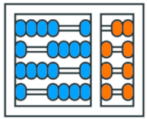


Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

UNICAMP

MC102 - Aula 20

Exemplos sobre Recursão

Algoritmos e Programação de Computadores

Turmas

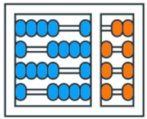
OVXZ

Prof. Lise R. R. Navarrete

lrommel@ic.unicamp.br

Quinta-feira, 02 de junho de 2022

19:00h - 21:00h (CB06)



Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

UNICAMP

MC102 – Algoritmos e Programação de Computadores

Turmas

OVXZ

https://ic.unicamp.br/~mc102/

Site da Coordenação de MC102

Aulas teóricas:

Terça-feira, 21:00h - 23:00h (CB06)

Quinta-feira, 19:00h - 21:00h (CB06)

Conteúdo

- Exemplo 1
- Exemplo 2
- Exemplo 3
- Exemplo 4
- Exemplo 5
- Exemplo 6
- Exemplo 7
- Exemplo 8

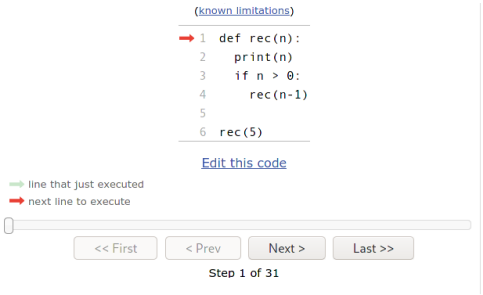
Exemplo 1

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

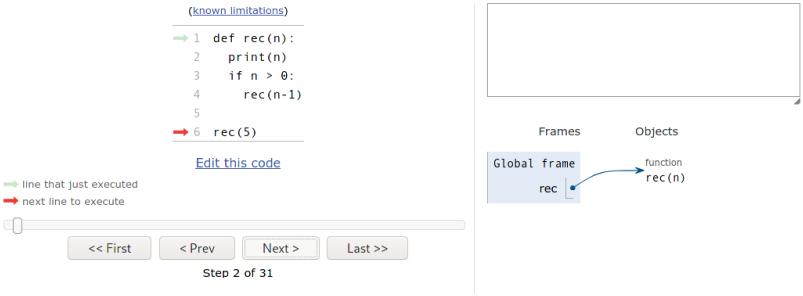
- Dada a seguinte função e chamada recursiva:

```
1 def rec(n):
2   print(n)
3   if n > 0:
4     rec(n-1)
5
6 rec(5)
```

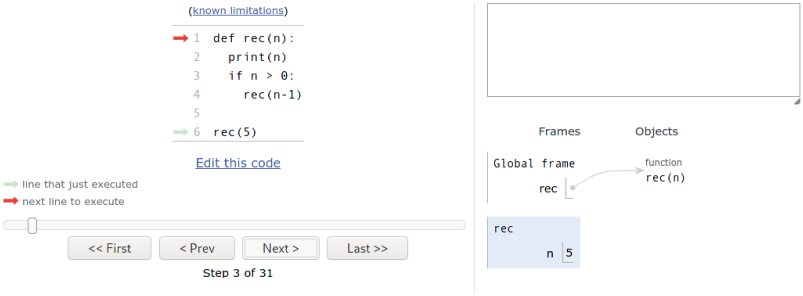
- O que a função imprime como resposta?



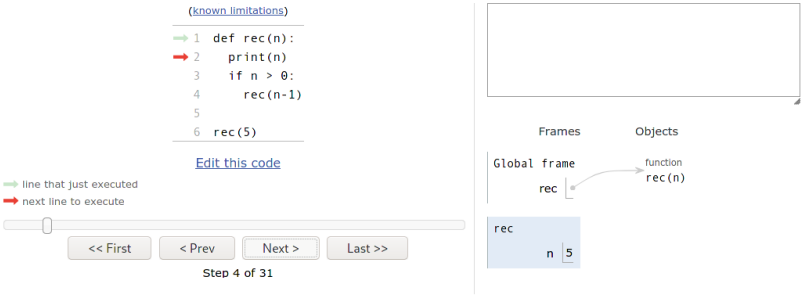
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



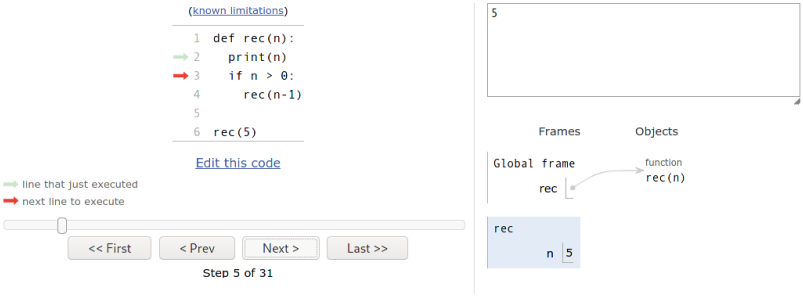
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



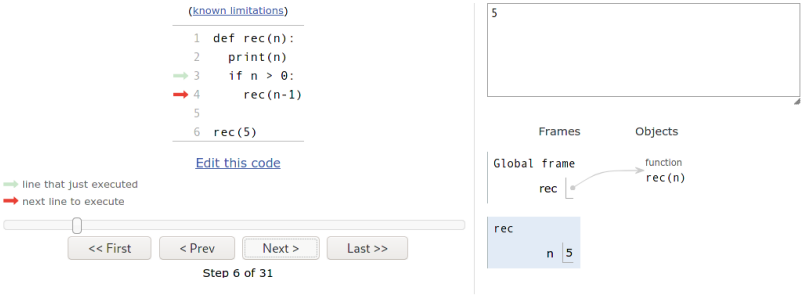
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)



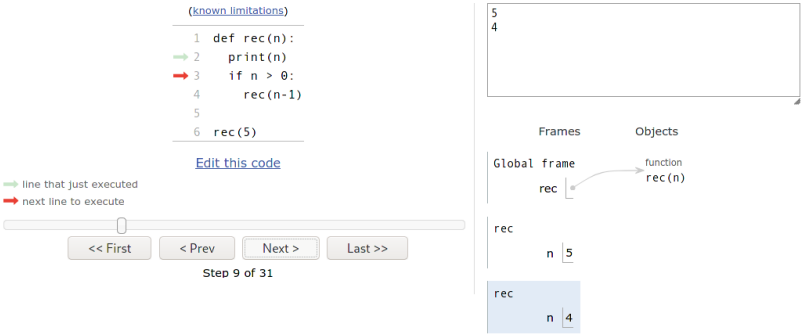
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)

https://pythontutor.com (http://tinyurl.com/pmb52mz5)

https://pythontutor.com (http://tinyurl.com/pmb52mz5)



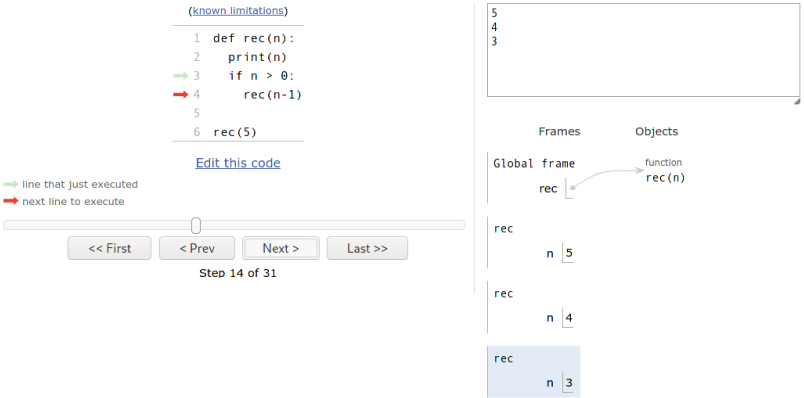
https://pythontutor.com (http://tinyurl.com/pmb52mz5)

https://pythontutor.com (http://tinyurl.com/pmb52mz5)

https://pythontutor.com (http://tinyurl.com/pmb52mz5)

