Definindo funções

SSC0301

Prof Marcio Delamaro

ICMC USP SÃO CARLOS SSC

Biblioteca padrão

- len(s) – tamanho de um string ou lista
- math.sin(x) – seno do ângulo x
- type(r) – indica o tipo de um objeto r
- random.randint(a,b) – retorna um número aleatório no intervalo (a,b)
- Imagine, por exemplo, o que aconteceria se não existisse math.sin()

ICMC USP SÃO CARLOS SSC

Com X Sem

```
z1 = math.sin(x)

z2 = math.sin(y)
```

```
seno = x
while erro < 0.00001
    termo = ……
    ……
z1 = seno
seno = y
while erro < 0.00001
    termo = ……
    ……
z2 = seno
```

ICMC USP SÃO CARLOS SSC

O que é uma função?

- Uma trecho de programa que alguém implementou
- Pode ser “chamado” dentro do nosso programa
- Pode receber valores como parâmetros
  - z1 = math.sin(x)
- Pode devolver um valor
  - z1 = math.sin(x)

O que é uma função?

- Uma trecho de programa que alguém implementou
- Pode ser “chamado” dentro do nosso programa
- Pode receber valores como parâmetros
  - z1 = math.sin(x)
- Pode devolver um valor
  - z1 = math.sin(x)

Cada função recebe tipos diferentes de parâmetros e devolve tipos diferentes de objetos.

ICMC USP SÃO CARLOS SSC

Como funciona

meu_programa.py

```
while ▒▒▒▒▒▒▒▒:
    if ▒▒▒▒▒▒▒▒▒▒▒▒:
       print('▒▒▒▒')
    else:
       ▒▒▒▒▒▒▒▒

y = sin(0.7)

for ▒▒▒▒▒▒▒▒:
    ▒▒▒▒
    ▒▒▒▒▒▒▒▒
if ▒▒▒▒▒▒▒▒▒▒:
    ▒▒▒▒▒
```

Alguém implementou essa função

função math.sin()

```
if ▒▒▒▒▒▒▒▒▒▒:
    ▒▒▒▒▒
while ▒▒▒▒▒▒▒▒:
for ▒▒▒▒▒▒▒▒:
    ▒▒▒▒
```

Meu programa chama essa função

Como funciona

meu_programa.py

```
while ▪▪▪▪▪▪▪▪▪▪:
    if ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪:
        print('▪▪▪▪▪')
    else:
        ▪▪▪▪▪▪▪▪

y = sin(0.7)

for ▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪
    ▪▪▪▪▪▪▪▪▪
if ▪▪▪▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪▪
```

função math.sin()

```
if ▪▪▪▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪▪
while ▪▪▪▪▪▪▪▪▪▪:
for ▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪
```

Como funciona

meu_programa.py

```
while ▪▪▪▪▪▪▪▪▪▪:
    if ▪▪▪▪▪▪▪▪▪▪▪▪▪:
        print('▪▪▪▪▪')
    else:
        ▪▪▪▪▪▪▪▪

y = sin(0.7)

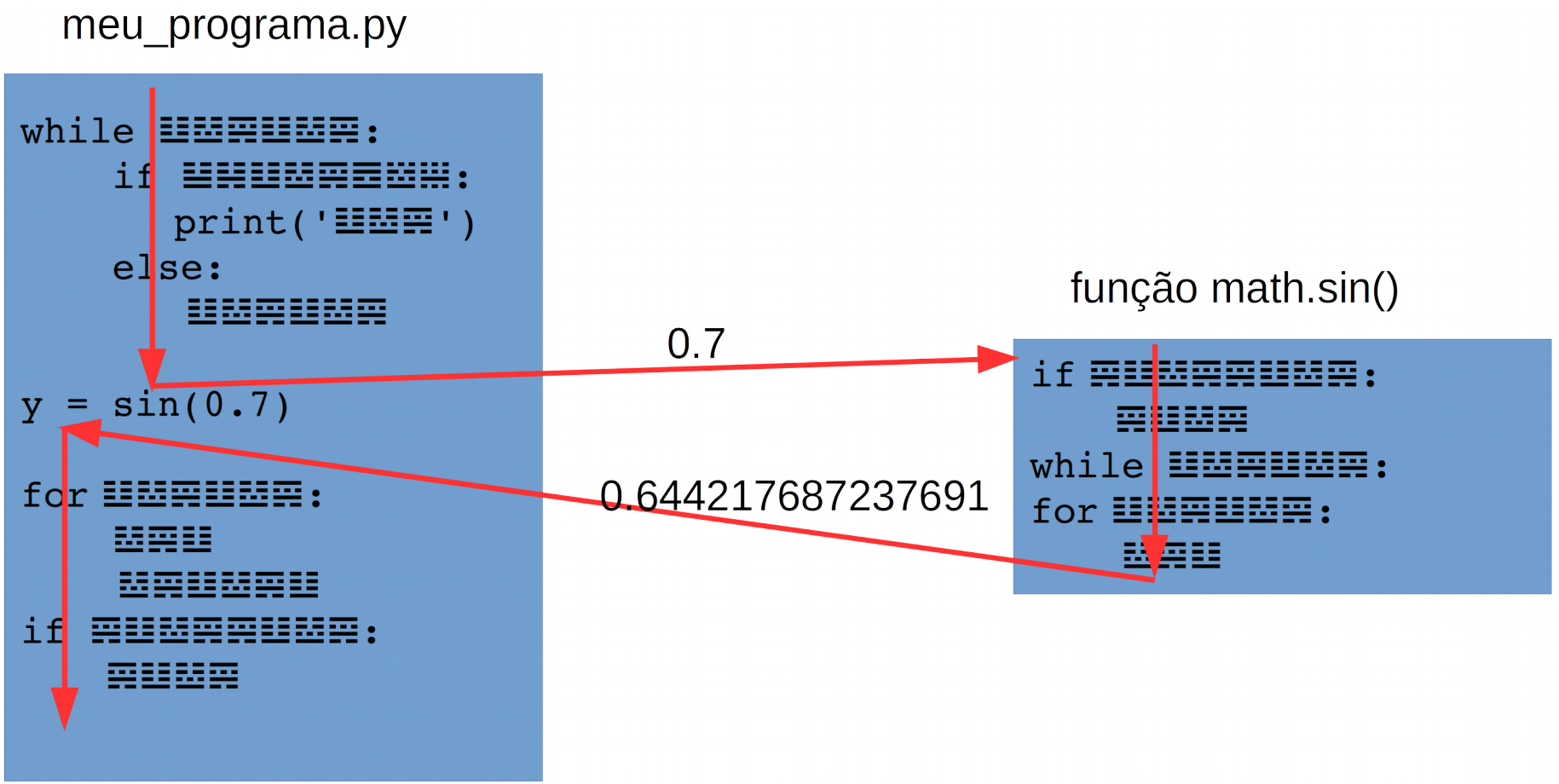
for ▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪
    ▪▪▪▪▪▪▪▪▪
if ▪▪▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪▪
```

