

# Funções Recursivas

Prof. Alberto Costa Neto  
Programação em Python

# Fatorial

- Calculando o fatorial de 5:

$$(1) \quad 5! = 5 * 4 * 3 * 2 * 1$$

- Generalizando, temos que:

$$(2) \quad n! = n * n-1 * n-2 * \dots * 1$$

- Portanto, poderíamos reescrever (1) como:

$$(3) \quad 5! = 5 * 4!$$

# Fatorial

- Calculando o fatorial de 5:

$$5! = 5 * 4!$$

$$4! = 4 * 3!$$

$$3! = 3 * 2!$$

$$2! = 2 * 1!$$

$$1! = 1$$

$$0! = 1$$

## Definição

$$n! = 1, \quad \text{se } n = 0$$

$$n * (n-1)!, \quad \text{se } n > 0$$

# Definindo uma Função Recursiva

- Em Python, assim como em outras linguagens de programação, uma **função** pode chamar outras funções
- Quando uma função chama ela mesma, é denominada:

**Função Recursiva**



Fonte: <https://i1.wp.com/sala.inf.br/wp-content/uploads/2017/05/desmistificando.jpg>

# Como elaborar uma função recursiva

1. Definir o **caso base** ou **caso de parada**
  - No fatorial, ocorre quando  $n = 0$
2. Definir o **caso geral**, que **levará em algum momento ao caso base**
  - No fatorial, ocorre quando  $n > 0$

## Definição

$$\begin{array}{ll} n! = 1, & \text{se } n = 0 \\ n * (n-1)!, & \text{se } n > 0 \end{array}$$

# Definindo Fatorial em Python

```
def fat(n):  
    if n == 0:  
        return 1  
    elif n > 0:  
        return n * fat(n-1)
```

## Definição

$$\begin{array}{ll} n! = 1, & \text{se } n = 0 \\ n * (n-1)!, & \text{se } n > 0 \end{array}$$

# Definindo Fatorial em Python

```
def fat(n):  
    if n == 0:  
        return 1  
    else:  
        return n * fat(n-1)
```

Se  $n$  é um número natural, podemos usar o else para simplificar

## Definição

$$\begin{aligned} n! &= 1, & \text{se } n = 0 \\ n * (n-1)!, & \text{se } n > 0 \end{aligned}$$

Programa

```
print(fat(3))
```

# Executando Fatorial

Definição

$$\begin{aligned} n! &= 1, & \text{se } n = 0 \\ n * (n-1)!, & \text{se } n > 0 \end{aligned}$$



Programa  
`print(fat(3))`

Função (com  $n = 3$ )  
`def fat(n):`  
    `if n == 0:`  
        `return 1`  
    `else:`  
        `return n * fat(n-1)`

# Executando Fatorial

Definição

$$\begin{aligned} n! &= 1, & \text{se } n = 0 \\ n * (n-1)!, & \text{se } n > 0 \end{aligned}$$

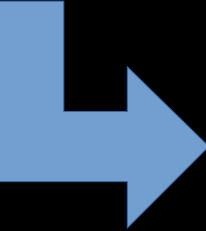
# Executando Fatorial

Programa  
`print(fat(3))`

Função (com  $n = 3$ )  
`def fat(n):`  
    `if n == 0:`  
        `return 1`  
    `else:`  
        `return n * fat(n-1)`


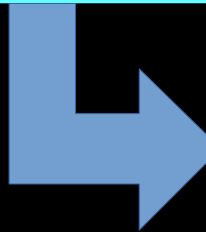
Definição

$n! = 1,$       se  $n = 0$   
 $n * (n-1)!,$    se  $n > 0$



Função (com  $n = 2$ )  
`def fat(n):`  
    `if n == 0:`  
        `return 1`  
    `else:`  
        `return n * fat(n-1)`

Função (com  $n = 1$ )  
`def fat(n):`  
    `if n == 0:`  
        `return 1`  
    `else:`  
        `return n * fat(n-1)`



