Processamento da Informação
– Teoria –

Recursividade

Semana 08
Prof. Jesús P. Mena-Chalco

15/06/2013

Uma função chama outra função

Vimos exemplos de uma função chamar uma outra função.

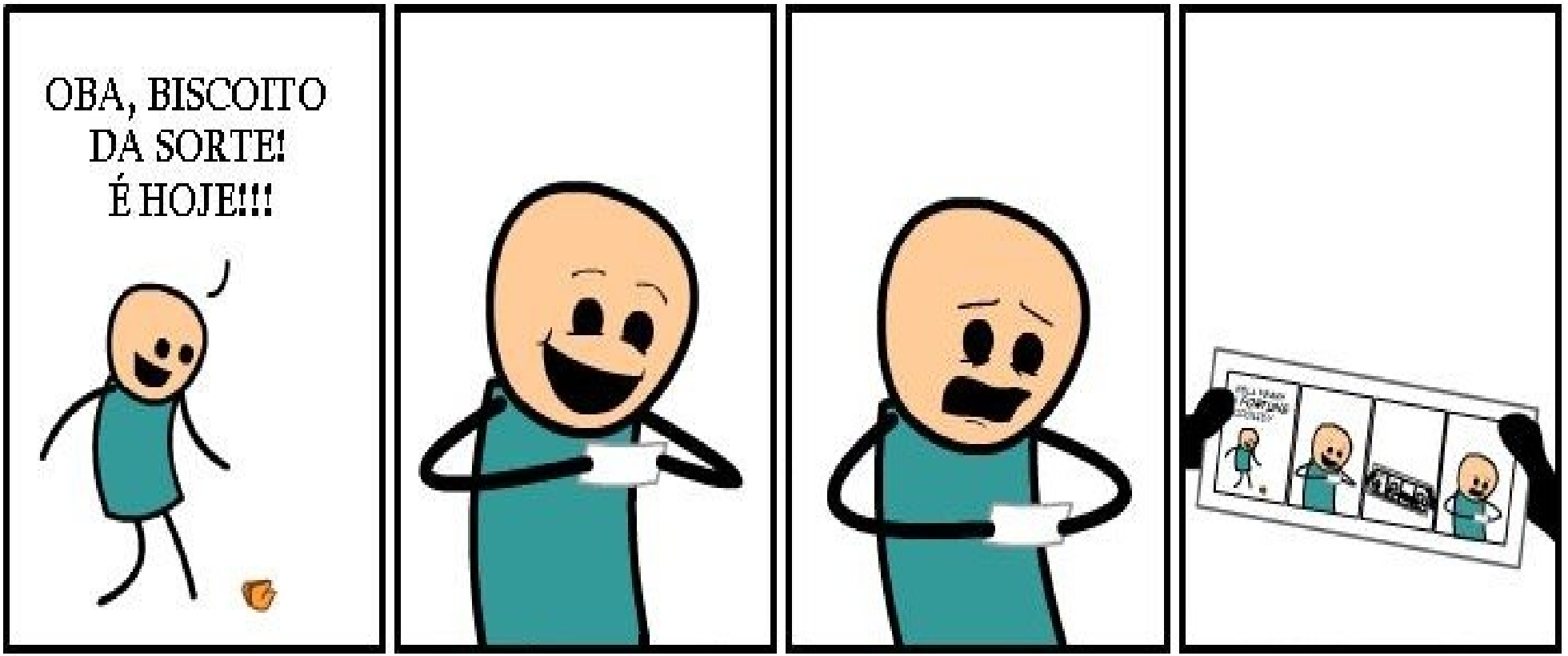
```
def fatorial1(n):
    mult = 1
    for p in range(1,n+1):
        mult = mult*p
    return mult
```

A função **fatoria1** chama a função **range**.

Uma função chama outra função

Também é válido uma função chamar a si mesma.

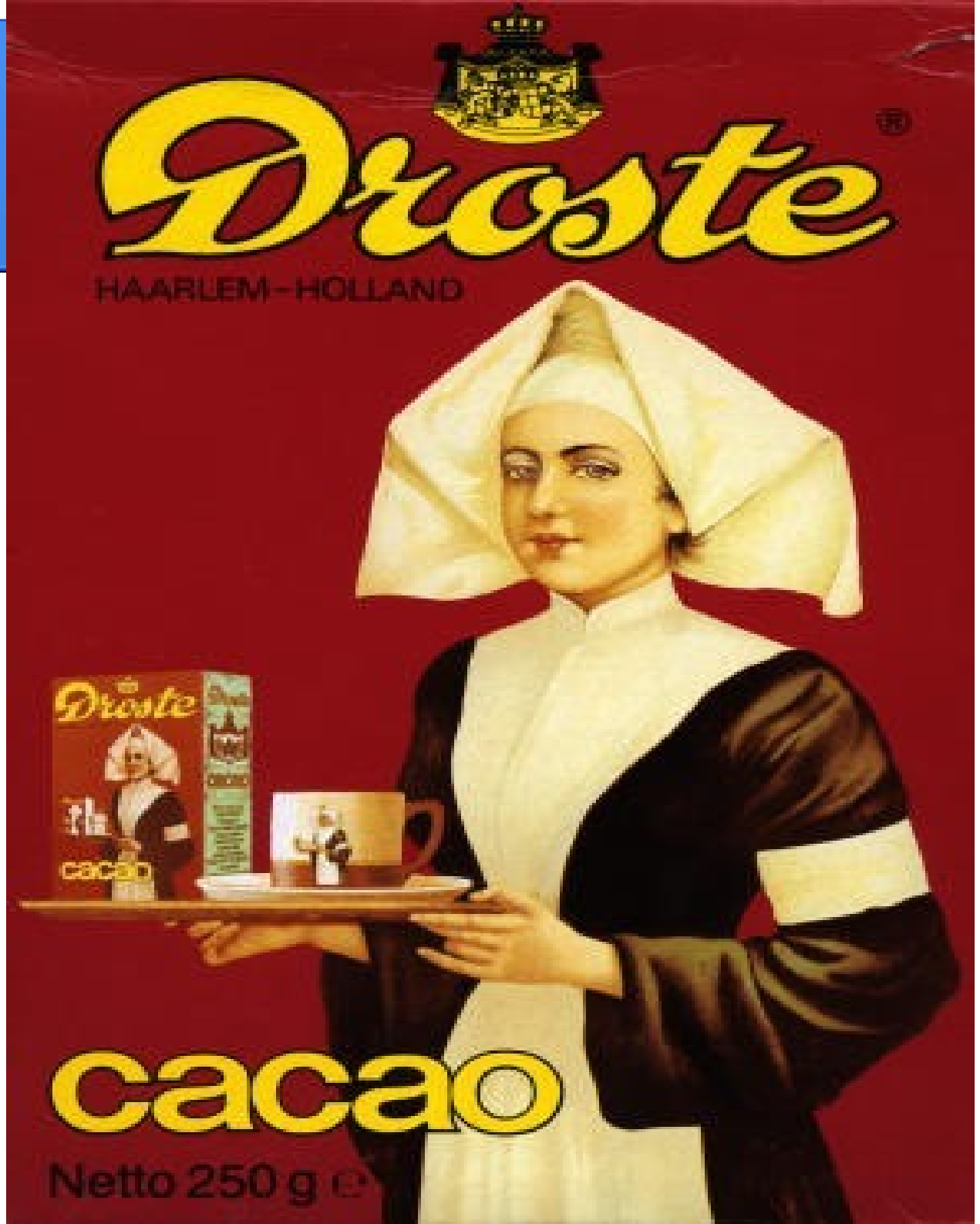
Recursividade



Recursividade é uma das coisas mágicas e interessantes em Programação.