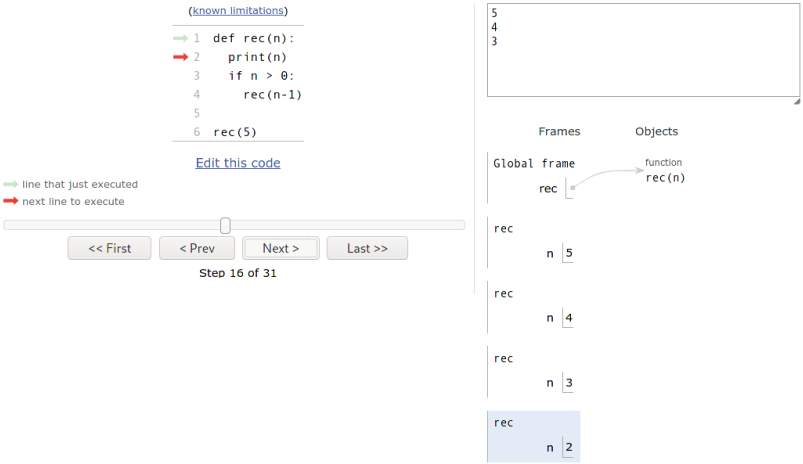
https://pythontutor.com (http://tinyurl.com/pmb52mz5)

https://pythontutor.com (http://tinyurl.com/pmb52mz5)

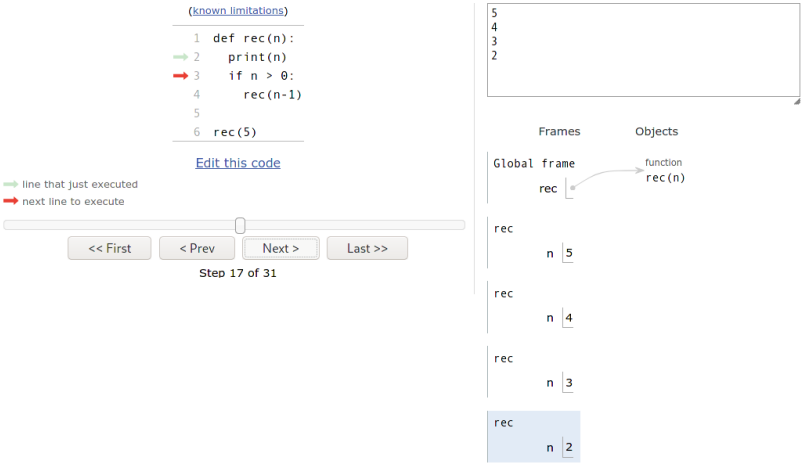


https://pythontutor.com (http://tinyurl.com/pmb52mz5)

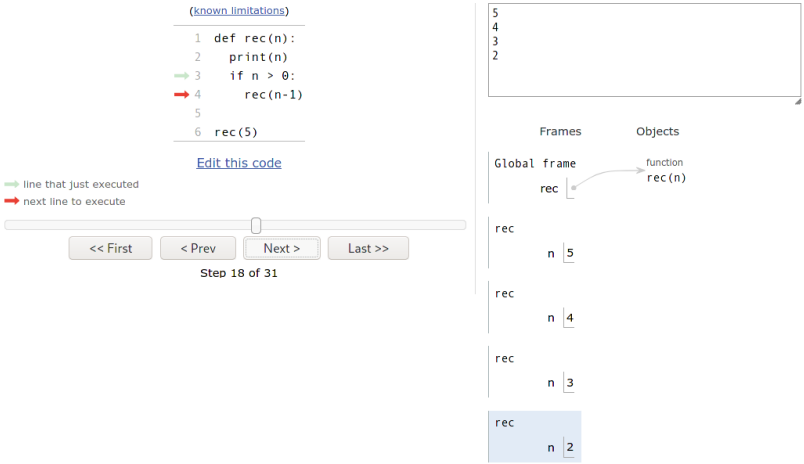
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



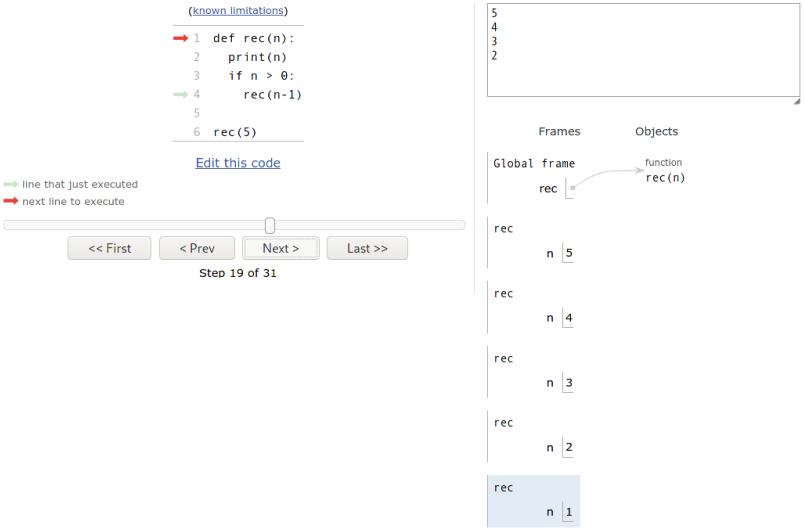
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



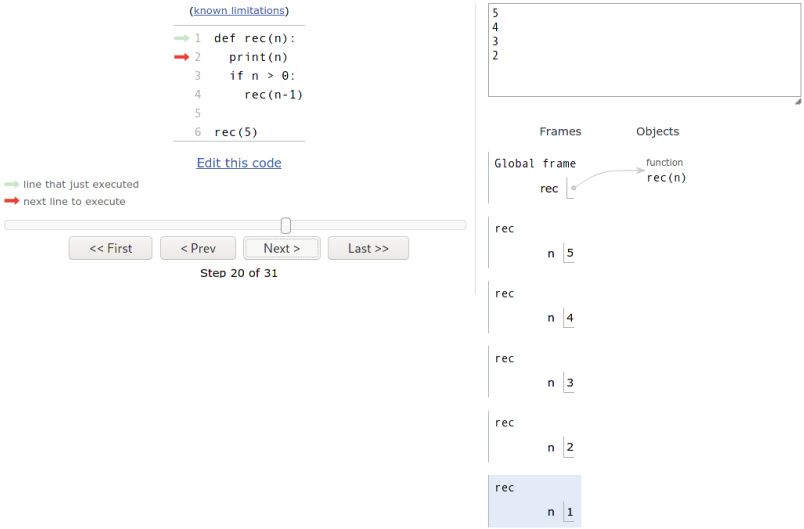
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



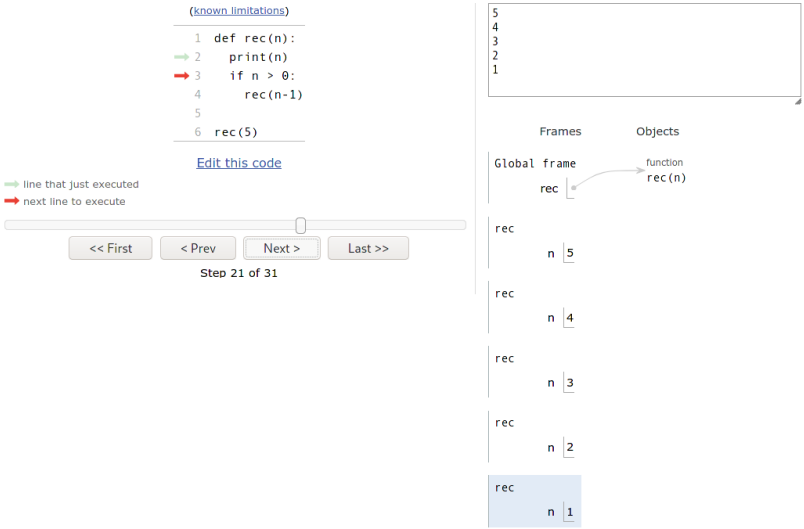
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



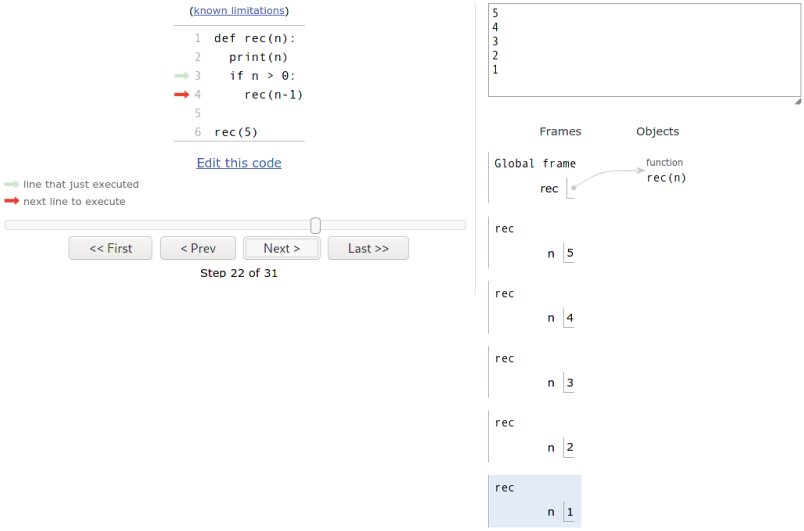
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



