

# Variáveis e instruções de repetição

---

Marco A L Barbosa  
malbarbo.pro.br

Departamento de Informática  
Universidade Estadual de Maringá



# Conteúdo

Introdução

Variáveis

Instruções de repetição

Exemplos

Atividades

# Introdução

O valor  $e^x$ , onde  $e$  é o número de Euler e  $x$  um valor real qualquer, pode ser calculado pela seguinte série de Taylor

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$$

## Exercício

Calcule o valor de  $e^2$  usando a série anterior.

# Introdução

$n$	dividendo ( $x^n$ )	divisor ( $n!$ )	termo ( $\frac{x^n}{n!}$ )	soma ( $e^x$ )
0	1	1	$\frac{1}{1} = 1.0$	1.0
1	2	1	$\frac{2}{1} = 2.0$	3.0
2	4	2	$\frac{4}{2} = 2.0$	5.0
3	8	6	$\frac{8}{6} = 1.333333$	6.333333
4	16	24	$\frac{16}{24} = 0.666666$	7.0
5	32	120	$\frac{32}{120} = 0.266666$	7.266666
6	64	720	$\frac{64}{720} = 0.088888$	7.355555
7	128	5040	$\frac{128}{5040} = 0.0253968$	7.380952
8	256	40320	$\frac{256}{50320} = 0.0063492$	7.387301

Para todas as linhas, exceto a primeira

- O valor de  $x^n$  foi obtido multiplicando o valor de  $x^{n-1}$  (isto é,  $x^n$  da linha anterior) pelo valor de  $x$  (Ex:  $16 = 8 \times 2$ )
- O valor de  $n!$  foi obtido multiplicando o valor de  $(n-1)!$  (isto é,  $n!$  da linha anterior) pelo valor de  $n$  (Ex:  $120 = 24 \times 5$ )
- O valor de  $e^x$  foi obtido pelo valor de  $e^x$  da linha anterior mais o valor de  $\frac{x^n}{n!}$  (Ex:  $7.26666 = 7.0 + 0.26666$ )

- Veremos como expressar este tipo de processo com funções em Python
- Para isso precisamos entender dois conceitos
  - Variáveis
  - Instruções de repetição

# Variáveis



- Uma variável na matemática é um símbolo que representa um objeto matemático (número, vetor, função, etc)
  - Tem este nome porque os argumentos (variáveis) das funções podem variar
- Até agora pensamos em variáveis em Python como as variáveis na matemática
- Agora veremos que variáveis na computação podem não corresponder diretamente as variáveis em matemática

- Uma variável na computação é um nome associado com uma célula de memória (local de armazenamento)
  - Na célula de memória associada com a variável é armazenado um valor (número, string, etc)
- O fato de uma variável corresponder a uma célula de memória tem duas implicações
  - A célula de memória associada com a variável pode ser alterada
  - O valor armazenado na célula de memória associada com a variável pode ser alterado

# Variáveis

Considere o seguinte exemplo

```
>>> x = 10
```

```
>>> y = x
```

```
>>> x
```

```
10
```

```
>>> y
```

```
10
```

```
>>> x = 20
```

```
>>> x
```

```
20
```

```
>>> y
```

```
10
```

```
>>> x = x + 3
```

```
>>> x
```

```
23
```

- Quando a instrução `x = 10` é executado o Python reserva uma célula de memória para o valor 10 e associa esta célula ao nome `x`
- Quando a instrução `y = x` é executada a célula de memória associada com `x` passa também a ser associada com `y`, ou seja, tanto `x` quando `y` estão associados com a mesma célula de memória

- Quando a instrução  $x = 20$  é executado o Python reserva uma nova célula de memória para o valor 20 e associa esta célula com o nome  $x$ . Note que a memória associada com  $y$  não é alterada
- Quando a instrução  $x = x + 3$  é executada o Python lê o valor armazenado na célula de memória associado com  $x$  (20), soma com 3, reserva uma nova célula de memória para o valor 23 e associa esta célula ao nome  $x$ . A célula de memória previamente associada com  $x$  fica a disposição para armazenar outro valor