Recursividade

Anuncio de cacao com uma imagem recursiva.



Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```

Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```

contagem_regressiva(3)

Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```

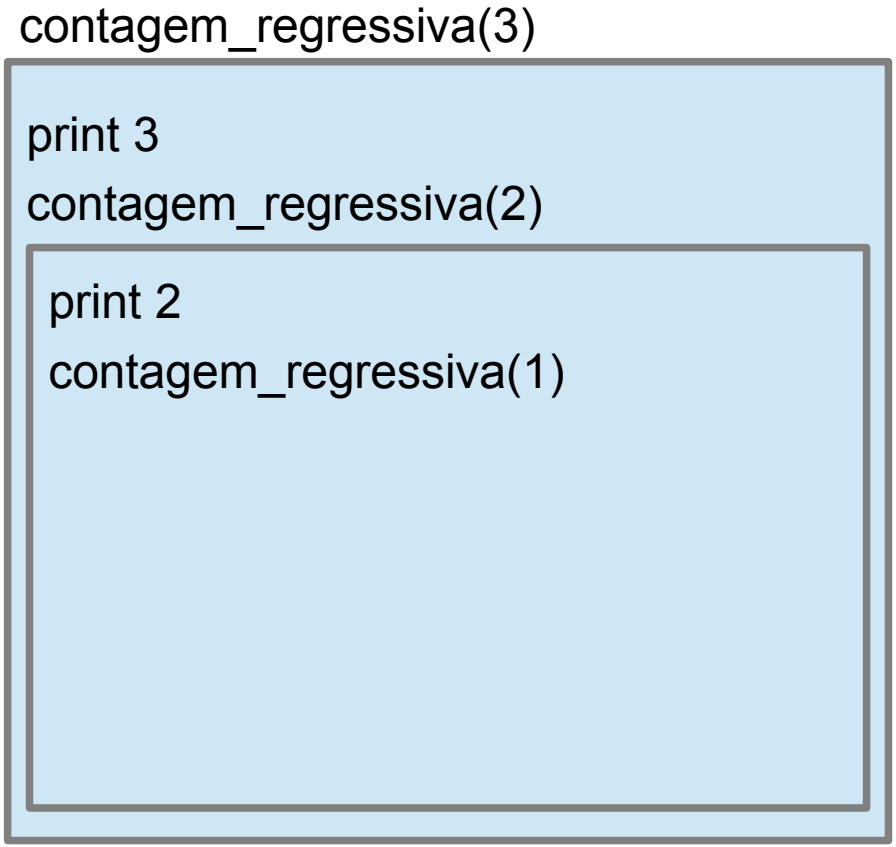
contagem_regressiva(3)

print 3

contagem_regressiva(2)

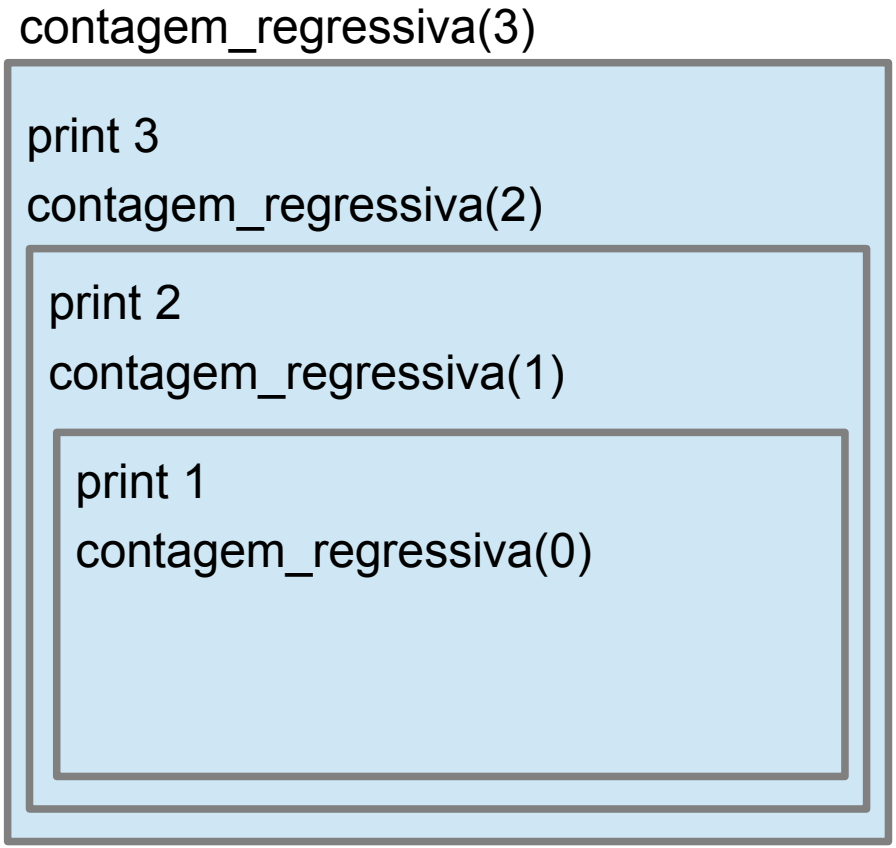
Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```



