a 1.0
A raiz de 2 é igual a 1.4142135623730951
A raiz de 3 é igual a 1.7320508075688772
A raiz de 4 é igual a 2.0
A raiz de 5 é igual a 2.23606797749979
>>>
```

Comando for

- Exemplo

```
compras = ["Miojo","Ovo","Leite","Pão"]
print("Lista de compras")
for item in compras:
    print("Produto: ",item)
```

- Saída

```
>>>
Lista de compras
Produto:  Miojo
Produto:  Ovo
Produto:  Leite
Produto:  Pão
>>>
```

Função range

- A função **range()** permite gerar sequências de valores em progressão aritmética
  - Muito útil para gerar as listas de valores para o comando **for**

- Formas de uso
  - range(N): gera valores inteiros de 0 até N-1
  - range(I,N) : gera valores inteiros de I até N-1
  - range(I,N,D) : gera os valores inteiros I, I+D, I+2D, ... inferiores a N.

Função range

- Usando a função **range()** junto com o comando **for**

```
#Gerar valores: 0, 1, 2, 3, 4
for x in range(5):
    print("Valor = ", x)

#Gerar valores: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
for x in range(10):
    print("Valor = ", x)

#Gerar valores: 3, 4, 5, 6, 7, 8, 9
for x in range(3,10):
    print("Valor = ", x)

#Gerar valores: 3, 5, 7, 9
for x in range(3,10,2):
    print("Valor = ", x)
```

Exemplo for

- Imprimindo os números entre A e B

Comando while

```
a = int(input("Digite o valor de A: "))
b = int(input("Digite o valor de B: "))
while a <= b:
    print(a)
    a = a + 1

print("Fim do programa")
```

Comando for

```
a = int(input("Digite o valor de A:"))
b = int(input("Digite o valor de B:"))
for x in range(a,b+1):
    print(x)
```

Comando for

- Diferença entre **for** e **while**
  - Comando **while**
    - Repete uma sequência de comandos enquanto uma condição for verdadeira
  - Comando **for**
    - Repete uma sequência de comandos "N vezes" ou "para N valores"
- Atenção
  - Podemos sempre re-escrever um comando **for** com **while**
  - Nem sempre podemos re-escrever um comando **while** como um **for**

Exercício

- Escreva, um programa para calcular o fatorial de um número N. Tente fazer usando **for** e **while**.

Exercício

Comando while

```
N = int(input("Digite o valor de N: "))
fat = 1
i = 1
while i <= N:
    fat = fat * i
    i = i + 1
print("O fatorial de ",N," é",fat)
```

Comando for

```
N = int(input("Digite o valor de N: "))
fat = 1
for i in range(1,N+1):
    fat = fat * i
print("O fatorial de ",N," é",fat)
```

Comando break

- O comando **break** serve para quebrar a execução de um comando de repetição (**for** ou **while**)
  - O **break** faz com que a execução do programa continue na primeira linha seguinte ao loop ou bloco de comandos que está sendo interrompido

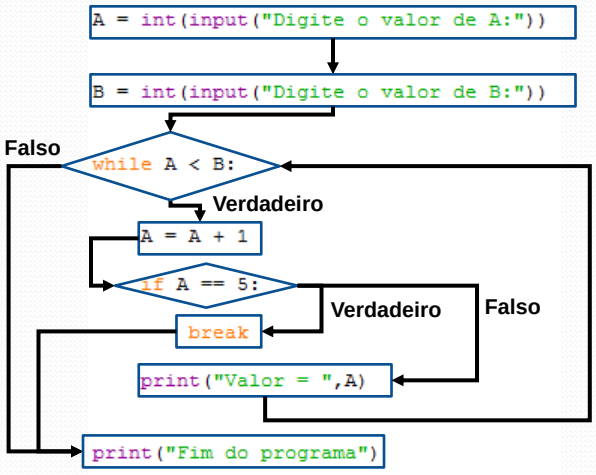
Comando break

```
A = int(input("Digite o valor de A:"))
B = int(input("Digite o valor de B:"))
while A < B:
    A = A + 1
    if A == 5:
        break
    print("Valor = ",A)

print("Fim do programa")
```

```
>>>
Digite o valor de A:1
Digite o valor de B:10
Valor =  2
Valor =  3
Valor =  4
Fim do programa
```

Comando break



Comando continue

- O comando **continue** serve para interromper apenas a iteração atual de um comando de repetição (**for** ou **while**)
  - Pula essa iteração do loop
  - Os comandos que sucedem o comando **continue** no bloco não são executados

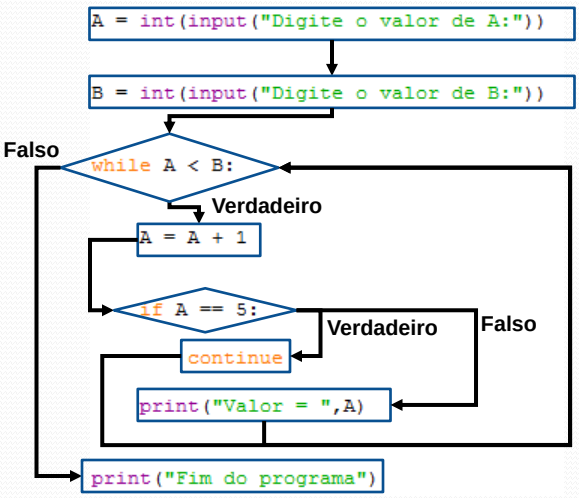
Comando continue

```
A = int(input("Digite o valor de A:"))
B = int(input("Digite o valor de B:"))
while A < B:
    A = A + 1
    if A == 5:
        continue
    print("Valor = ",A)

print("Fim do programa")
```

```
>>>
Digite o valor de A:1
Digite o valor de B:10
Valor =  2
Valor =  3
Valor =  4
Valor =  6
Valor =  7
Valor =  8
Valor =  9
Valor =  10
Fim do programa
```

Comando continue



Material Complementar

- Vídeo Aulas
  - Aula 13 - Comando while
  - https://youtu.be/LyHexIGdT-E
  - Aula 14 - Comando for
  - https://youtu.be/A9lJCksMaYE
  - Aula 15 - Comandos break e continue
  - https://youtu.be/1WtHazoPn70