Função (com  $n = 0$ )  
`def fat(n):`  
    `if n == 0:`  
        `return 1`  
    `else:`  
        `return n * fat(n-1)`

# Sequência de Fibonacci

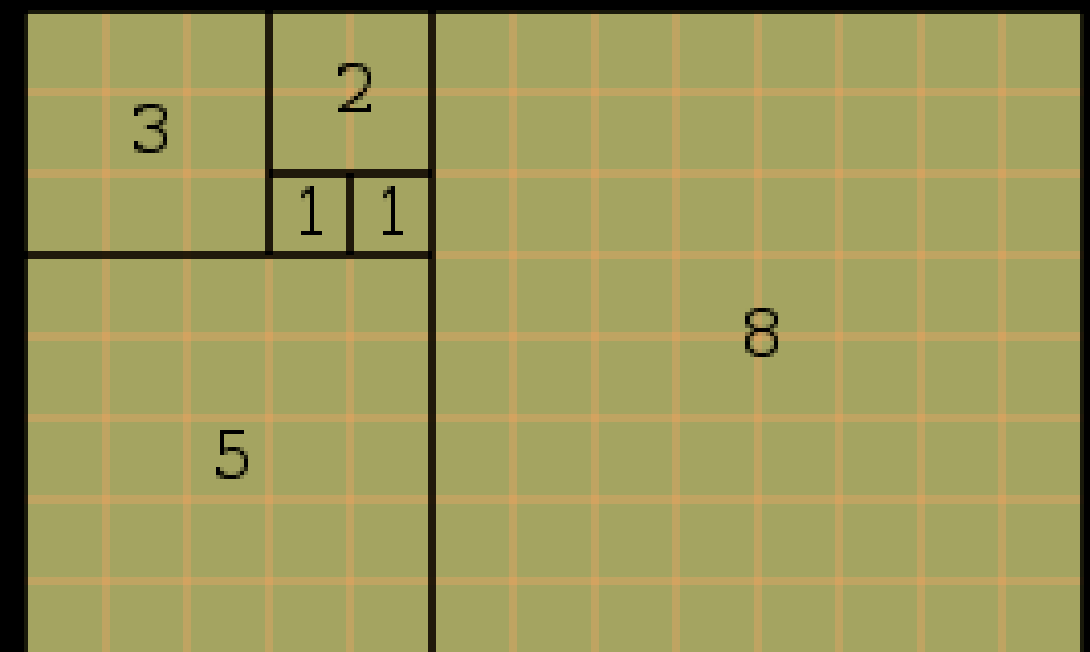
- Na matemática, a **Sequência de Fibonacci** é uma sequência de números inteiros, começando normalmente por 0 e 1, na qual, cada termo subsequente corresponde à soma dos dois anteriores.

Fonte: [https://pt.wikipedia.org/wiki/Sequência\\_de\\_Fibonacci](https://pt.wikipedia.org/wiki/Sequência_de_Fibonacci)

## Definição

$$\begin{aligned} \text{fib}(n) &= 0, & \text{se } n &= 0 \\ \text{fib}(n) &= 1, & \text{se } n &= 1 \\ \text{fib}(n) &= \text{fib}(n-1) + \text{fib}(n-2), & \text{se } n &> 1 \end{aligned}$$