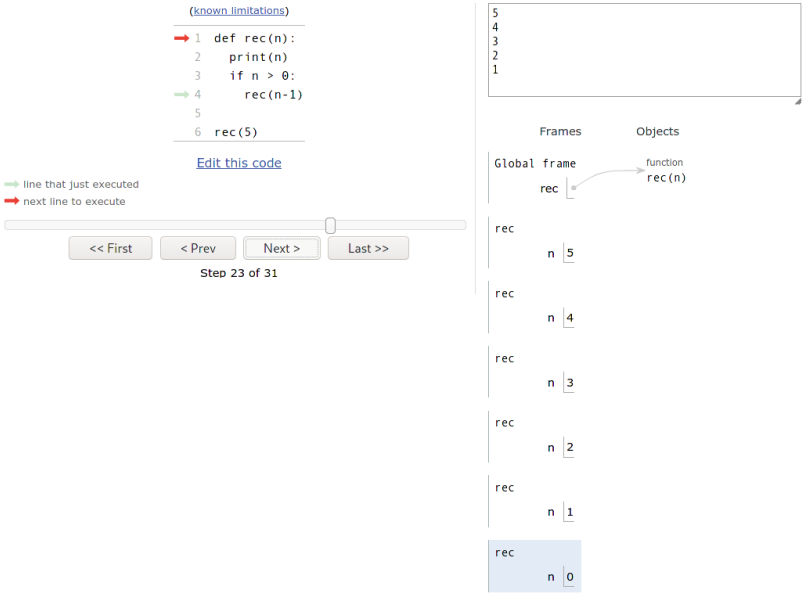
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



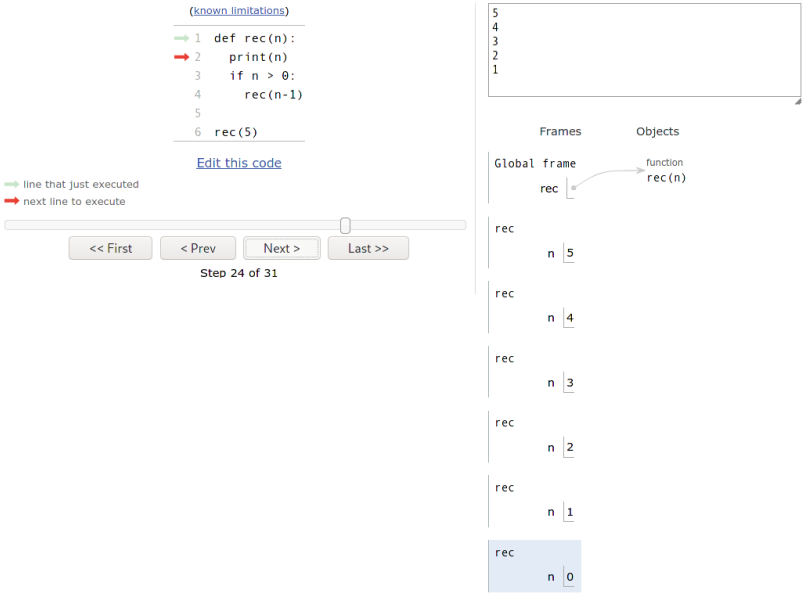
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



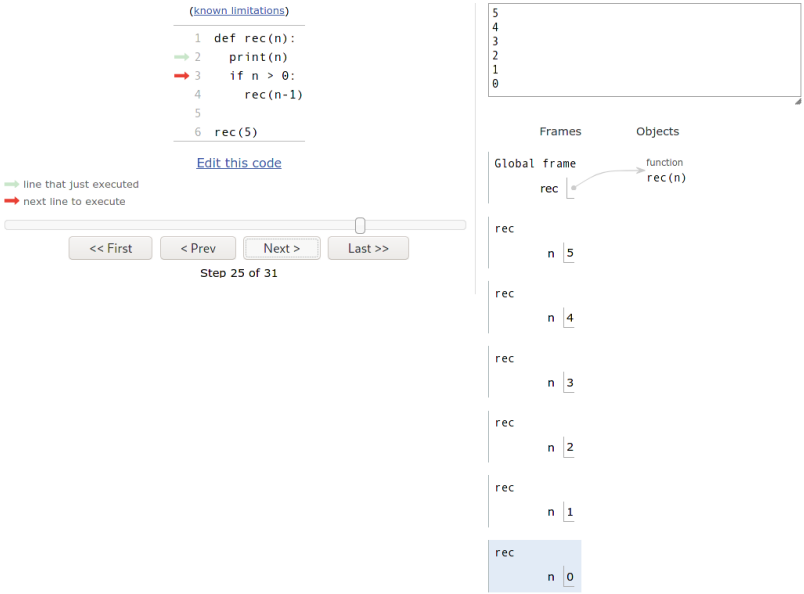
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)

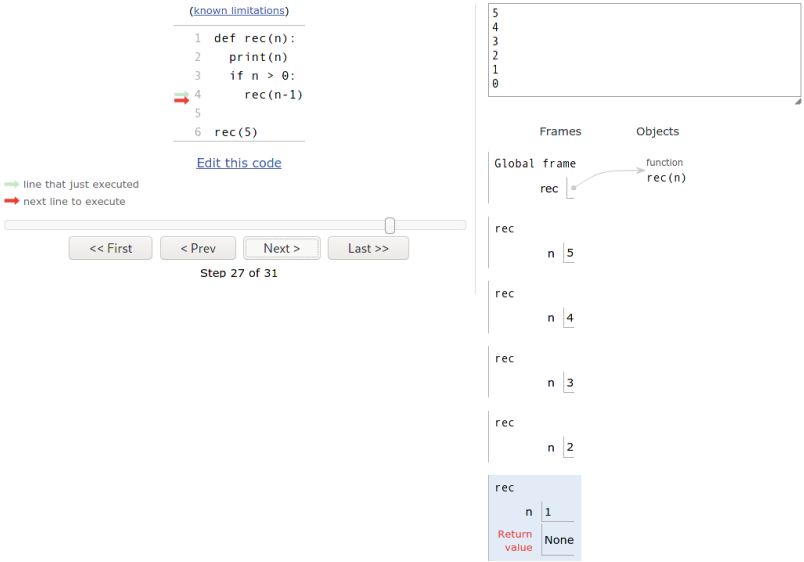


https://pythontutor.com (http://tinyurl.com/pmb52mz5)

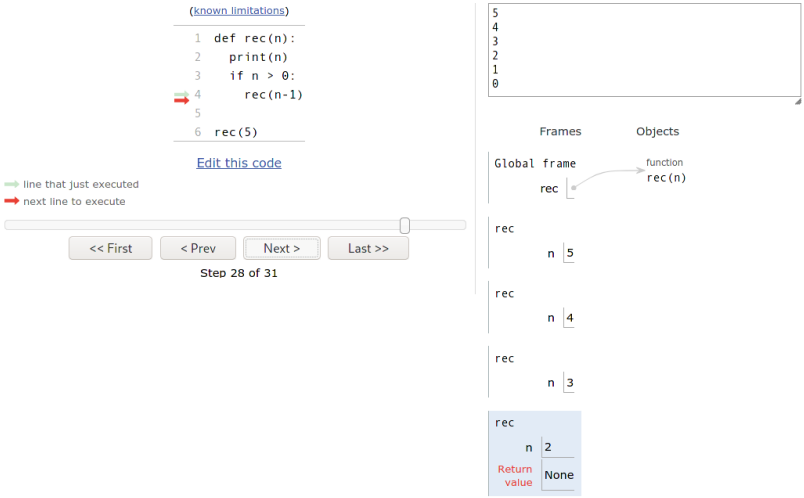


https://pythontutor.com (http://tinyurl.com/pmb52mz5)