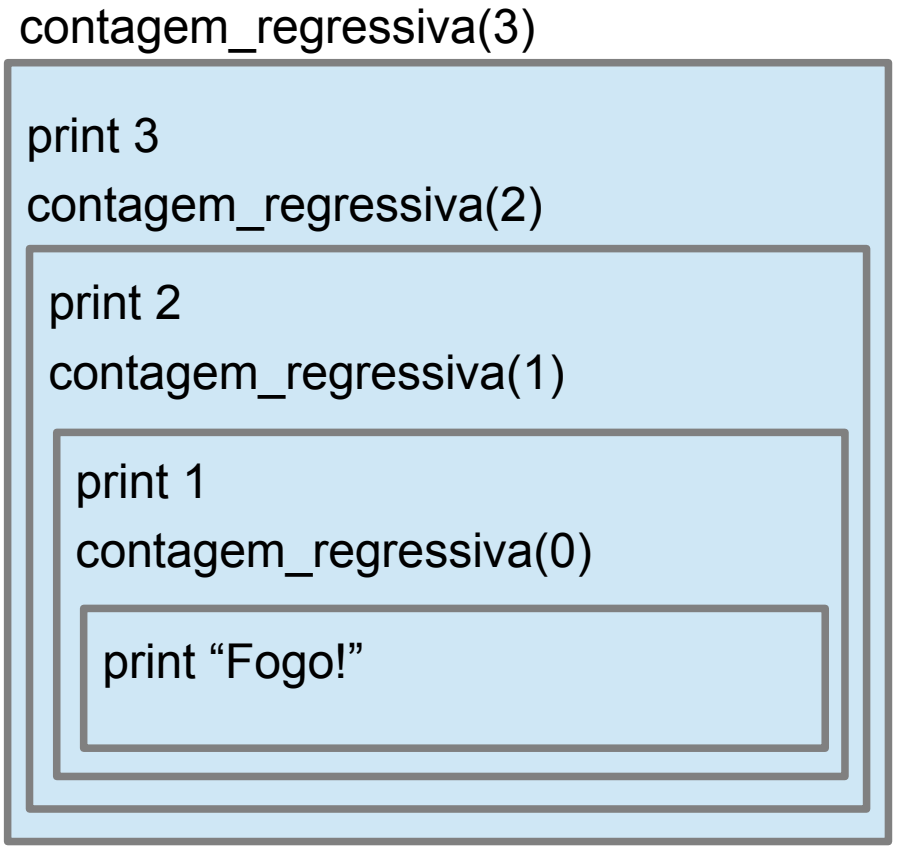
Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```



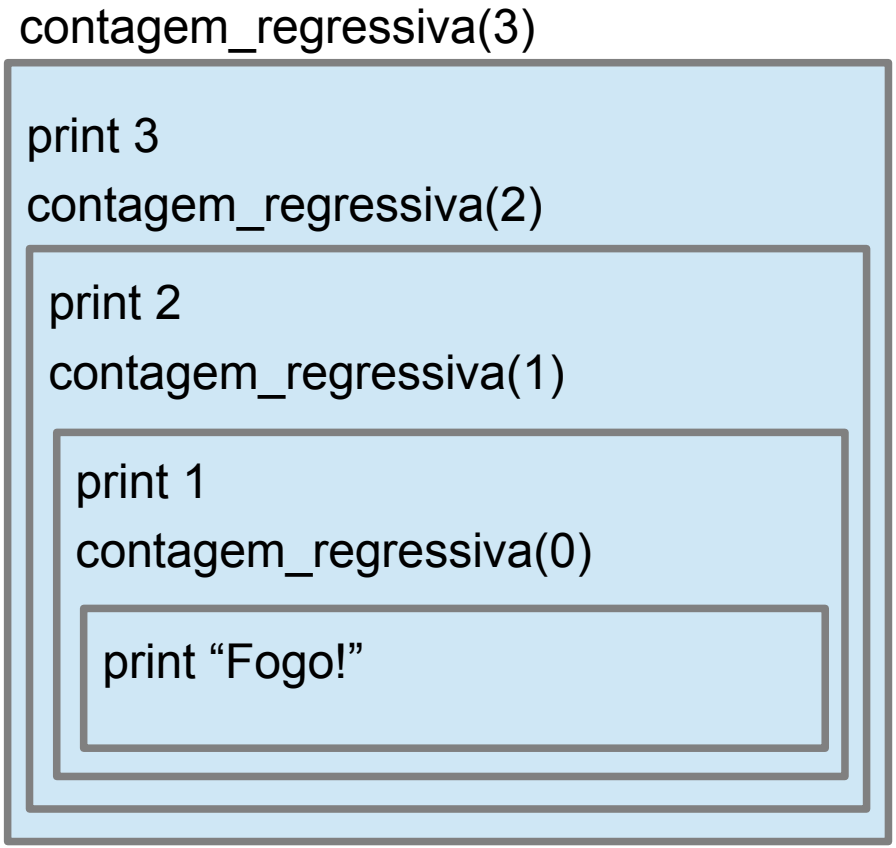
Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```



Recursividade

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```



O processo de uma função chamando a si mesma é chamado de **Recursividade,** e tais funções são ditas **recursivas.**

Recursividade x Iteração

Porque não usar Iteração ao invés de Recursividade?

Depende muito do estilo de programação. Entretanto, algumas vezes é mais apropriado usar Recursividade para resolver um problema.

```
def contagem_regressiva(n):
    if n==0:
        print "Fogo!"
    else:
        print n
        contagem_regressiva(n-1)
```

```
def contagem_regressiva2(n):
    while n>0:
        print n
        n = n-1
    print "Fogo!"
```

Fibonacci (iterativo)

Crie uma função que permita imprimir o **n-éssimo** número da sequência de Fibonacci. Considere apenas um laço 'while'.

