s06-estruturas-de-repetição

April 13, 2021

1 ESTRUTURAS DE REPETIÇÃO

1.0.1 Semana 06

Uma estrutura de repetição permite que um ou mais comandos sejam executados repetidamente.

Neste caderno, estudamos uma estrutura de repetição chamada `for`. Como vamos ver, estruturar o programa para utilizar repetições permite com que problemas complexos possam ser escritos com poucas linhas.

1.1 O laço `for`

Vamos começar com um exemplo simples.

O programa a seguir imprime a saudação `olá mundo` 10 vezes na tela. Preste atenção que o comando que será repetido está identado (com espaçamento) em relação ao `for`.

[2]:
```
for i in range(10):
    print("olá mundo")
```

```
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
```

Vamos ver o que acontece quando colocamos dois comandos para serem repetidos na sequência

[5]:
```
for i in range(10):
    print("olá mundo")
    print("hello world")
```

```
olá mundo
hello world
olá mundo
hello world
```

```
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
olá mundo
hello world
```

A execução anterior resultou nas mensagens intercaladas. No entanto, uma outra possibilidade seria estruturar o programa como a seguir

[6]:
```
for i in range(10):
    print("olá mundo")
for i in range(10):
    print("hello world")
```

```
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
olá mundo
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
```

No caso anterior, o comando `print("olá mundo")` é repetido 10 vezes. Em seguida, um novo `for` repete 10 vezes o comando `print("hello world")`

Uso do passo.

[9]:
```
for i in range(0, 10, 2):
    print(i)
```

```
0
2
4
6
8
```

1.1.1 Sintaxe

A sintaxe da estrutura de repetição `for`

```
for <variável> in <sequência>:
  <comando 1>
  <comando 2>
  ...
  <comando n>
```

1.1.2 A função `range()`

A função `range()` cria um intervalo.

Existem 3 formas de utilizá-la.

Forma 1 Nesta forma, a linguagem cria o intervalo semi-aberto $[0, stop[$

```
range(stop)
```

Forma 2 Nesta forma, a linguagem cria o intervalo semi-aberto $[start, stop[$

```
range(start, stop)
```

Forma 3 Nesta forma, a linguagem cria o intervalo semi-aberto $[start, stop[$ com passo 2

```
range(start, stop, step)
```

Visualização dos intervalos Para ver os elementos que a função `range()` irá iterar, podemos utilizar um conversor para `list()`.

```
list(x)      # Convert x para uma lista
```

Exemplos.

```
list( range(4) )
   => [0, 1, 2, 3]

list( range(2, 4) )
   => [2, 3]

list( range(2, 10, 3) )
   => [2, 5, 8]

list( range(5, 2, -1))
   => [5, 4, 3]
```

[12]:
```
list( range(2, 10) )
```

[12]:
```
[2, 3, 4, 5, 6, 7, 8, 9]
```

Note que quando o passo é `-1`, a sequência ainda *para antes* do final (*stop*) fornecido

1.1.3 Iterando sequências

O comando `for` permite a iteraração de **sequências**. Uma **sequência** em Python é um conjunto de elementos ordenados. A forma mais simples da sequência é uma **lista**, onde os elementos ficam entre colchetes.

Note que a variável `i` vai assumir o valor de cada elemento da lista `[8,3,0]` em cada iteração.

[14]:
```
for i in [8, 3, 0]:
    print(i, i ** 2)
```

```
8 64
3 9
0 0
```

Veja que a forma a seguir é equivalente ao código anterior.

[15]:
```
lista = [8, 3, 0]
for i in lista:
    print(i, i ** 2)
```

```
8 64
3 9
0 0
```

1.1.4 Loops de acumulação

Em alguns casos, é necessário utilizar loops de acumulação. Considere por exemplo a somatória.

$$\sum_{k=0}^{10} k$$

[21]:
```
k = 0
for i in range(11):
    k = k + i

print("O valor final do acumulador eh: ", k)
```

```
O valor final do acumulador eh:  55
```

O caso do produtório é bem similar.

$$\prod_{i=1}^{10} k$$

[20]:
```
k = 1
for i in range(1,11):
    k = k * i

print("O valor final do acumulador de multiplicacao eh: ", k)
```

```
O valor final do acumulador de multiplicacao eh:  3628800
```

Exercício de fixação

Por que na somatória a variável `k` é inicializada com 0, e no produtório a variável `k` é inicializada com 1?

Sintaxe dos acumuladores

```
<acumulador> = <valor-inicial>
loop:
     <acumulador> = <acumulador> <operação> <quantidade>
```

1.1.5 A soma harmônica

A **soma harmônica** pode ser definida da seguinte forma.

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots \frac{1}{n} = \sum_{k=1}^{n} \frac{1}{k}$$

[22]:
```
def harmonica(n):
    # Computa a soma de 1/k para k=1 a n
    total = 0
    for k in range(1, n + 1):
        total = total + 1 / k
    return total

def main():
    n = int(input('Entre com um inteiro positivo: '))
    print("A soma de 1/k para k = 1 a", n, "eh", harmonica(n))

main()
```

```
Entre com um inteiro positivo: 3
A soma de 1/k para k = 1 a 3 eh 1.8333333333333333
```

Atalhos de atribuição

```
x += y
x -= y
x *= y
x /= y
```

1.2 Exercícios

[1] No programa da soma harmônica, explique o uso de `n + 1` ao invés de `n` no `range(n + 1)`.

[2] Liste todas as variáveis usadas no programa da soma harmônica e descreva o escopo de cada.

[3] Determine os elementos que serão iterados para cada uma dessas funções `range`. Dica: utilize a função `list()` para visualizar o retorno de cada chamada de função ao `range()`

(a) `range(10)`

(b) `range(5, 10)`

(c) `range(10, 5)`

(d) `range(3, 10, 2)`

(e) `range(10, 0, -1)`

(f) `range(10, 0, -2)`

(g) `range(0, 10, -1)`

(h) `range(0, 1, 0.1)`

[4] Escreva um programa que imprime uma citação de sua escolha exatamente 1000 vezes.

[5] Escreva um programa que imprime uma tabela de valores de n e 2^n para $n = 1, 2, ..., 10$. Estes valores parecem familiares?

[6] Escreva um programa que imprime uma tabela de valores de n, $\log n$, $n \log n$, n^2, e 2^n para $n = 10, 20, ..., 200$. O `log` é uma função do módulo `math`. A função $n \log n$ cresce mais que n ou n^2? Como você descreveria o crescimento de 2^n? E de $\log n$?

Dica: utilize a função `log` na biblioteca `math`.

[7] Escreva um programa que imprime uma tabela para as áreas dos círculos de raio $r = 1, 2, ..., 10$.

[8] Escreva um programa que, dado um número n, calcule a soma dos n pares utilizando o laço `for`. Não considere o 0 como par.