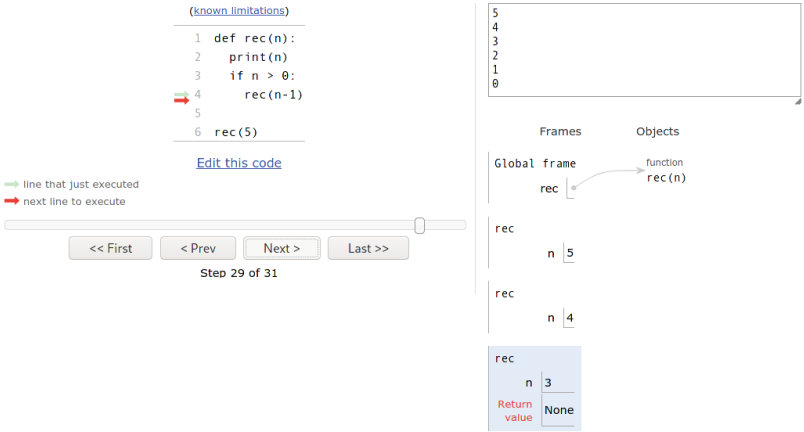
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



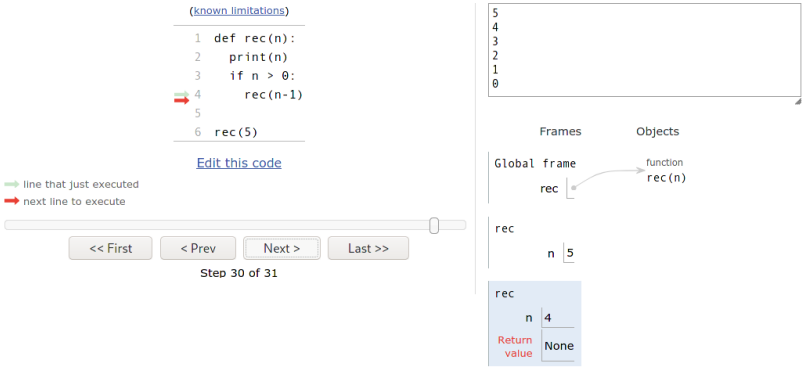
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



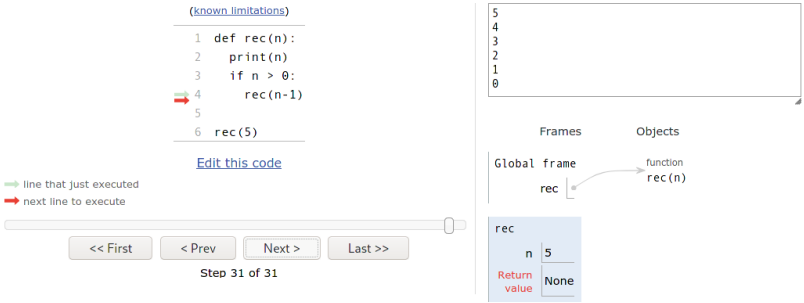
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



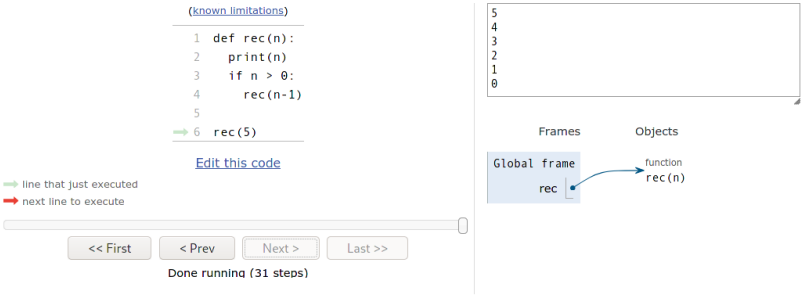
https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)



https://pythontutor.com (http://tinyurl.com/pmb52mz5)

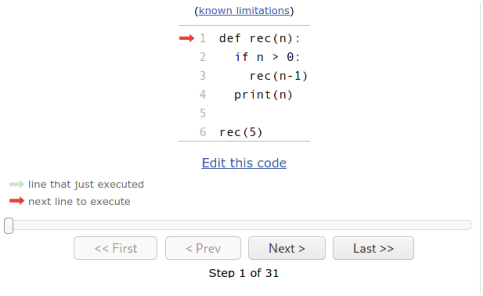
Exemplo 2

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

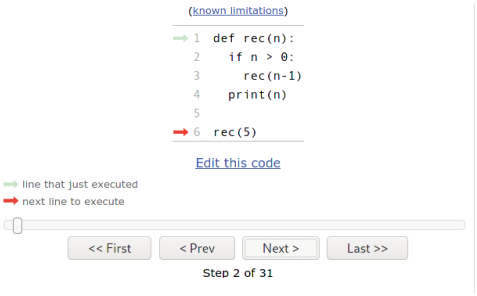
- Mudando a posição do comando de impressão na função.

```
1 def rec(n):
2   if n > 0:
3     rec(n-1)
4   print(n)
5
6 rec(5)
```

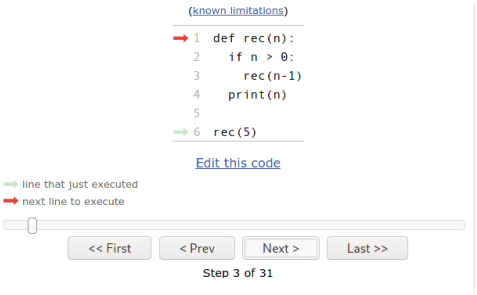
- O que a função imprime como resposta agora?

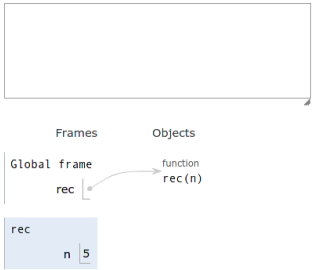


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

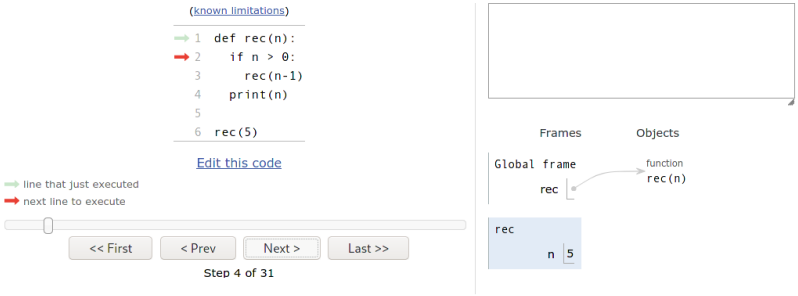


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

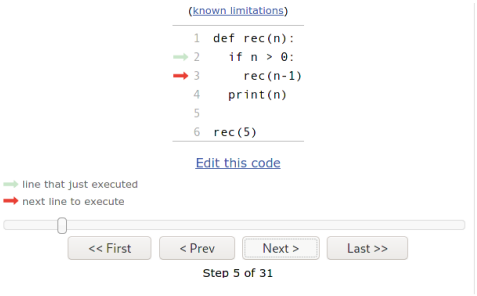


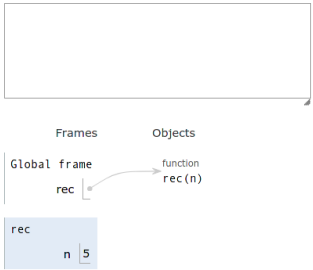


https://pythontutor.com (https://tinyurl.com/5ddj66wj)



https://pythontutor.com (https://tinyurl.com/5ddj66wj)

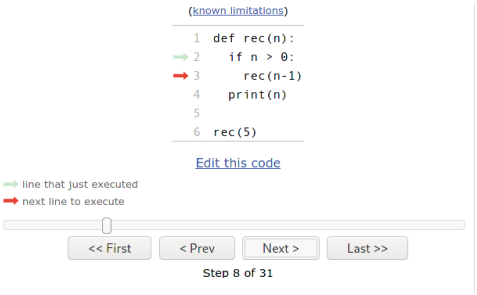


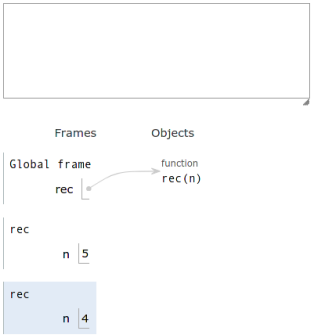


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

https://pythontutor.com (https://tinyurl.com/5ddj66wj)

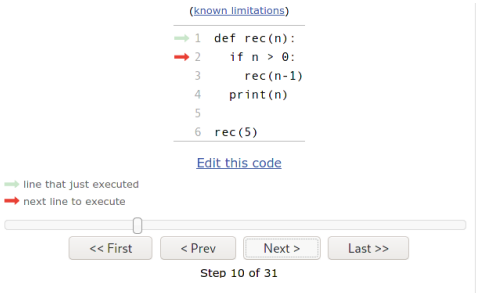
https://pythontutor.com (https://tinyurl.com/5ddj66wj)