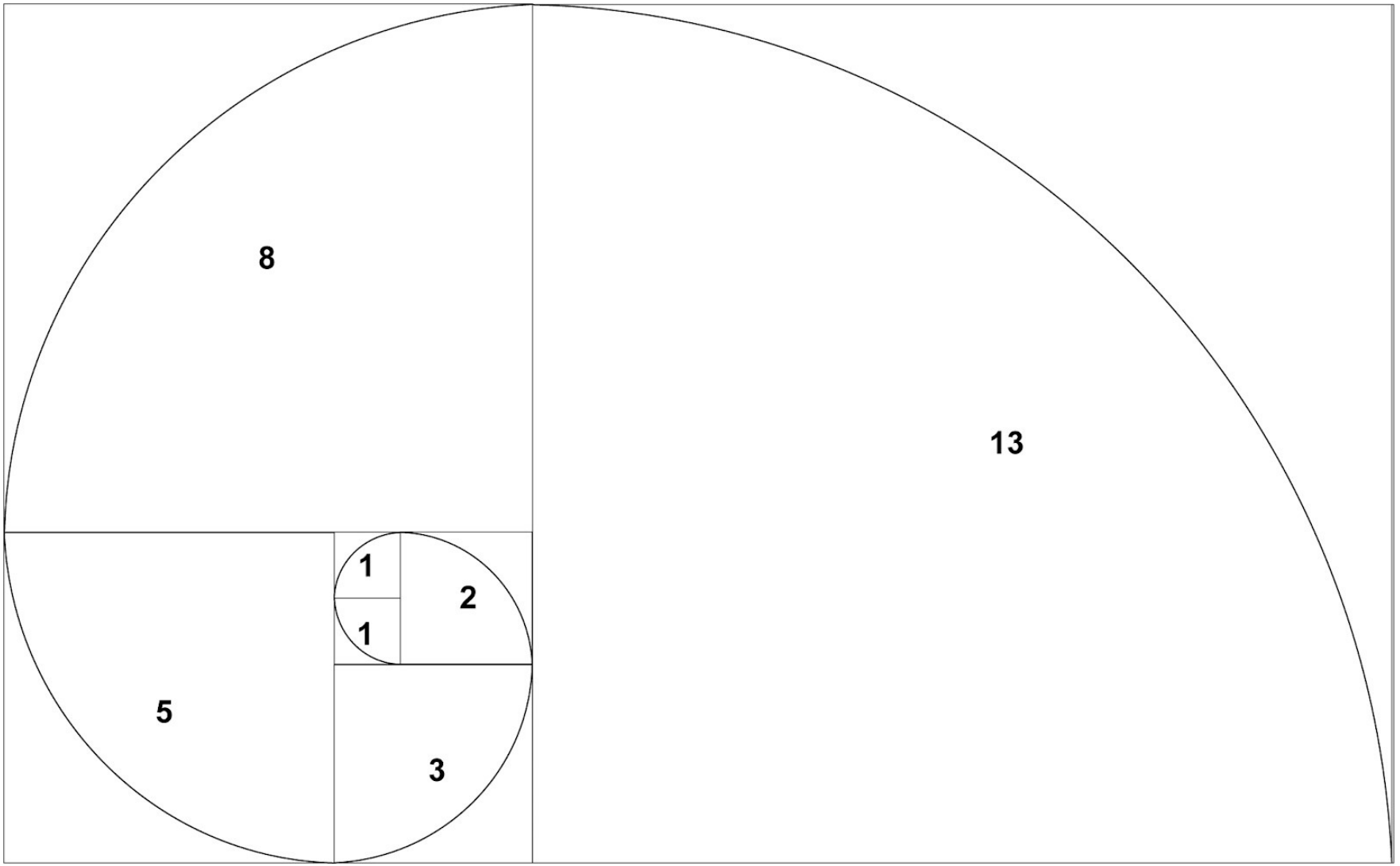
F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
0	1	1	2	3	5	8	13	21	34	55	89	144	233	377	610	987	1597	2584	4181	6765

Cabeçalho: def fibonacci(n):

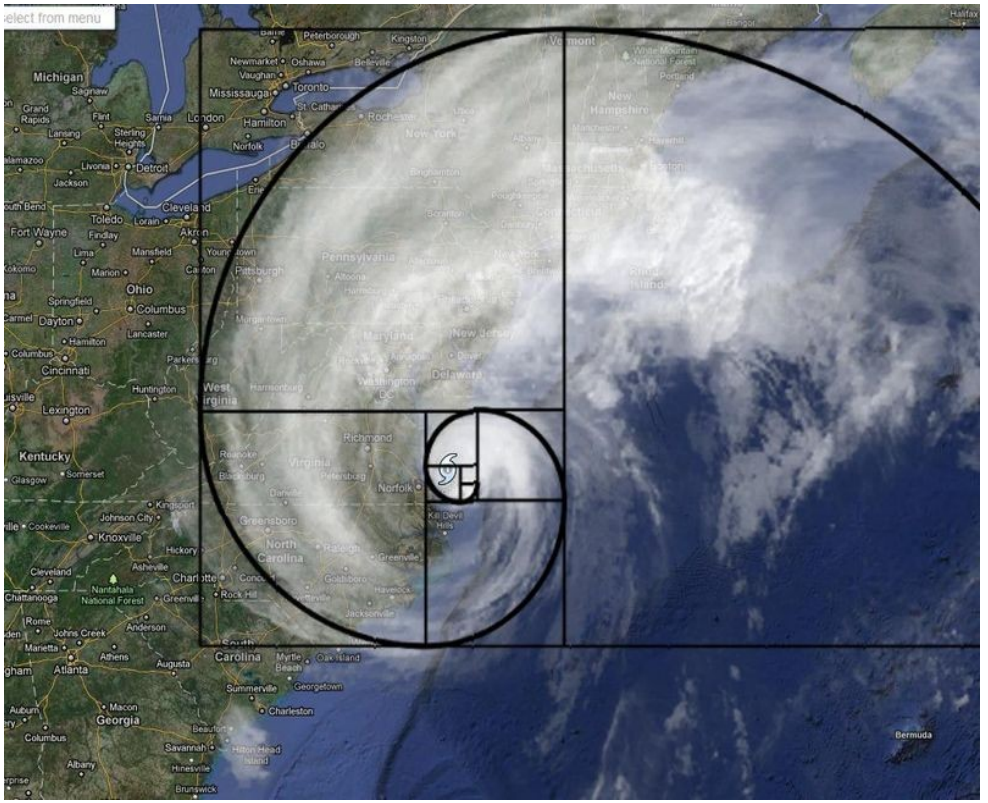
Fibonacci (iterativo)