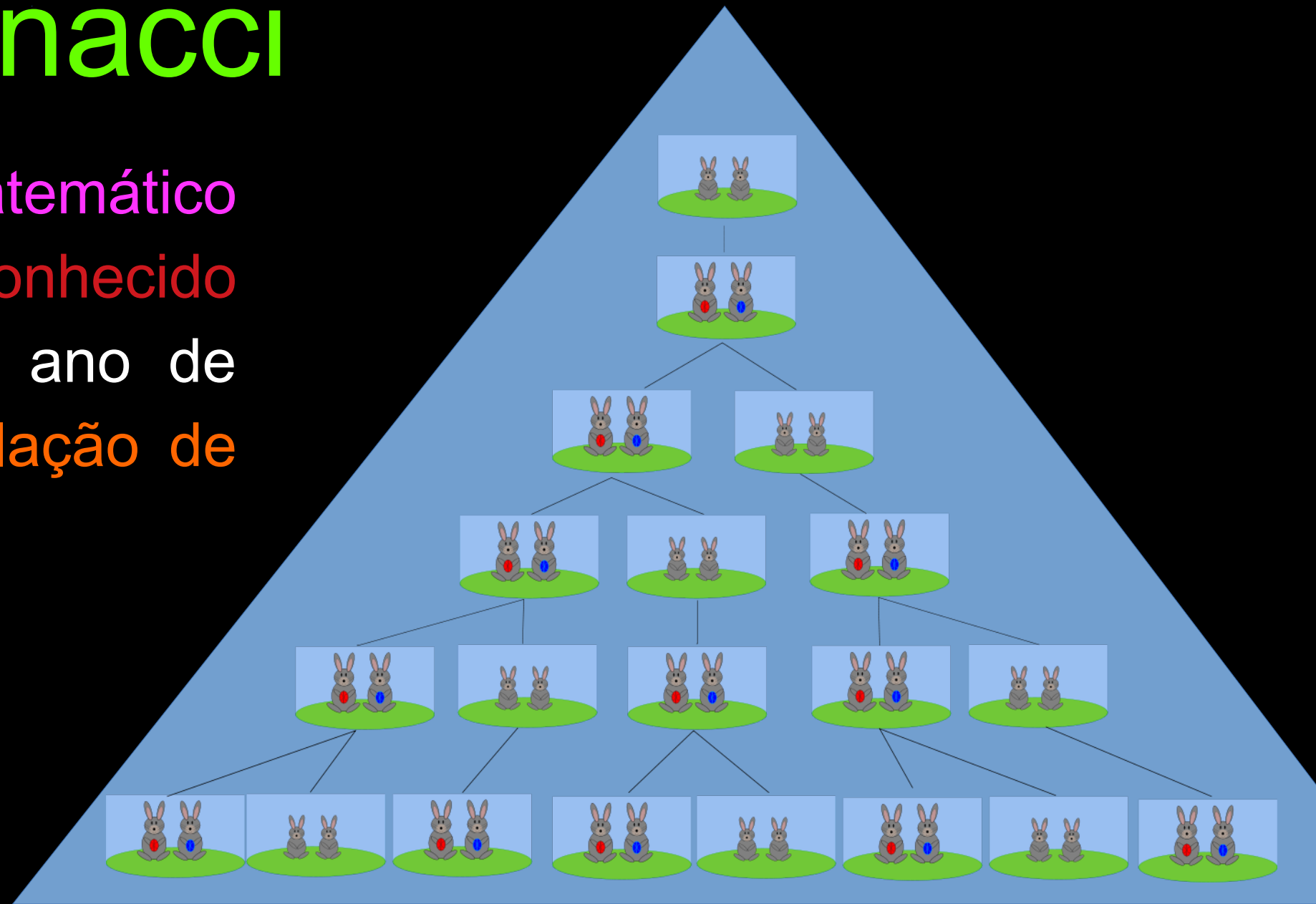
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...



# Sequência de Fibonacci

- A sequência recebeu o nome do **matemático italiano Leonardo de Pisa, mais conhecido por Fibonacci**, que descreveu, no ano de 1202, o **crescimento de uma população de coelhos**, a partir da sequência.

- No primeiro mês nasce apenas um casal;
- Casais amadurecem sexualmente (e reproduzem-se) apenas após o 2º mês;
- Não há problemas genéticos no cruzamento consanguíneo;
- Todos os meses, cada casal fértil dá a luz a um novo casal; e
- Os coelhos nunca morrem.