0.7

função math.sin()

```
if ▪▪▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪▪
while ▪▪▪▪▪▪▪▪▪▪:
for ▪▪▪▪▪▪▪▪▪▪:
    ▪▪▪▪▪
```

Como funciona



Yes, we can!

- E funções que não existem?
- Podemos defini-las nós mesmos
- Programa estatística
  - ler os dados e armazená-los em uma lista;
  - computar a média;
  - computar a variância;
  - computar mediana; e
  - computar a moda.

Programa estatísticas

```
x = le_dados()
print('Valor de média: {:.4f}'.format(media(x)))
print('Valor de variância: {:.4f}'.format(variancia(x)))
print('Valor do DP: {:.4f}'.format(math.sqrt(variancia(x))))
print('Valor de mediana: {:.4f}'.format(mediana(x)))
print('Valor de moda: {}'.format(moda(x)))
```

ICMC USP SÃO CARLOS SSC

Começando

```
def le_dados():
```

```
x = le_dados()
Função retorna uma lista
```

Começando

```
def le_dados():
```

Indica a definição de uma função

Começando

```
def le_dados():
```

Indica o nome da função

Começando

```
def le_dados():
```

Os comandos que estiverem aqui serão executados quando a função for chamada

Função `le_dados`

```
def le_dados():
    dados = []
    r = 0
    i = 1
    while r >= 0:
        r = float(input('Digite o valor {}: '.format(i)))
        if r < 0:
            print('Entrada de dados terminou')
        else:
            dados.append(r)
        i += 1
```

Função `le_dados`

```
def le_dados():
    dados = []
    r = 0
    i = 1
    while r >= 0:
        r = float(input('Digite o valor {}: '.format(i)))
        if r < 0:
            print('Entrada de dados terminou')
        else:
            dados.append(r)
        i += 1
```

```
x = le_dados()
Função retorna uma lista
O que falta?
```

Função `le_dados`

```
def le_dados():
    dados = []
    r = 0
    i = 1
    while r >= 0:
        r = float(input('Digite o valor {}: '.format(i)))
        if r < 0:
            print('Entrada de dados terminou')
        else:
            dados.append(r)
        i += 1
    return dados
```

```
x = le_dados()
Função retorna uma lista
O que falta?
```

Indicar qual é o resultado da função

Observações importantes

- Todas as variáveis utilizadas na função não fazem sentido fora dela. (locais)
- Função le_dados() não requer nenhum parâmetros
  - Comparar com a função math.sin

Função média

```
def media(dados):
```

Essa função recebe um parâmetro. No caso, é a lista sobre a qual deve ser computada a média.

Função média

```
def media(dados):
```

Essa função recebe um parâmetro. No caso, é a lista sobre a qual deve ser computada a média.

Função média

```
def media(dados):
```

Essa função recebe um parâmetro. No caso, é a lista sobre a qual deve ser computada a média.

Esse nome de parâmetro não tem nada a ver com o nome usado na função le_dados. Poderia ser x, y, z a, b, conjunto_de_dados...

Função média

```
def media(dados):
    vamos tentar
```

Essa função recebe um parâmetro. No caso, é a lista sobre a qual deve ser computada a média.

Esse nome de parâmetro não tem nada a ver com o nome usado na função le_dados. Poderia ser x, y, z a, b, conjunto_de_dados...

Função média

```
def media(dados):
    soma = 0.0
    for r in dados:
        soma += r
    return soma / len(dados)
```

Função média

```
def media(dados):
    soma = 0.0
    for r in dados:
        soma += r
    return soma / len(dados)
```

Resultado é um float

Exercício

- Implemente as demais funções do nosso programa

```
x = le_dados()

print('Valor de média: {:.4f}'.format(media(x)))

print('Valor de variância: {:.4f}'.format(variancia(x)))

print('Valor do DP: {:.4f}'.format(math.sqrt(variancia(x))))

print('Valor de mediana: {:.4f}'.format(mediana(x)))

print('Valor de moda: {}'.format(moda(x)))
```

variância

```
def variancia(dados):
    m = media(dados)
    var = 0.0
    for r in dados:
        var += (r - m) ** 2
    return var / (len(dados) -1)
```

Retornando tuplas

- Com o comando return é possível retornar mais do que um valor
- Para isso usamos tuplas
- Por exemplo, retornar variância de desvio padrão

Função variância (V 2)

```
def variancia(dados):
    m = media(dados)
    var = 0.0
    for r in dados:
        var += (r - m) ** 2
    var /= (len(dados) -1)
    dp = math.sqrt(var)
    return  (var,dp)
```

Função variância (V 2)

```
def variancia(dados):
    m = media(dados)
    var = 0.0
    for r in dados:
        var += (r - m) ** 2
    var /= (len(dados) -1)
    dp = math.sqrt(var)
    return (var,dp)
```