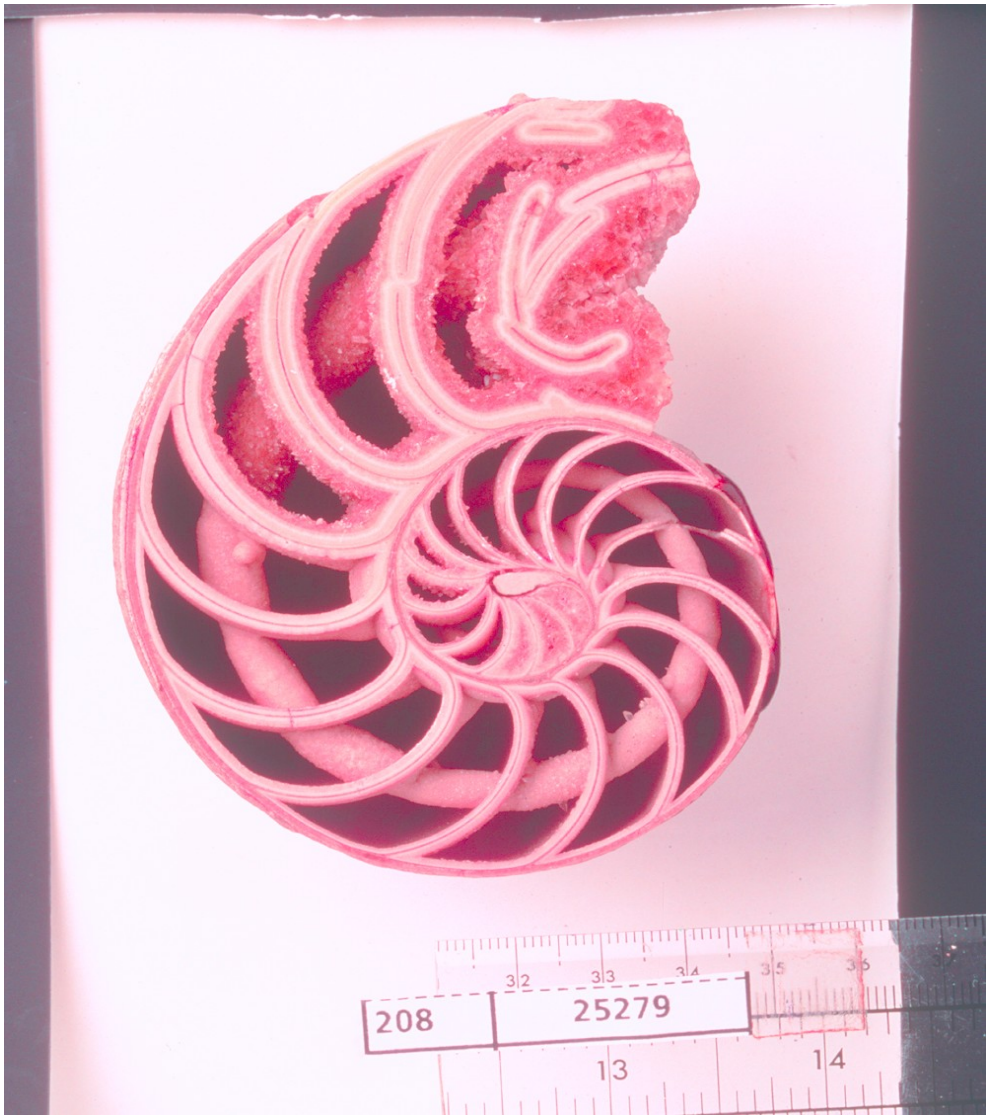
```
def fibonacci(n):
    k  = 1
    t1 = 0
    t2 = 1
    while k<n:
        seguinte = t1+t2
        t1 = t2
        t2 = seguinte
        k  = k+1
    print t2
```

Fibonacci



Fibonacci

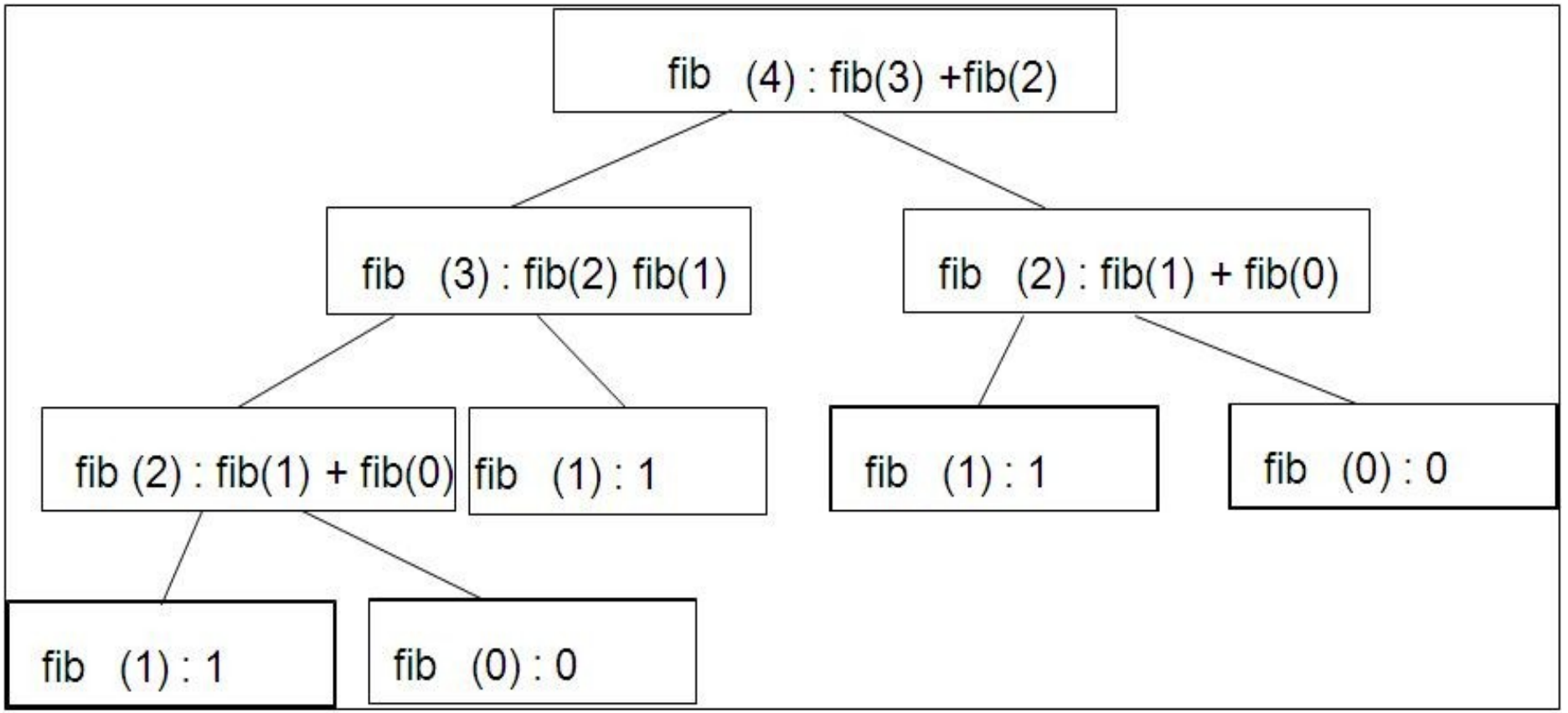

Michigan
Toronto
Mississauga
Hamilton
Detroit
Cleveland
Pennsylvania
Ohio
Columbus
Cincinnati
Kentucky
Lexington
Virginia
Richmond
Norfolk
North Carolina
Greensboro
Raleigh
Knoxville
Atlanta
Georgia
Charleston
Bermuda