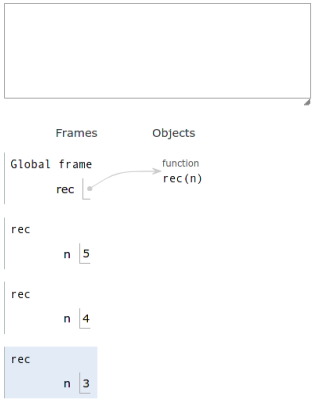




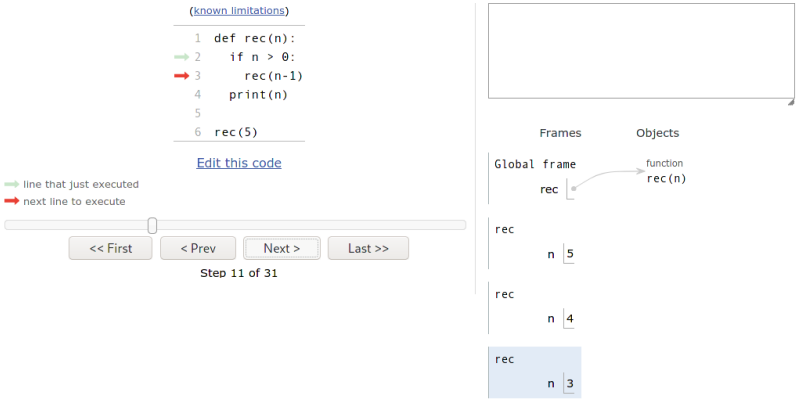
https://pythontutor.com (https://tinyurl.com/5ddj66wj)

https://pythontutor.com (https://tinyurl.com/5ddj66wj)

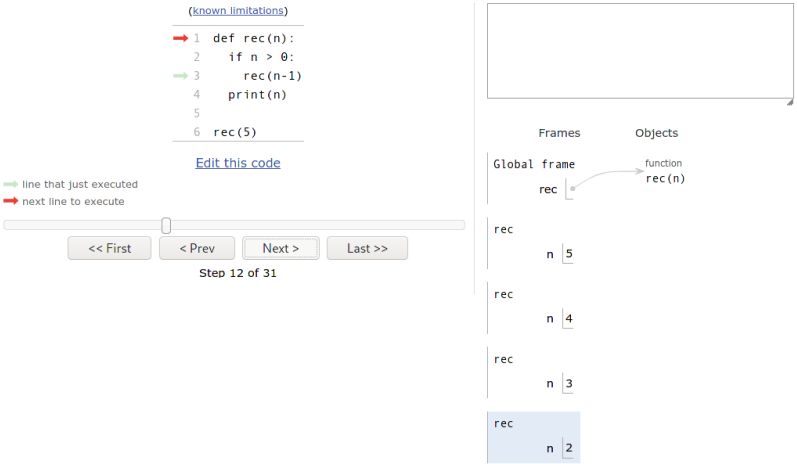




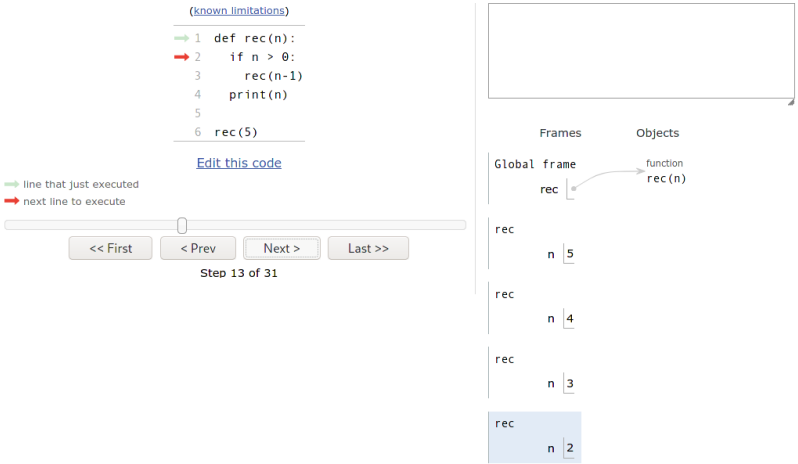
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



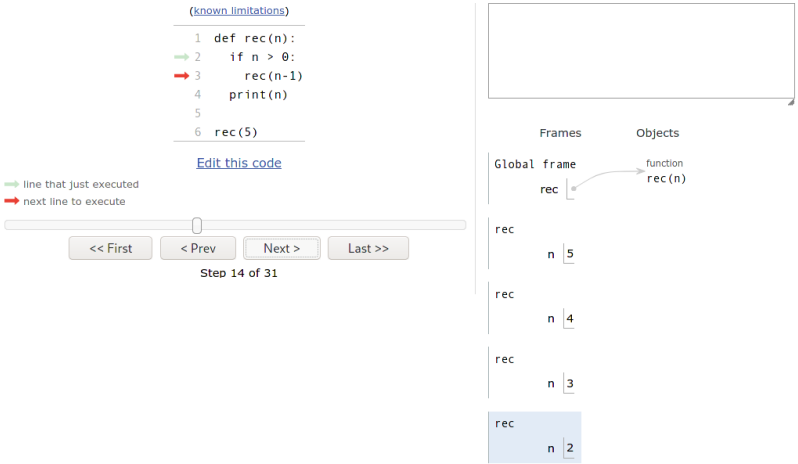
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



https://pythontutor.com (https://tinyurl.com/5ddj66wj)

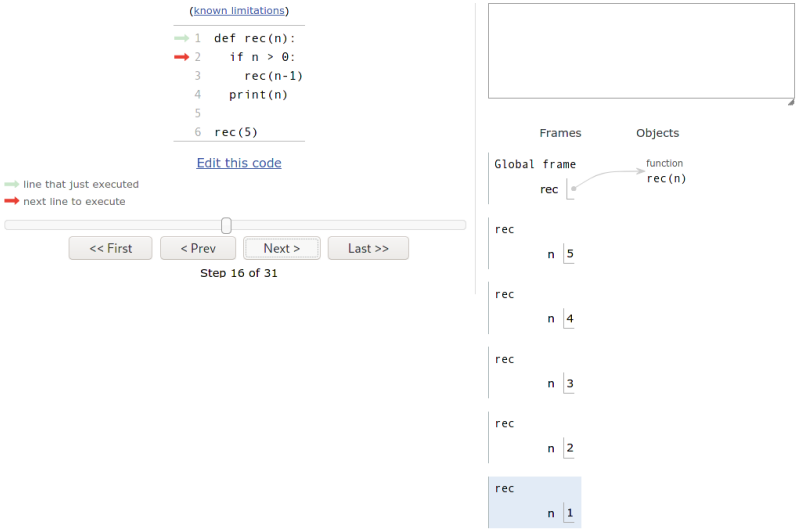


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

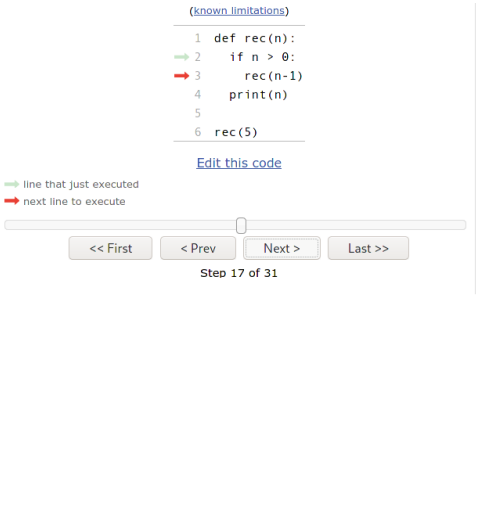


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

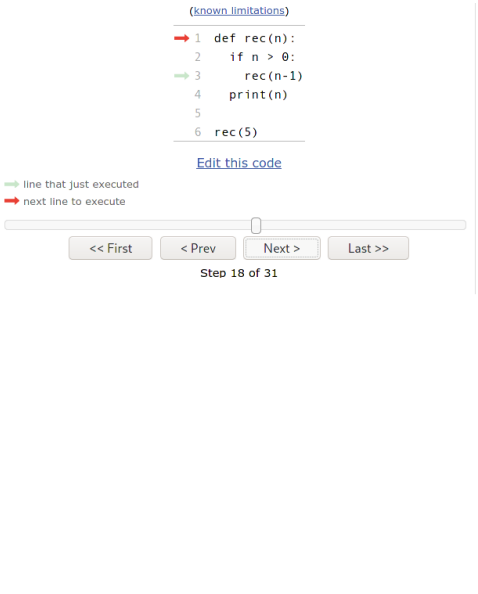
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