Uma tupla com 2 valores

Programa V2

```
x = le_dados()

print('Valor de média: {:.4f}'.format(media(x)))

(v,dp) = variancia(x)

print('Valor de variância: {:.4f}'.format(v))

print('Valor do DP: {:.4f}'.format(dp))

print('Valor de mediana: {:.4f}'.format(mediana(x)))

print('Valor de moda: {}'.format(moda(x)))
```

Função sem valor

- Às vezes queremos uma função que “faça alguma coisa” mas não retorne nenhum valor
- Por exemplo, mostrar os valores da moda dos dados
- ```
  print('Valor de mediana: {:.4f}'.format(mediana(x)))
  mostra_moda(moda(x))
  ```
  ```

Função sem valor

```
def mostra_moda(moda):
    s = 'Os valores da moda são: '
    for x in moda:
        s += '{:.4f} '.format(x)
    print(s)
    return
```

Função sem valor

```
def mostra_moda(moda):
    s = 'Os valores da moda são: '
    for x in moda:
        s += '{:.4f} '.format(x)
    print(s)
    return
```

return no final é opcional

Função sem valor

```
def mostra_moda(moda):
    s = 'Os valores da moda são: '
    for x in moda:
        s += '{:.4f} '.format(x)
    print(s)
    return
```

k = mostra_moda(moda(x))

ERRO!!!!!

return no final é opcional

return anywhere

- Podemos usar o return em qualquer ponto
- Termina a execução da função

```
def le_dados():
    dados = []
    r = 0
    i = 1
    while r >= 0:
        r = float(input('Digite o valor {}: '.format(i)))
        if r < 0:
            print('Entrada de dados terminou')
            return dados
        else:
            dados.append(r)
        i += 1
```

Exercício

- Escreva uma função que recebe três parâmetros que representam os lados de um triângulo. O programa deve retornar: 0 se os valores não correspondem a um triângulo; 1 se correspondem a um triângulo equilátero; 2 se correspondem a um triângulo isósceles ou 3 se correspondem a um triângulo escaleno.

Resposta