208
25279
13
14

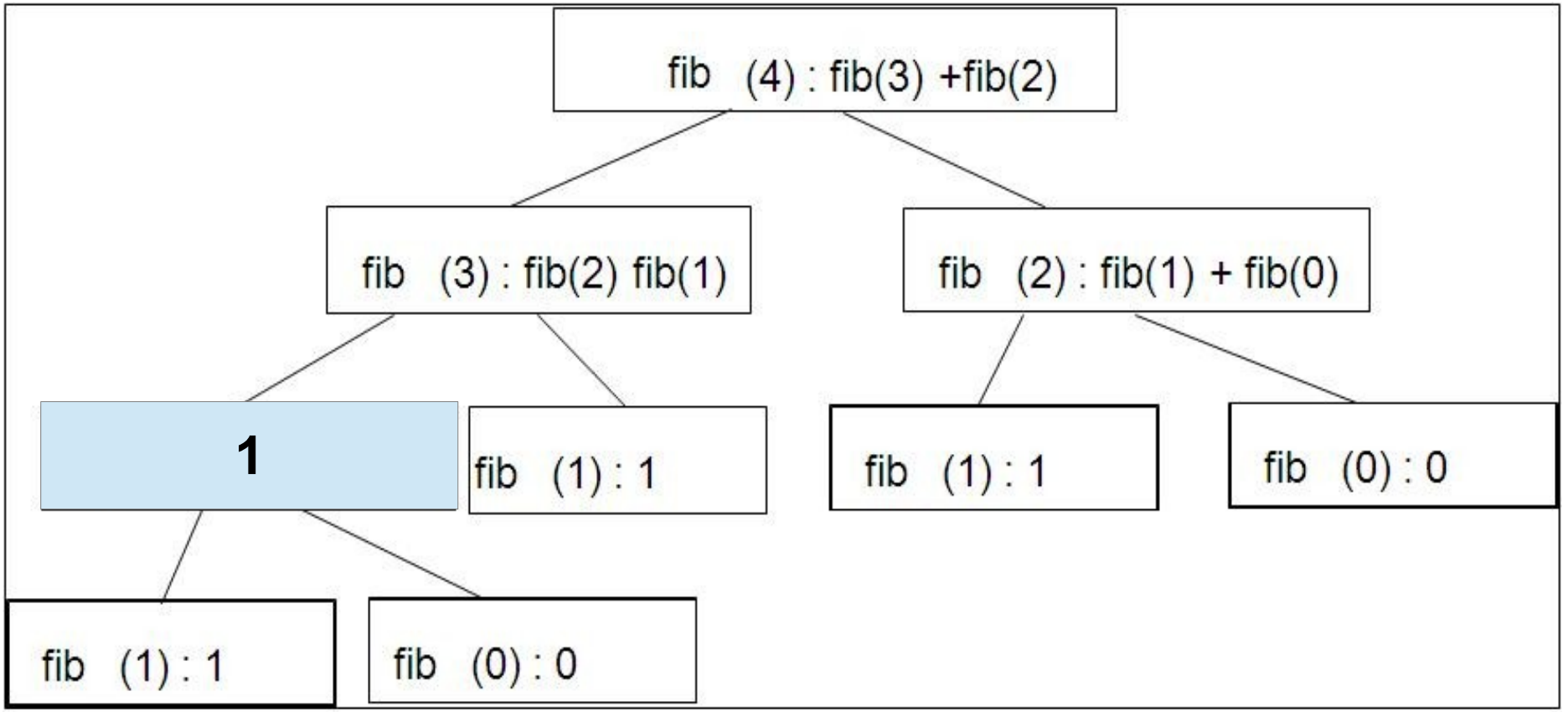
Fibonacci (recursivo)

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

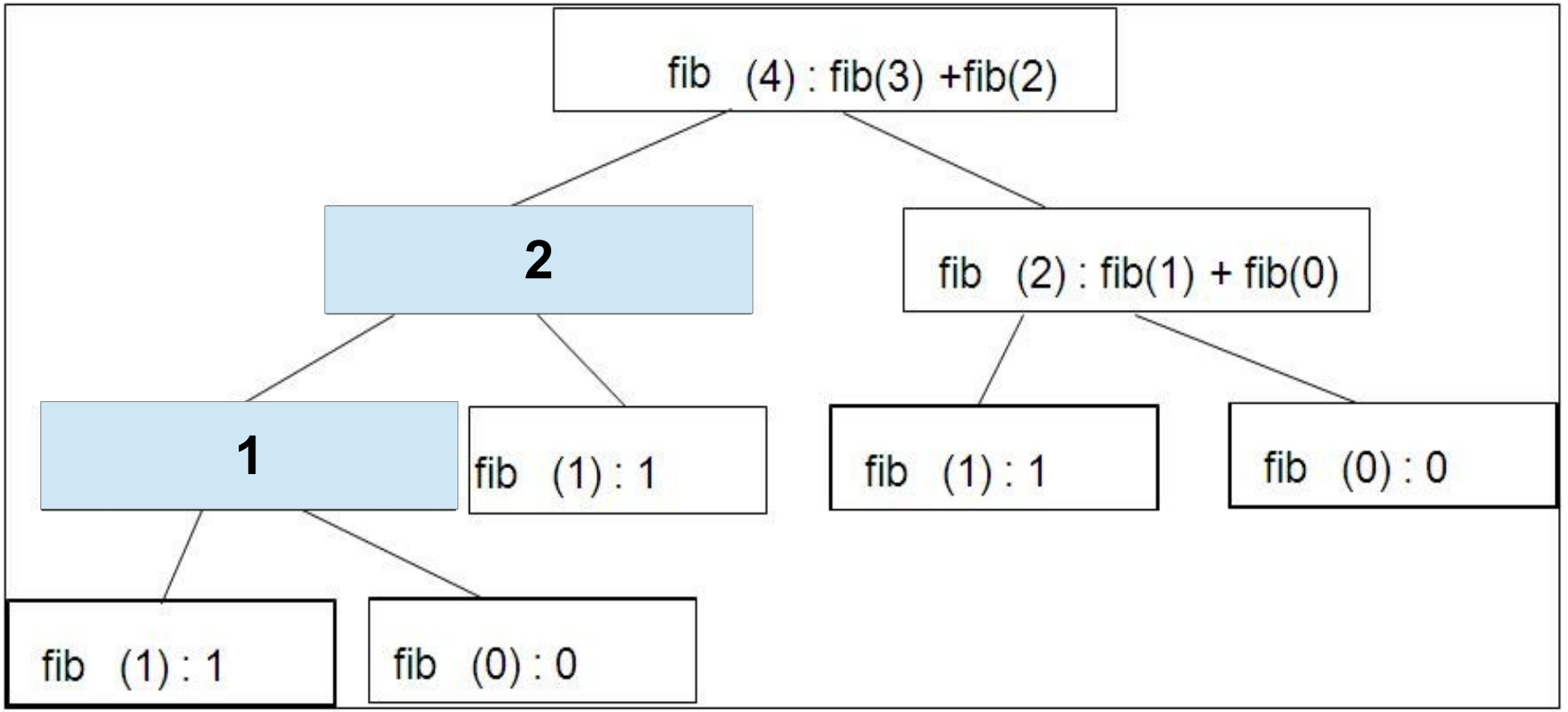
```
def fib(n):
    if n==0:
        return 0
    elif n==1:
        return 1
    else:
        return fib(n-1)+fib(n-2)
```