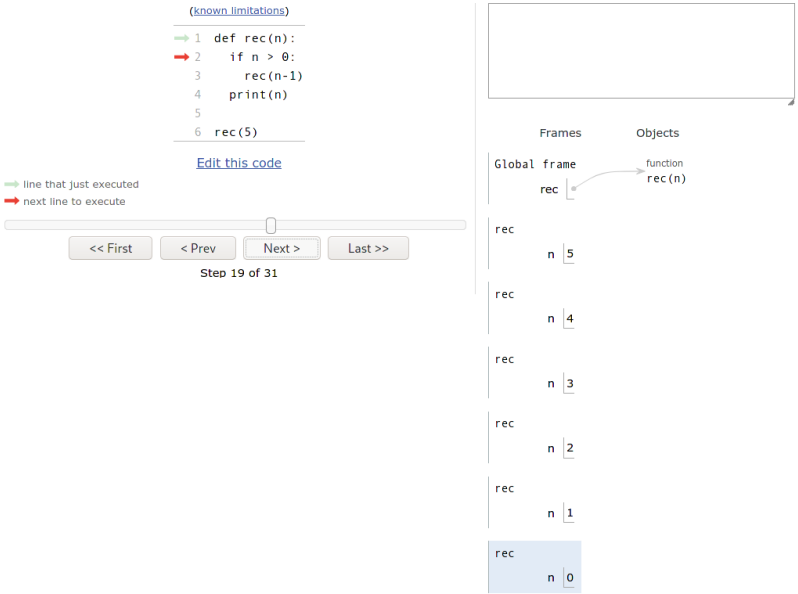
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



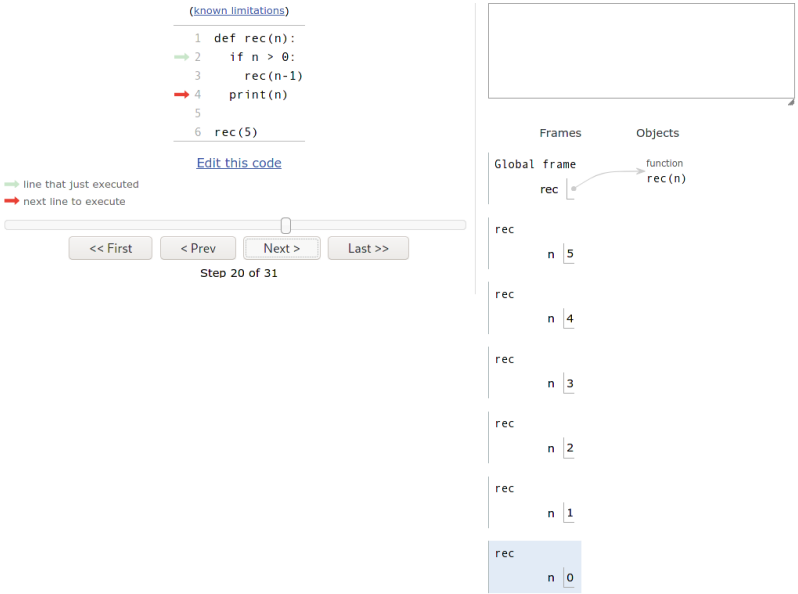
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



https://pythontutor.com (https://tinyurl.com/5ddj66wj)

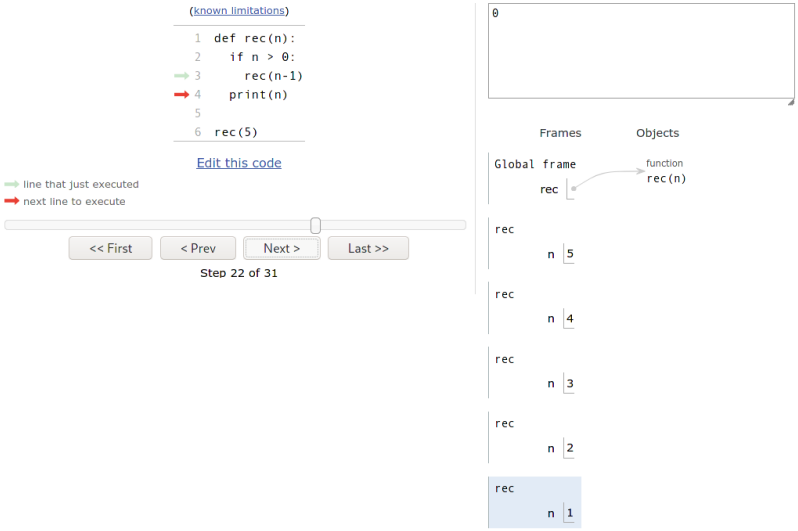


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

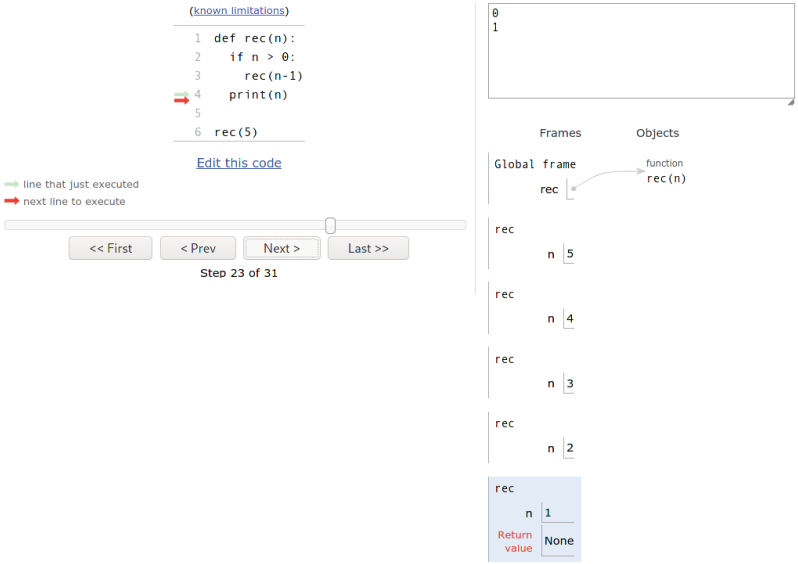


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

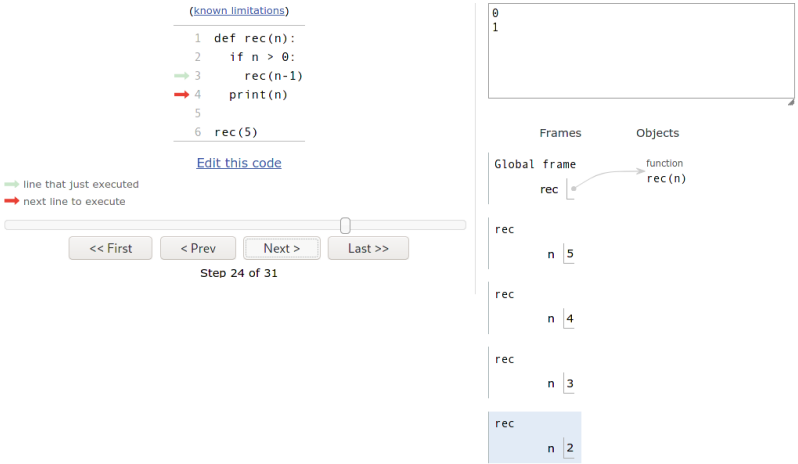
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



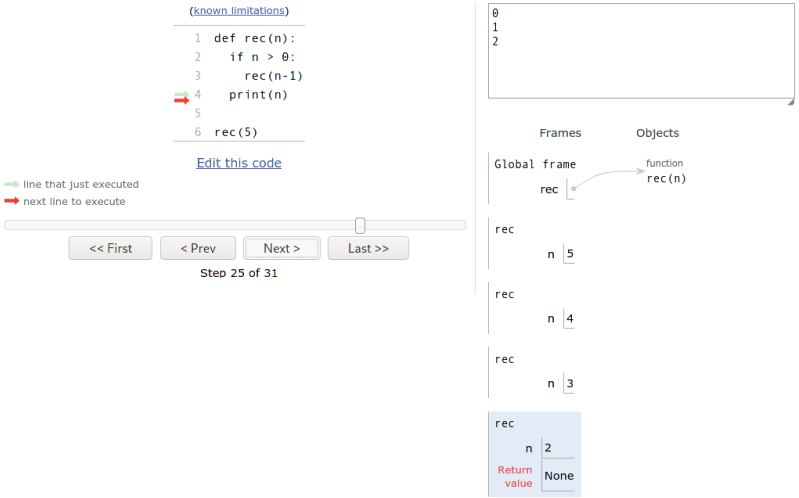
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