```
def tritip(a,b,c):
    if a + b <= c or a + c <= b or b + c <= a:
        return 0
    if a == b and b == c:
       return 1
    if a == b or a == c or b == c:
        return 2
    return 3
```

Recursividade

- Vimos que uma função pode chamar outra
- Uma função pode chama ela mesma
- Faz sentido?

Fatorial

- Qual o valor de n! ?
  - a multiplicação de todos os inteiros de 1 até n
-
```
def fatorial(n):
    for k in range(1,n+1):
        f *= k
    return f
```

Ou…

Fatorial

- Qual o valor de n! ?
  - a multiplicação de todos os inteiros de 1 até n
  - n * (n-1)! . Além disso, 0! = 1
- 4! = 4 * 3! = 4 * 3 * 2! = 4 * 3 * 2 * 1! = 4 * 3 * 2 * 1 * 0! = 4 * 3 * 2 * 1 * 1 = 24
- Podemos usar essa definição para escrever uma função que computa o fatorial?

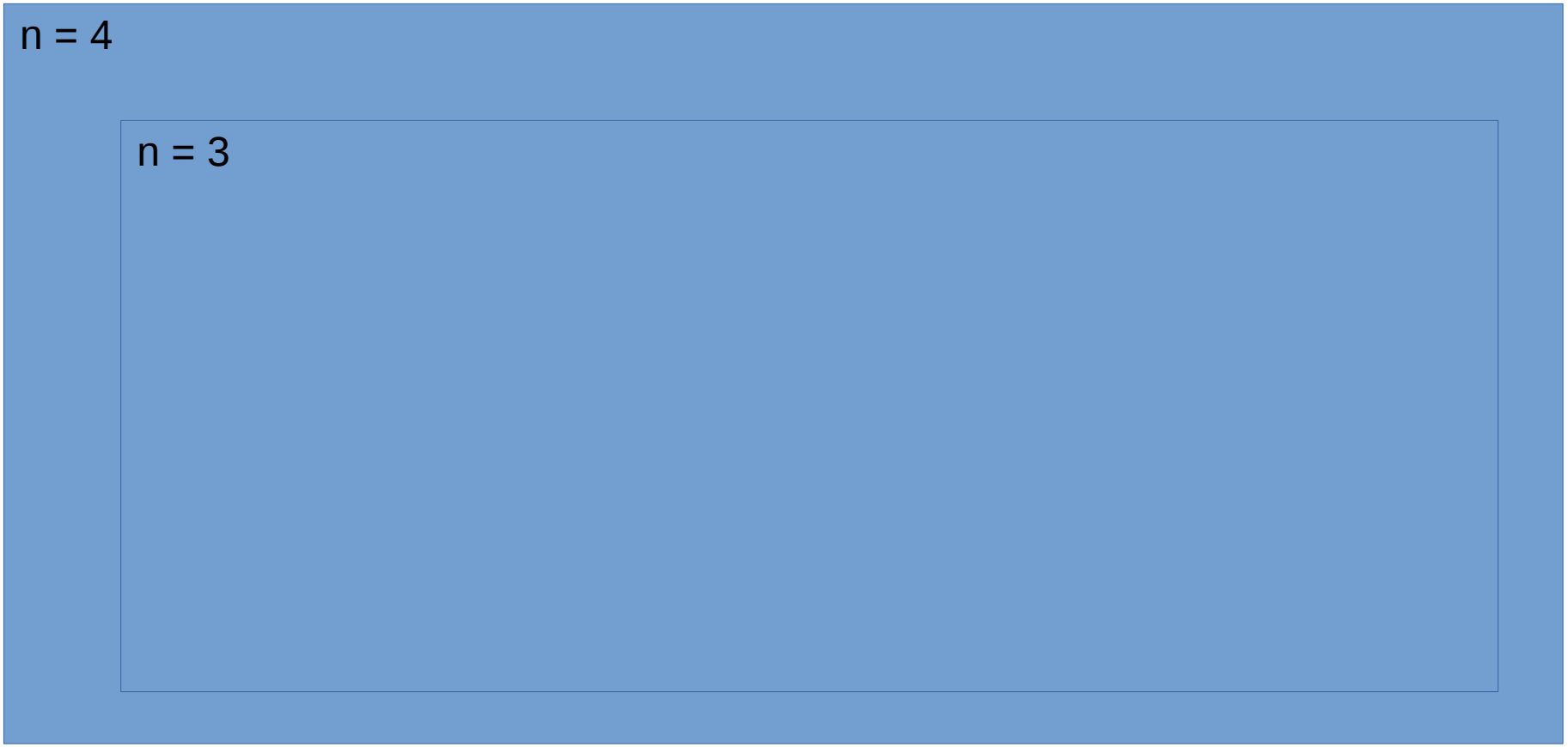
Fatorial

- Qual o valor de n! ?
  - a multiplicação de todos os inteiros de 1 até n
  - n * (n-1)! . Além disso, 0! = 1
- 4! = 4 * 3! = 4 * 3 * 2! = 4 * 3 * 2 * 1! = 4 * 3 * 2 * 1 * 0! = 4 * 3 * 2 * 1 * 1 = 24
-
```
def fatorial(n):
    if n == 0:
        return 1
    return n * fatorial(n-1)
```

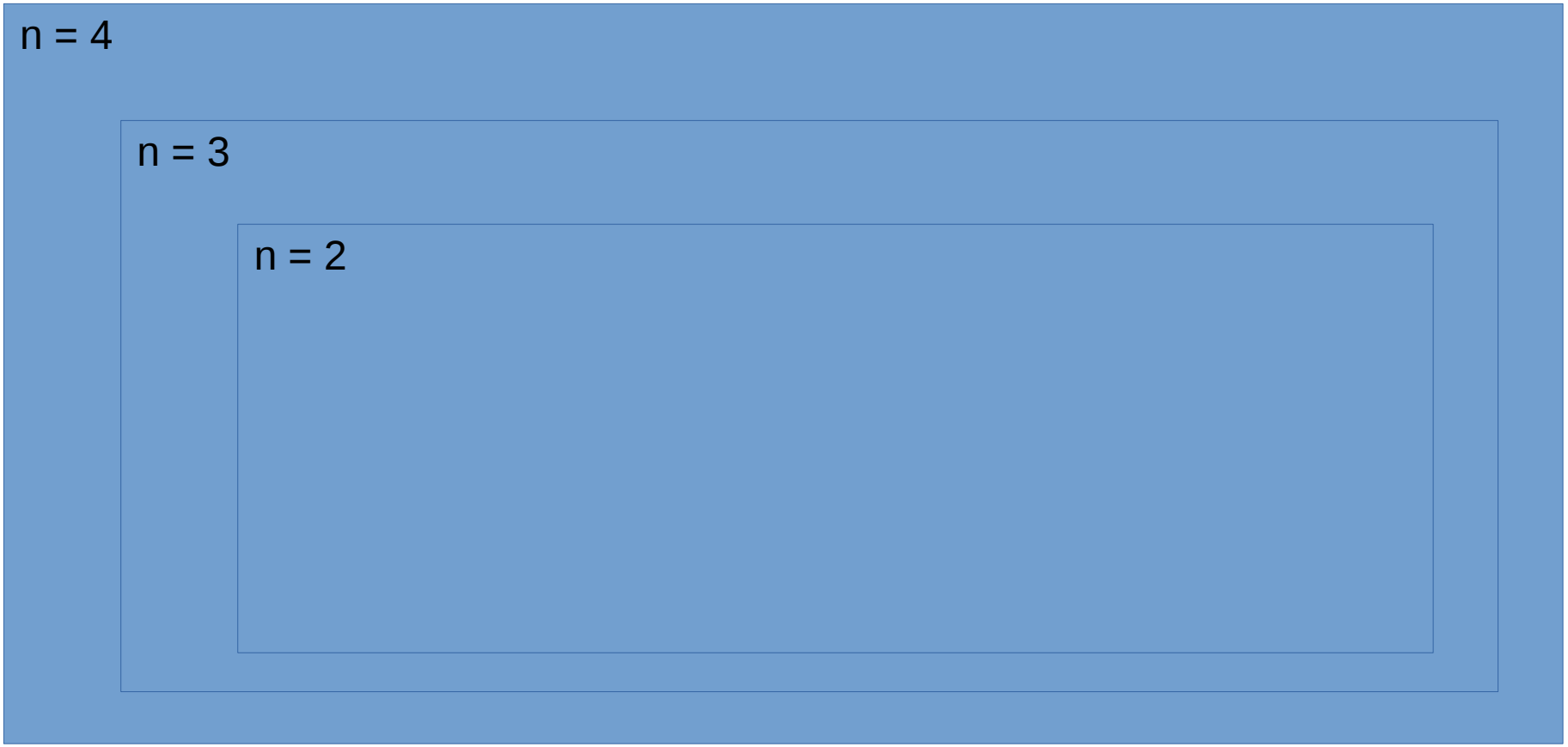
O que acontece?

n = 4

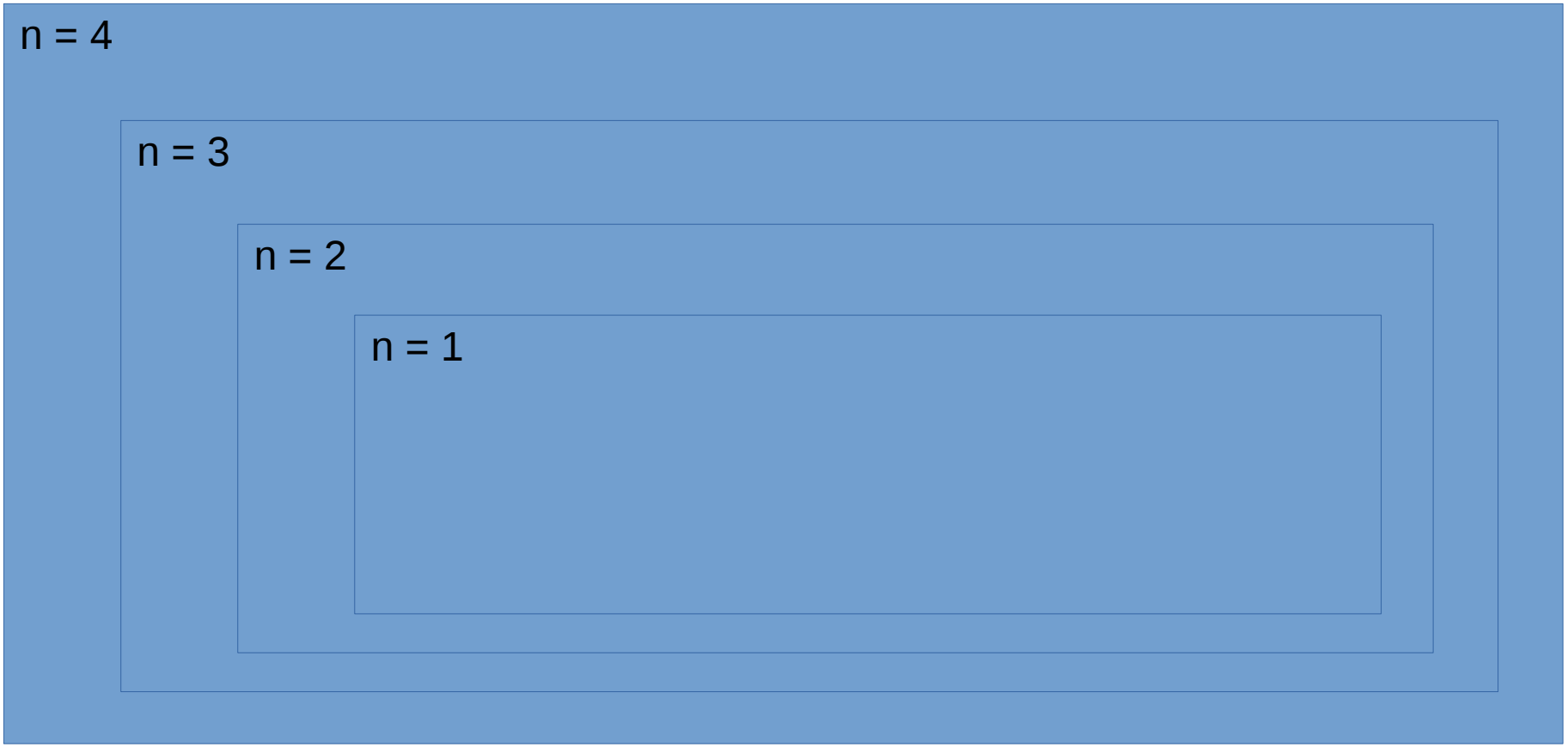
O que acontece?



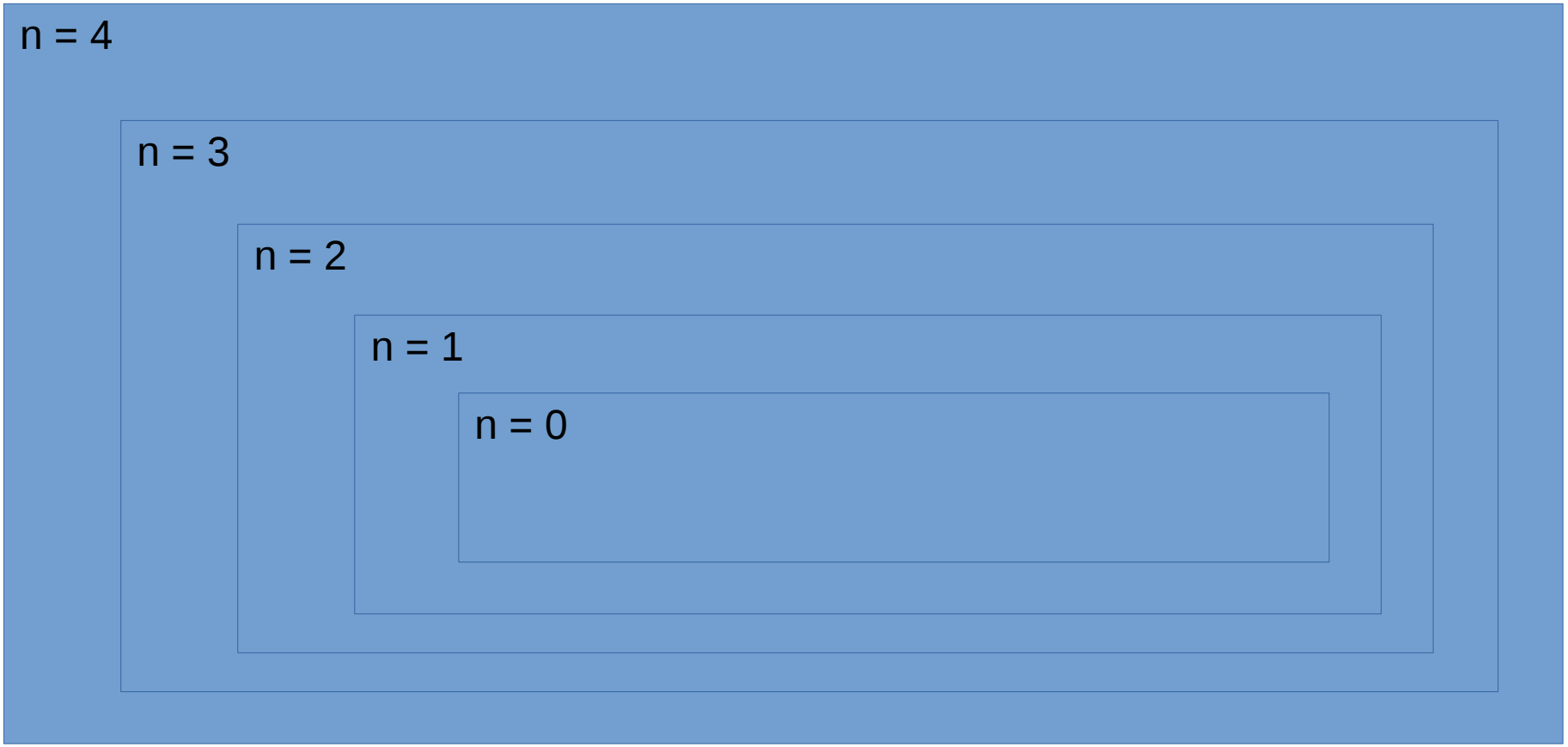
O que acontece?



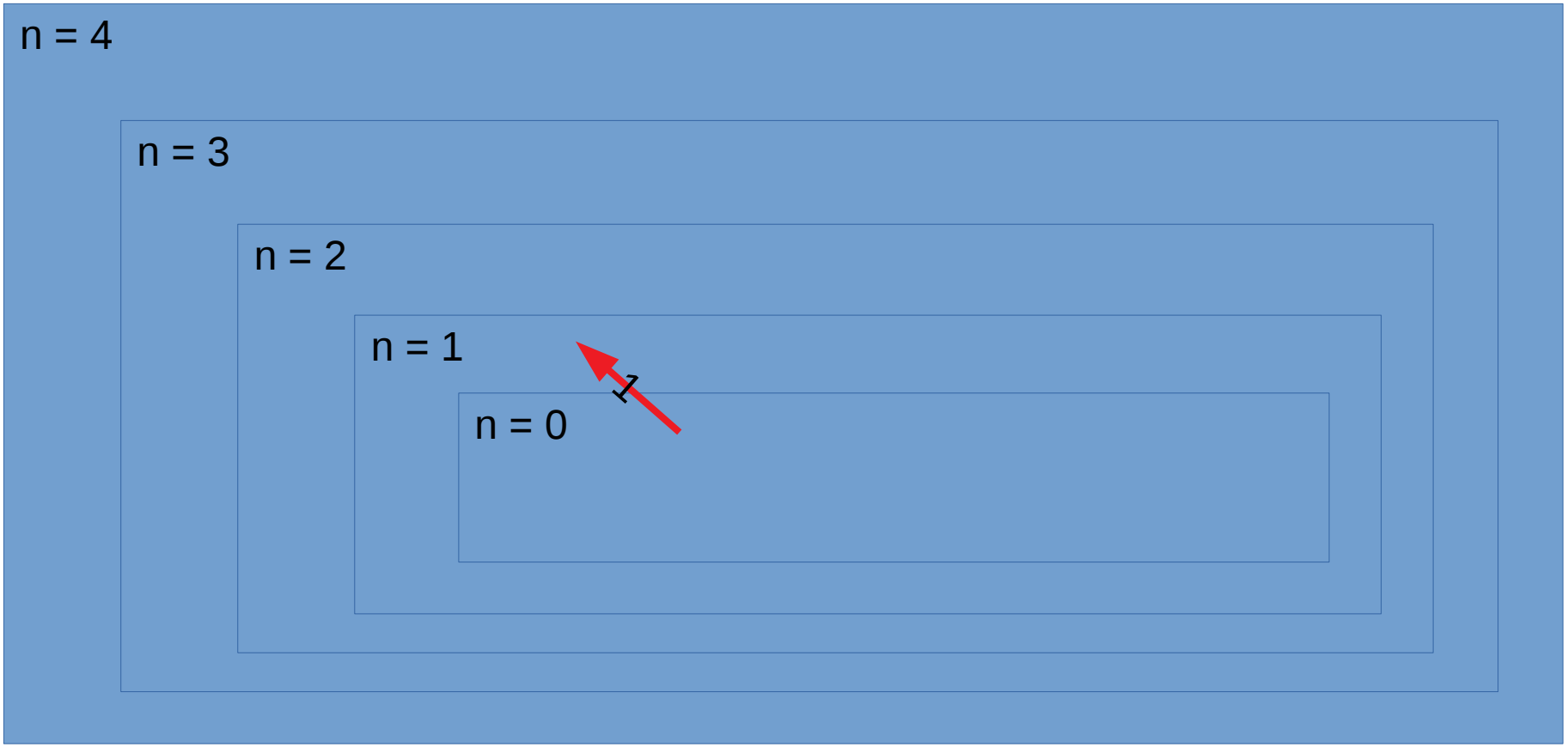
O que acontece?



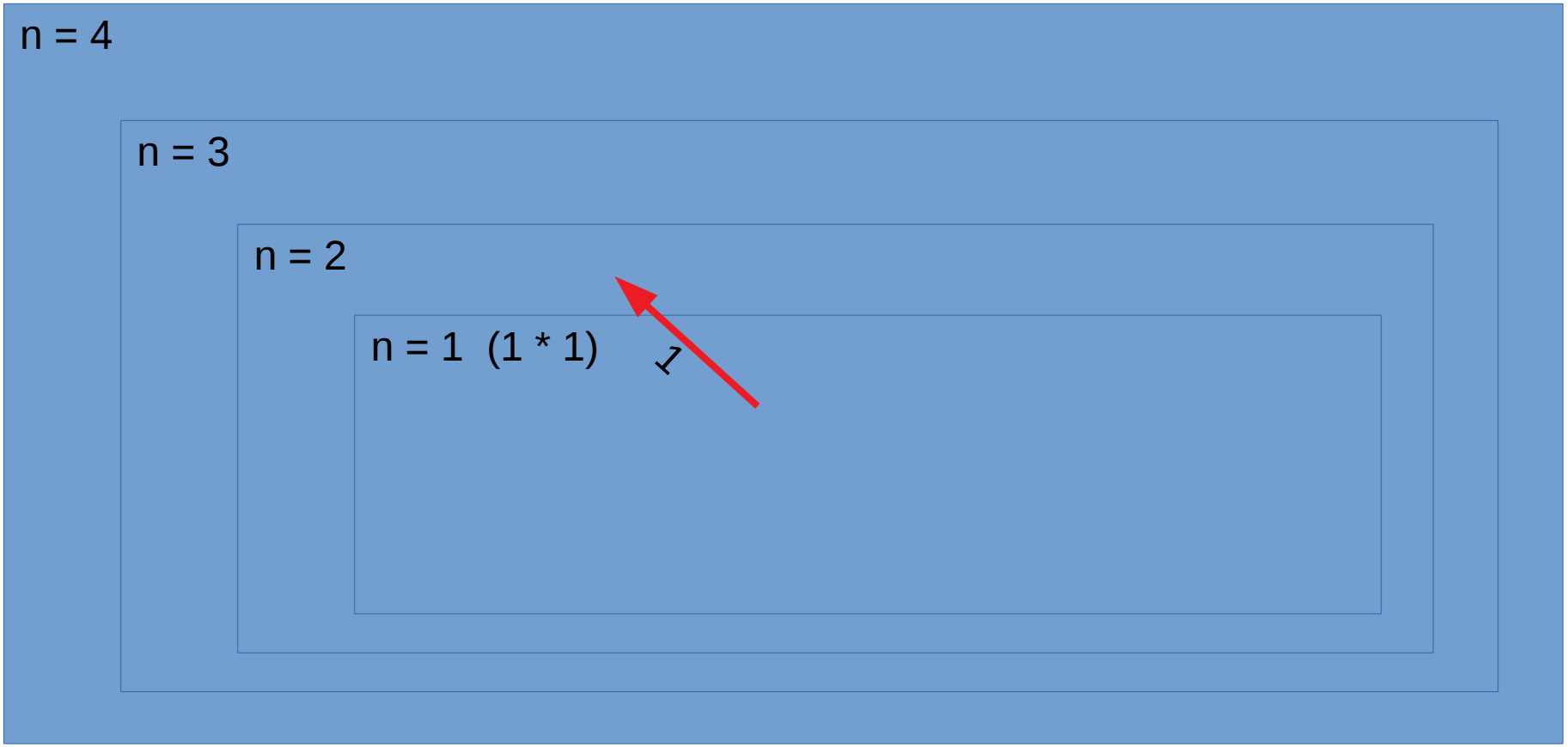
O que acontece?



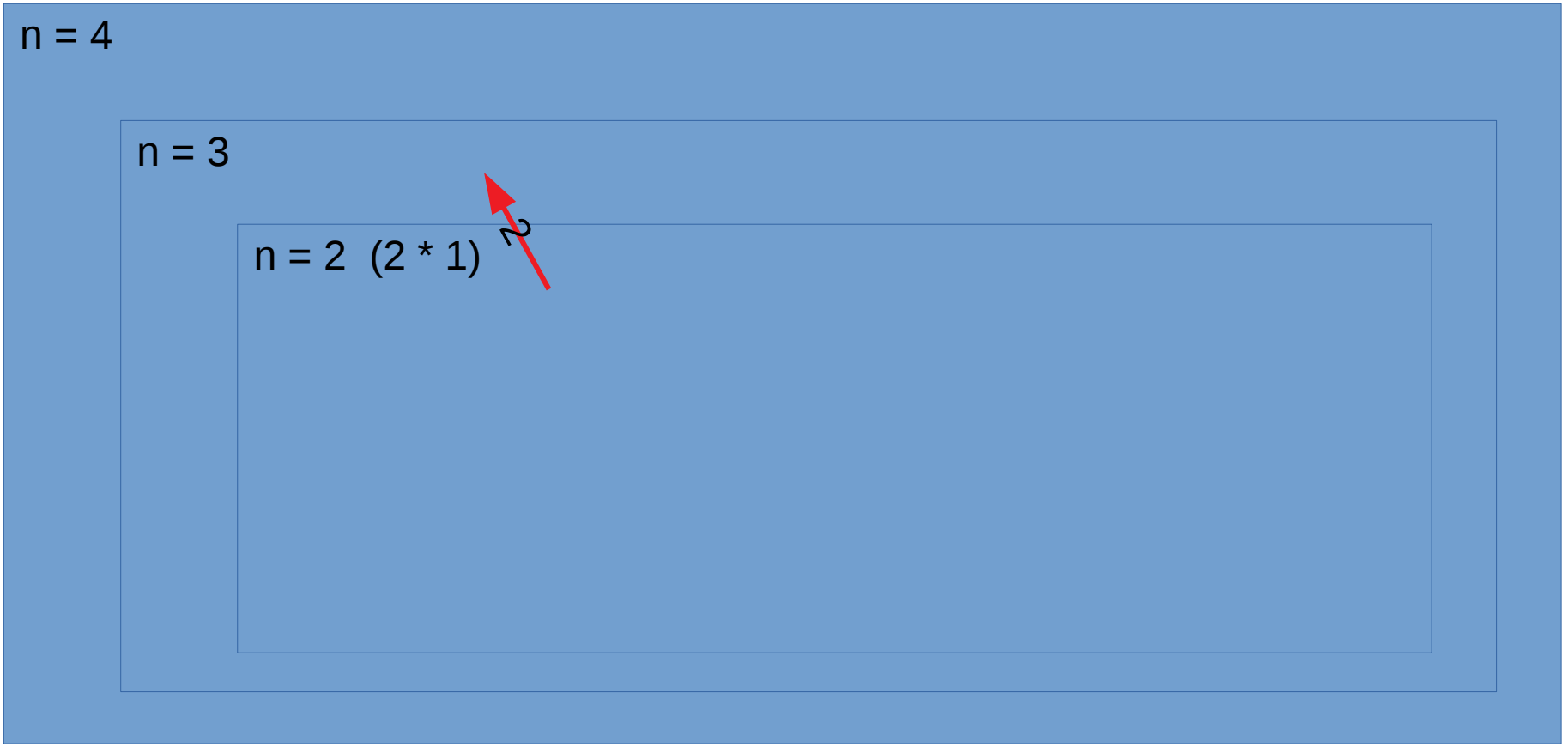
O que acontece?



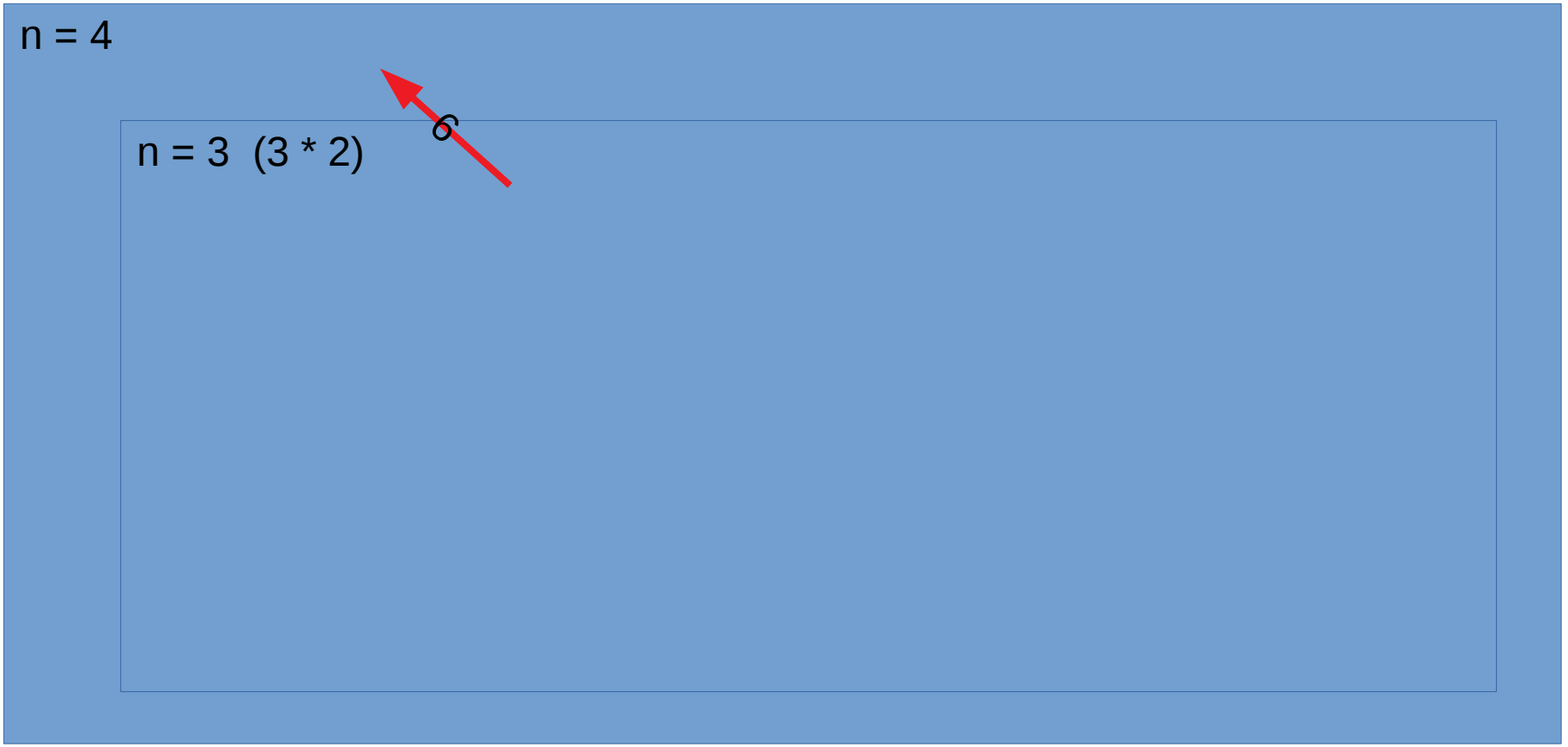
O que acontece?



O que acontece?



O que acontece?



O que acontece?

Outro exemplo

- MDC(x, x) = x;
- MDC(x, y) = MDC(x − y, y), se x > y;
- MDC(x, y) = MDC(y, x).
- Como implementar?

MDC

```
def mdc(x,y):
    if x == y:
        return x
    if x > y:
        return mdc(x-y, y)
    return mdc(y,x)
```

Advertência

- Cada chamada de função, consome memória
- O interpretador vai limitar o número de chamadas

Advertência

```
> python mdc.py
Entre com o 1o. valor: 1234567
Entre com o 2o. valor: 890
Traceback (most recent call last):
   File "mdc.py", line 13, in <module>
   print('O valor do MDC é: '.format(mdc(x,y)))
   File "mdc.py", line 6, in mdc
   return mdc(x-y, y)
   [Previous line repeated 994 more times]
   File "mdc.py", line 3, in mdc
   if x == y:
RecursionError: maximum recursion depth exceeded in comparison
```

Praticando

- 1. Crie uma função recursiva que receba um número inteiro N e calcule a soma dos números de 1 até N.

- 2. Escreva uma função recursiva para somar os elementos de uma lista de números. Ou seja, a função recebe uma lista como parâmetro e retorna um número, que é a soma dos elementos da lista.