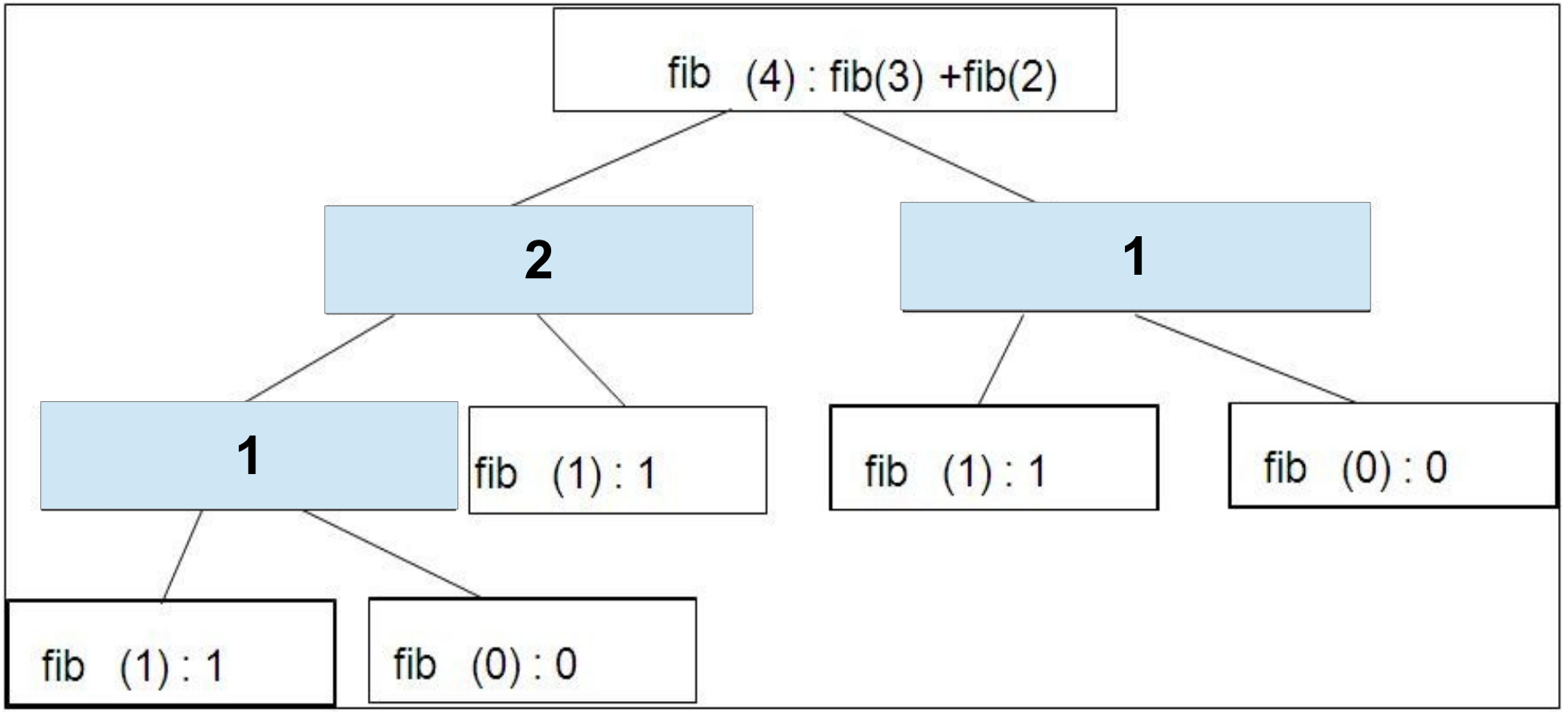
```
def fib(n):
    if n==0:
        return 0
    elif n==1:
        return 1
    else:
        return fib(n-1)+fib(n-2)
```



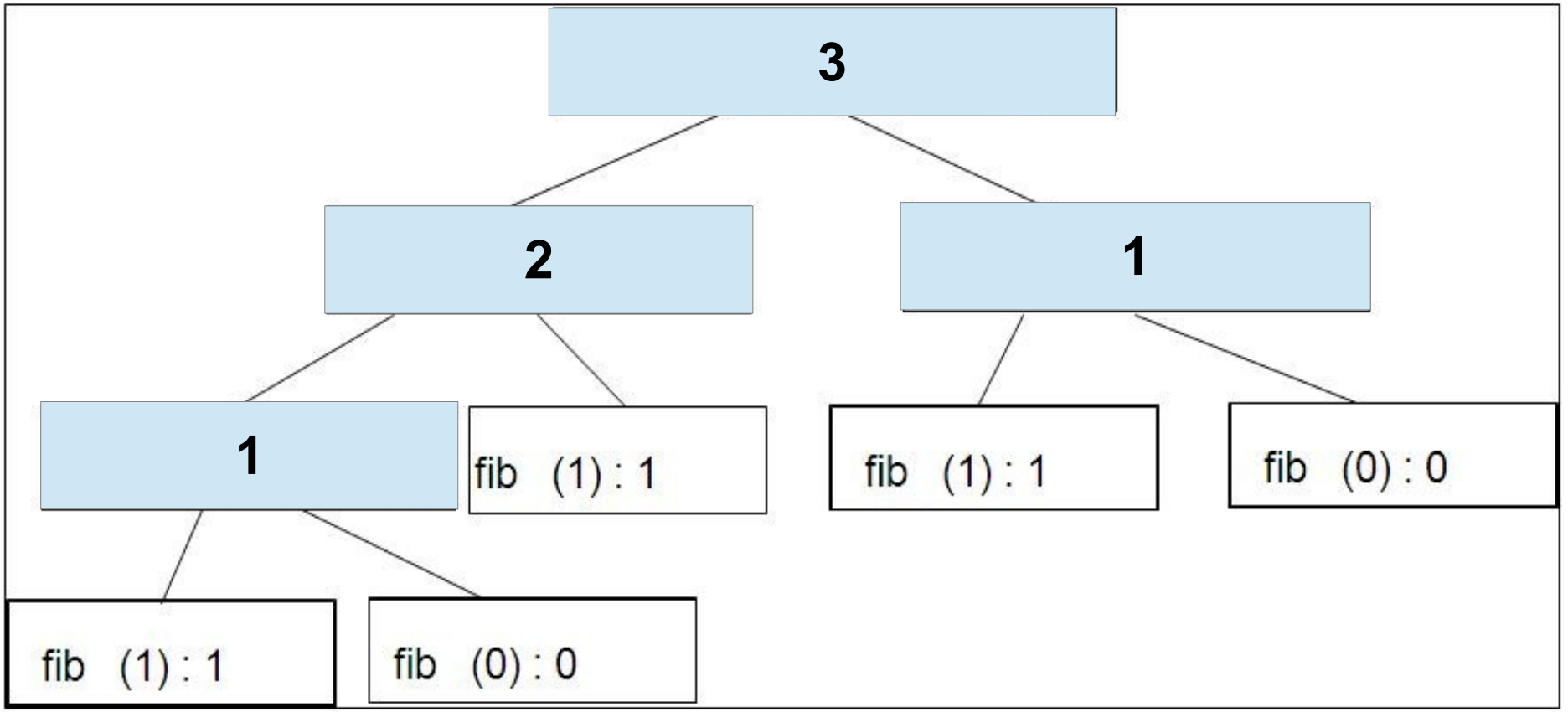
```
def fib(n):
    if n==0:
        return 0
    elif n==1:
        return 1
    else:
        return fib(n-1)+fib(n-2)
```