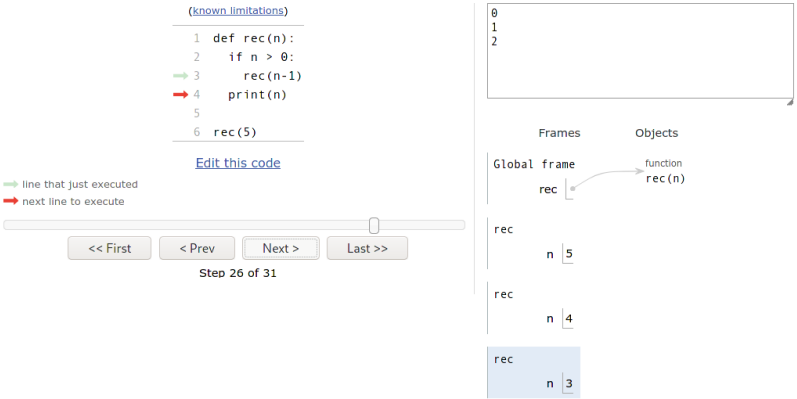
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



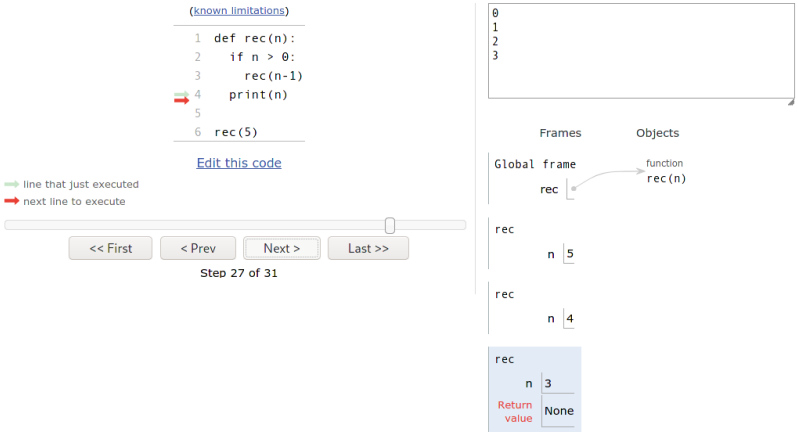
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



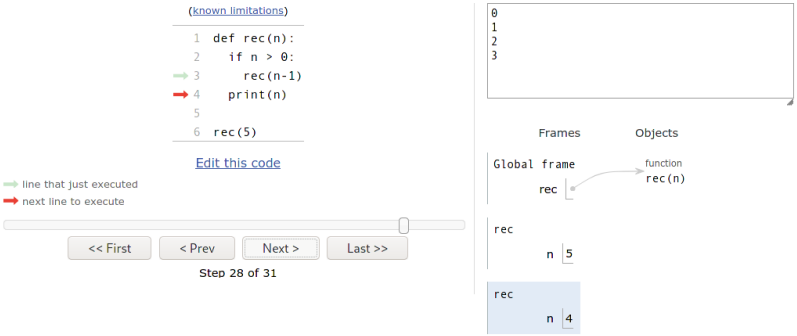
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



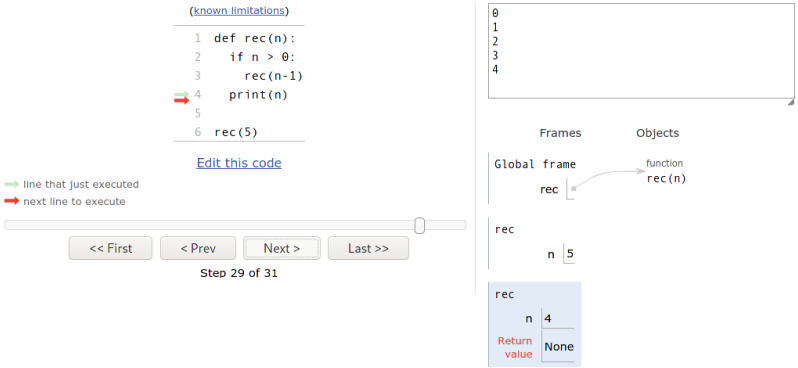
https://pythontutor.com (https://tinyurl.com/5ddj66wj)



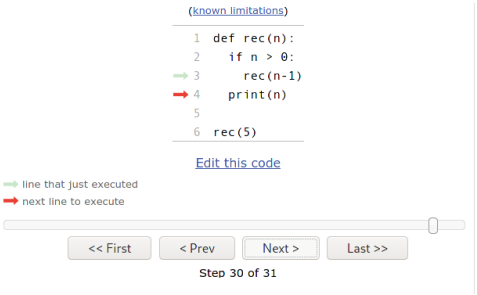
https://pythontutor.com (https://tinyurl.com/5ddj66wj)

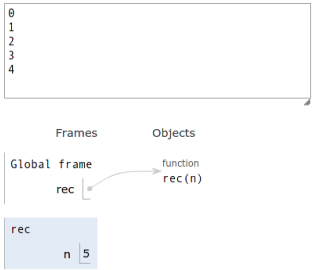


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

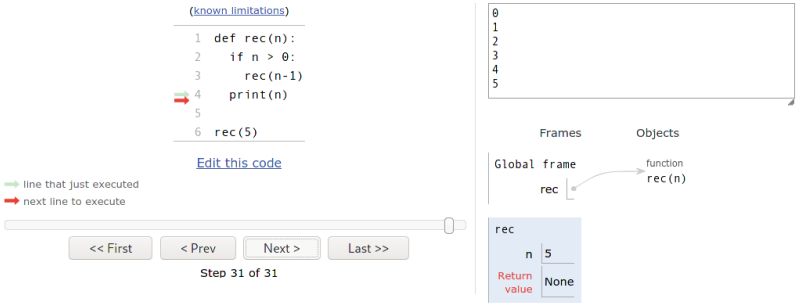


https://pythontutor.com (https://tinyurl.com/5ddj66wj)

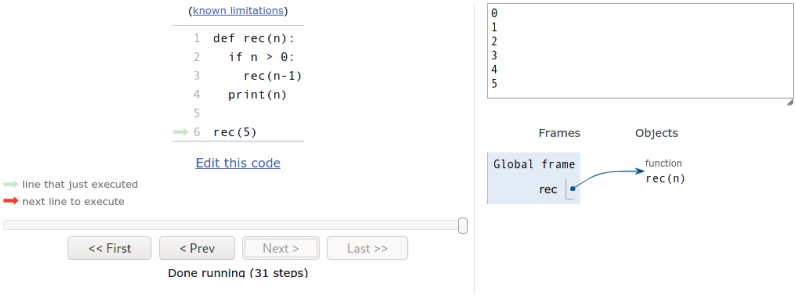




https://pythontutor.com (https://tinyurl.com/5ddj66wj)



https://pythontutor.com (https://tinyurl.com/5ddj66wj)



https://pythontutor.com (https://tinyurl.com/5ddj66wj)

Exemplo 3

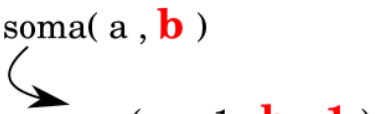
https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Soma de dois números inteiros não negativos, x e y, usando apenas incrementos e decrementos unitários.