- A expressão  $x = x + 3$  parece estranha
  - Como  $x$  pode ser **igual** a  $x + 3$ !?
  - **Pare!**
  - O símbolo  $=$  representa igualdade na matemática, mas em Python ele tem outro significado

- O símbolo `=` é chamado de atribuição em Python
  - A expressão `a = 10` (lê-se a recebe 10) significa
    - Associe com o nome `a` uma célula de memória com o valor 10
  - A expressão `a = 2 * a + 1` (lê-se a recebe duas vezes a mais 1) significa
    - Associe com o nome `a` uma célula de memória com o valor  $2 * a + 1$  (isto é, o valor da célula de memória atualmente associado com `a` multiplicado por 2 mais 1)

- O valor do lado direito de  $=$  é uma expressão
  - Se a expressão for o nome de uma variável, nenhuma nova célula de memória é reservada
  - Se a expressão for uma constante ou uma expressão composta, ela é avaliada e o resultado é armazenado em uma nova célula de memória
- O valor do lado esquerdo de  $=$  deve ser um nome (que será associado com a célula de memória que contém o resultado da expressão do lado direito)
  - A expressão  $10 = x$  não é válida porque 10 não é um nome



- Porque usar o mesmo nome para armazenar valores distintos? Não podemos usar um novo nome para armazenar cada valor?
  - Sim podemos! É isso que temos feito até agora
- Mas as vezes precisamos armazenar valores que estão sendo calculados, como a soma de uma série, e não sabemos quantos valores intermediários serão calculados. Como dar um nome diferente para cada valor se não sabemos de antemão quantos valores teremos?
  - Em geral, usamos a mesma variável para armazenar valores distintos em processos que envolvem repetição de instruções

## Instruções de repetição

# Instruções de repetição

- Até agora vimos uma instrução de controle, a seleção (`if`)
- Agora veremos outra instrução de controle, o enquanto (`while`), uma instrução de repetição
  - A instrução `while` repete um conjunto de instruções até que uma determinada condição seja alcançada

# Instruções de repetição

Qual o valor de `i` após o término do `while`?

```
>>> i = 0
>>> while i != 5:
...     i = i + 1
...
>>> i
? 5
```

## Instruções de repetição

Quais os valores de `i` e `n` após o término do `while`?

```
>>> i = 1
>>> n = 1
>>> while i <= 4:
...     n = n * i
...     i = i + 1
...
>>> (i, n)
(?, ?) (5, 24)
```

## Instruções de repetição

Quais os valores de `i`, `a` e `b` após o término do `while`?

```
>>> i = 0
>>> a = 0
>>> b = 1
>>> while i < 6:
...     t = b
...     b = a + b
...     a = t
...     i = i + 1
...
>>> (i, a, b)
(?, ?, ?) (6, 8, 13)
```

## Instruções de repetição

Quais os valores de `cont`, `n` e `i` após o término do `while`?

```
>>> cont = 0
>>> n = 10
>>> i = n
>>> while i > 0:
...     if n % i == 0:
...         cont = cont + 1
...     i = i - 1
...
>>> (cont, n, i)
(?, ?, ?) (4, 10, 0)
```

# Instruções de repetição

Dado a seguinte função

```
def fun(a, b):  
    s = 0  
    while a <= b:  
        s = s + a  
        a = a + 1  
    return s
```

Quais os valores de x e y após a execução de fun?

```
>>> x = 3  
>>> y = fun(x, x + 2)  
>>> (x, y)  
(?, ?) (3, 12)
```



# Exemplos

## Exemplo

Defina uma função que calcule o produto de dois números inteiros usando uma sequência de somas.