Fonte: [https://pt.wikipedia.org/wiki/Sequência\\_de\\_Fibonacci](https://pt.wikipedia.org/wiki/Sequência_de_Fibonacci)

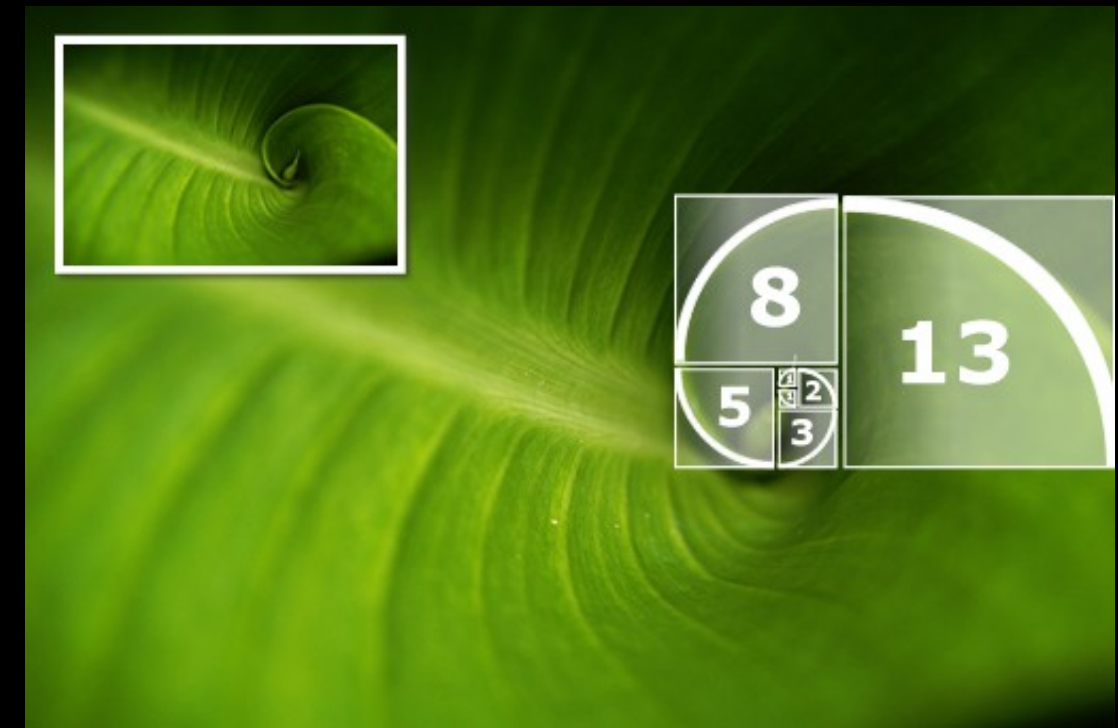
# Sequência de Fibonacci

## Definição

$$\begin{aligned} \text{fib}(n) &= 0, & \text{se } n &= 0 \\ \text{fib}(n) &= 1, & \text{se } n &= 1 \\ \text{fib}(n) &= \text{fib}(n-1) + \text{fib}(n-2), & \text{se } n &> 1 \end{aligned}$$

```
def fib(n):  
    if n == 0 or n == 1:  
        return n  
    else:  
        return fib(n-1) + fib(n-2)
```

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...



Espiral formada pela folha de uma bromélia, Fonte:  
[https://pt.wikipedia.org/wiki/Sequência\\_de\\_Fibonacci#/media/File:Bromelia.png](https://pt.wikipedia.org/wiki/Sequência_de_Fibonacci#/media/File:Bromelia.png)

# Executando Fibonacci

Programa  
`print(fib(3))`

Função (com  $n = 5$ )

```
def fib(n):  
    if n == 0 or n == 1:  
        return n  
    else:  
        return fib(n-1) + fib(n-2)
```