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



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



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

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

Fatorial (iterativo)

```
def fatorial(n):
    f = 1
    while n>1:
        f = f*n
        n = n-1
    return f
```

Fatorial (recursivo)

Crie uma **função recursiva** que permita calcular o fatorial de um número inteiro dado como parâmetro.

$$fatorial(n) = \begin{cases} 1 & , \text{se } n = 0 \\ n.fatorial(n-1) & , \text{se } n > 0 \end{cases}$$

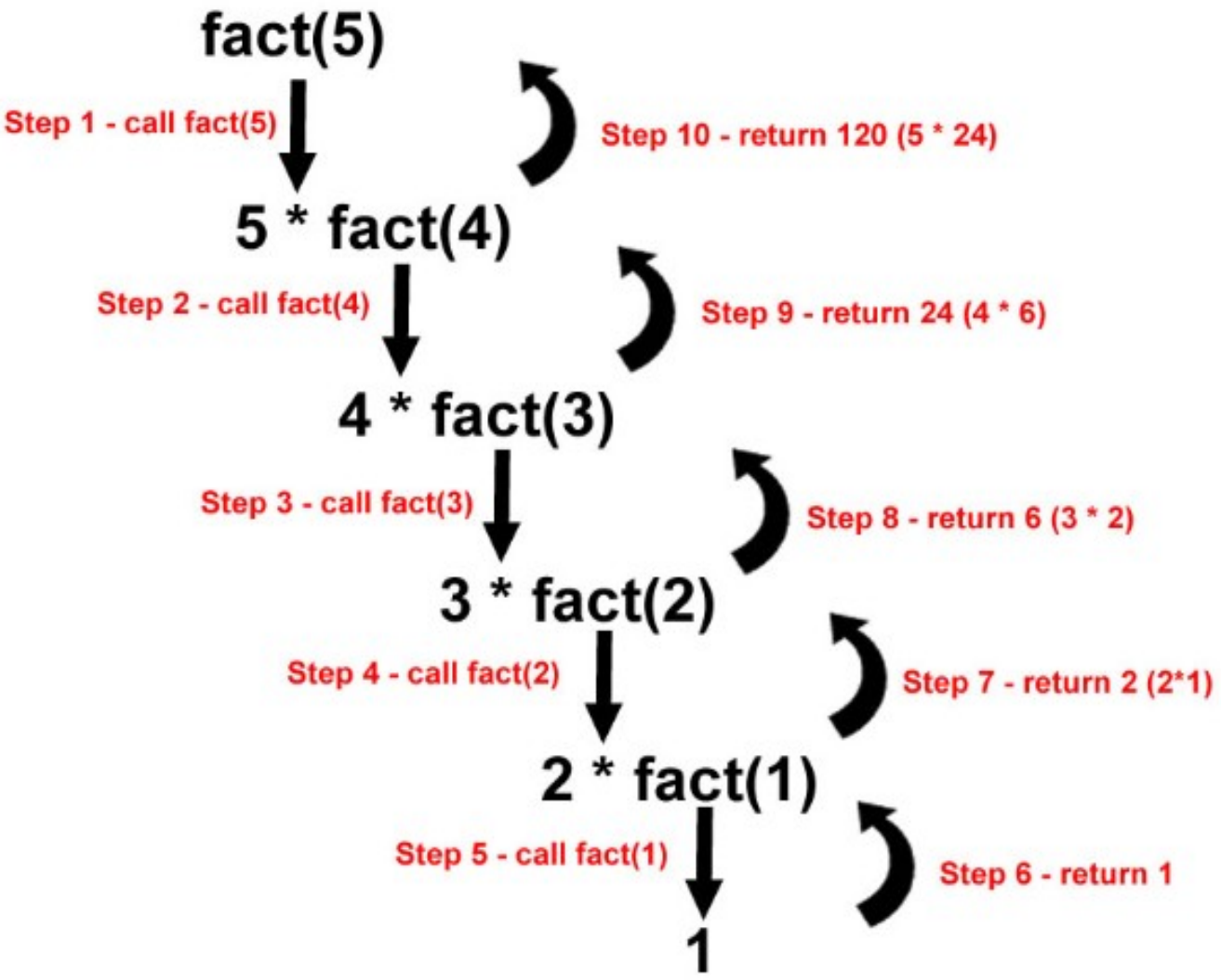
Cabeçalho: def fact(n):

Fatorial (iterativo)

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

Fatorial (iterativo)

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



Somatória (recursivo)

Crie uma **função recursiva** que permita somar todos os números naturais no intervalo 1 até n (dado como entrada).

Cabeçalho: def somatoria(n):

Somatória (recursivo)

```
def somatoria(n):
    if n==1:
        return 1
    else:
        return n+somatoria(n-1)
```

Atividade em aula

Questão 1: 2 pontos

Questão 2: 2 pontos

Questão 3: 3 pontos

Questão 4: 3 pontos

Atividade em aula

Questão 1:

Indique o que faz a seguinte função:

```
def recursiva():
    print 'recursiva'
    Recursiva()
```

Resposta:
Recursividade infinita.
A função para quando a profundidade máxima da recursividade é alcançada.

StackOverflowError

Atividade em aula

Questão 2:

Crie uma função recursiva para calcular n elevado a x (n^x)

```
def potencia(n,x):
    if x==1:
        return n
    else:
        return n*potencia(n,x-1)
```

Atividade em aula

Questão 3:

Indique o resultado apresentará a execução da seguinte função. Considere como parâmetros de entrada: frase='ufabc'. O que faz a função?

```
def funcaoR(frase):
    if len(frase)==1:
        return frase
    else:
        return funcaoR(frase[1:])+frase[0]
```

'ufabc' → 'cbafu'

Retorna a frase (uma string) de trás para frente.

Atividade em aula

Questão 4:

Indique o que realiza a seguinte função?
Teste com número1=13, número2=2.

```
def funcaoC(numero1, numero2):
    if numero1<numero2:
        print numero1
    else:
        funcaoC(numero1/numero2, numero2)
        print numero1%numero2
```

1
1
0
1

Converte o numero1, da base 10, para a base numero2.