## Exemplo

Seguindo a receita de projeto de funções

```
def multiplica(a, b):  
    '''  
    Inteiro Positivo, Inteiro Positivo -> Inteiro Positivo  
    Calcula o produto entre a e b usando sequencia de somas.  
    Exemplos  
    >>> multiplica(3, 4)  
    12  
    >>> multiplica(2, 5)  
    10  
    >>> multiplica(5, 0)  
    0  
    '''  
    return
```

## Exemplo

- O próximo passo é escrever o corpo
- Os passos anteriores do projeto das funções (principalmente os exemplos) devem guiar a escrita do corpo da função
- Como obter os resultados dos exemplos anteriores usando somas?

$$a \times b = \underbrace{a + a + \dots + a}_{b \text{ vezes}} = \sum_{n=1}^b a$$

## Exemplo

Vamos calcular `multiplica(3, 4)` usando uma sequência de somas. Temos que  $a = 3$  e  $b = 4$ ,  $n$  representa quantas vezes o valor de  $a$  foi somado e `prod` é o valor parcial da soma

$n$	prod (valor parcial de $a \times b$ )
0	0
1	3
2	6
3	9
4	12

## Exemplo

Observando a tabela anterior podemos fazer as seguintes observações

1. O valor inicial (primeira linha) de  $n$  é 0 e o de  $prod$  é 0
2. O valor de  $n$  da última linha é 4 (valor de  $b$ )
3. O valor de  $prod$  da última linha é 12 (valor de  $a \times b$ )
4. O valor de  $n$  de qualquer linha (diferente da primeira) é 1 a mais do que o valor de  $n$  da linha anterior
5. O valor de  $prod$  de qualquer linha (diferente da primeira) é 3 (valor de  $a$ ) a mais do que o valor de  $prod$  da linha anterior

## Exemplo

A partir das observações anteriores podemos escrever o corpo da função

```
def multiplica(a, b):  
    # valor inicial de n e prod (obs 1)  
    n = 0  
    prod = 0  
    # prod é calculado de forma iterativa  
    # isto é, através de uma repetição  
    # a repetição acontece enquanto n != b (obs 2)  
    while n != b:  
        # como o valor de n muda (obs 4)  
        n = n + 1  
        # como o valor de prod muda (obs 5)  
        prod = prod + a  
    # o resultado da função (obs 3)  
    return prod
```

## Exemplo

Dado dois inteiros positivos  $a$  e  $b$ , defina uma função que calcule o resto da divisão de  $a$  por  $b$  usando uma sequência de subtrações.



## Exemplo

Dado dois inteiros positivos  $m$  e  $n$ , defina uma função que encontre o menor número maior que  $m$  que seja divisível por  $n$ . Escreva duas versões da função, uma que usa repetição e outra não.

## Exemplo

Defina uma função que calcule o valor  $e^x$ , para um dado  $x$ , usando a série

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$$

# Atividades

1. Dado dois número inteiros positivos  $a$  e  $b$ , defina uma função que calcule o valor  $a^b$  usando uma sequência de multiplicações.
2. Dado dois número inteiros positivos  $a$  e  $b$ , onde  $a < b$ , defina uma função que conte a quantidade de números no intervalo  $[a, b]$  que sejam múltiplos de 7 e 13.

3. Defina uma função que calcule o valor  $\sin x$ , para um dado  $x$  em radianos, usando a série

$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots$$

4. Defina uma função que calcule o valor  $\cos x$ , para um dado  $x$  em radianos, usando a série

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots$$

5. Defina uma função que verifique se um dado número inteiro positivo é perfeito. Um número é perfeito se a soma dos seus divisores (excluindo ele mesmo) é igual a ele próprio. Por exemplo, 6 é um número perfeito pois  $6 = 1 + 2 + 3$ .
6. Defina uma função que verifique se um dado número inteiro positivo é primo. Um número inteiro positivo é primo se ele tem apenas dois divisores distintos, 1 e ele mesmo.

7. Dado um número inteiro positivo  $n \geq 4$ , defina uma função que devolva dois números  $a$  e  $b$ , tal que  $a$  e  $b$  sejam primos e  $n = a + b$ .

- Livro Pense em Python 2ª edição. Allen B. Downey
  - Capítulo 7 - Iteração