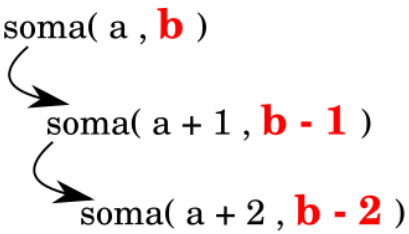
```
1 def soma(x, y):
2   if y == 0:
3     return x
4
5   return soma(x + 1, y - 1)
```

soma(a , **b**)

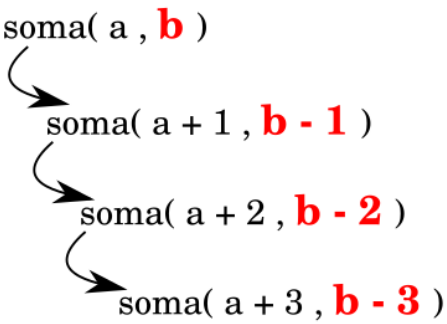
```
1 def soma(x, y):
2   if y == 0:
3     return x
4
5   return soma(x + 1, y - 1)
```



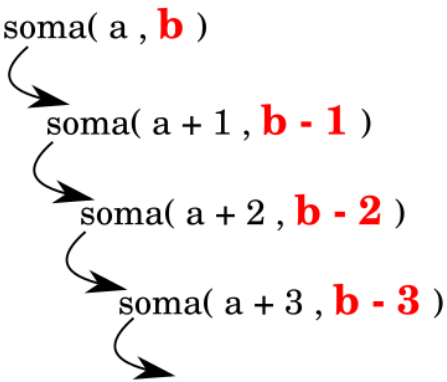

```
1  def soma(x, y):
2    if y == 0:
3      return x
4
5    return soma(x + 1, y - 1)
```



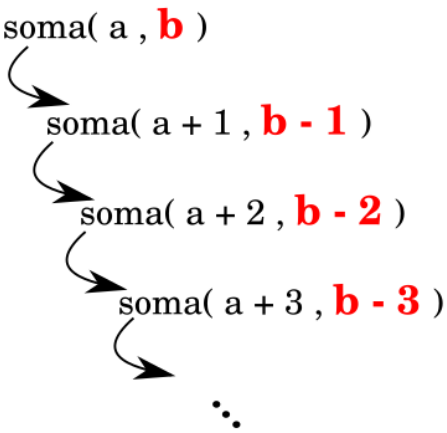

```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```



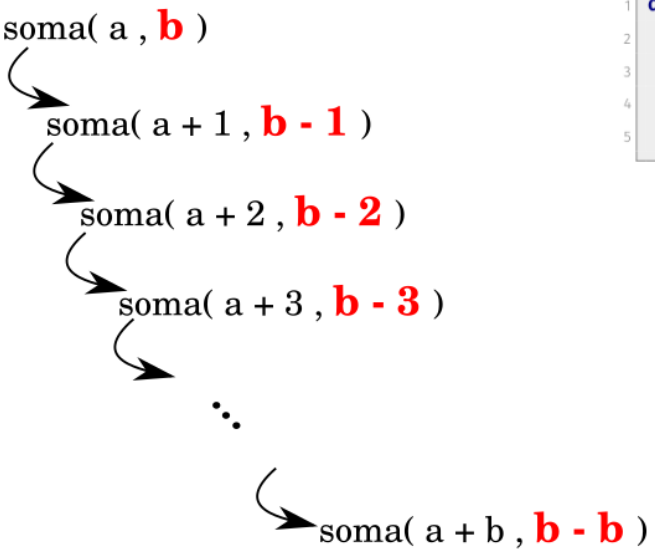

```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```



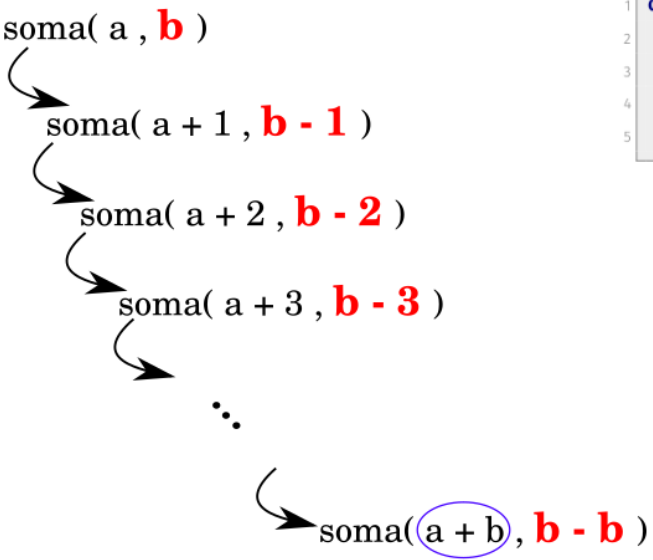

```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```




```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

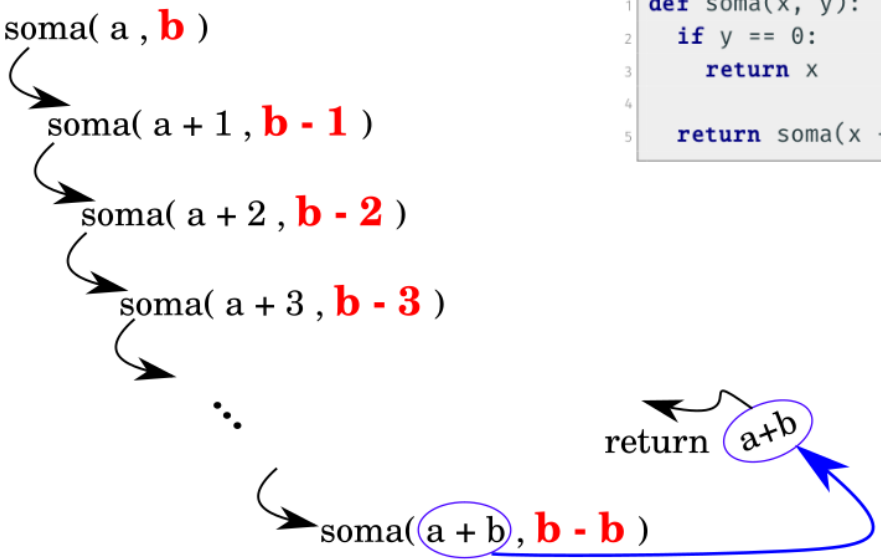



```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

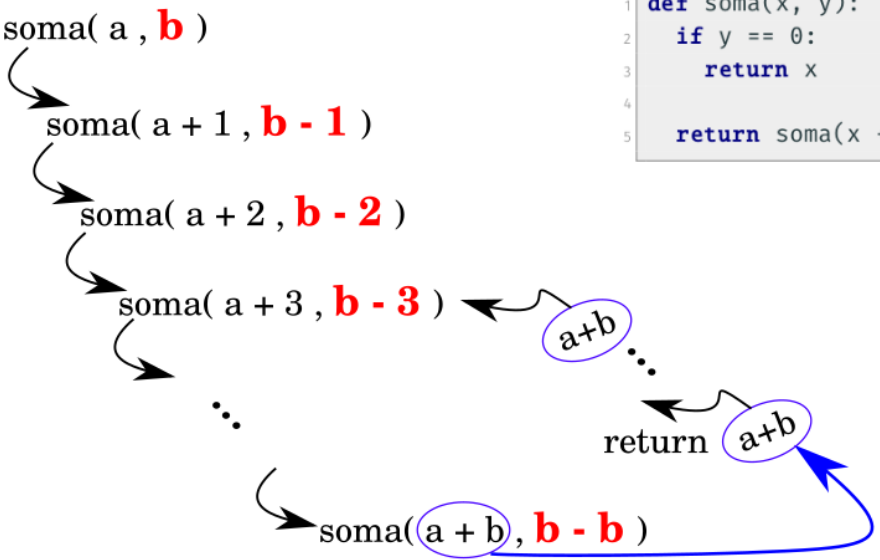



```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

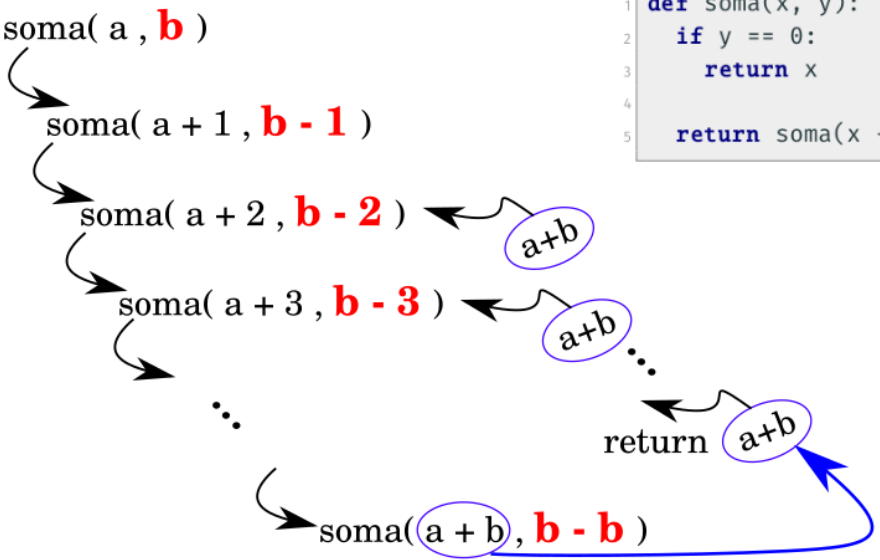
```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

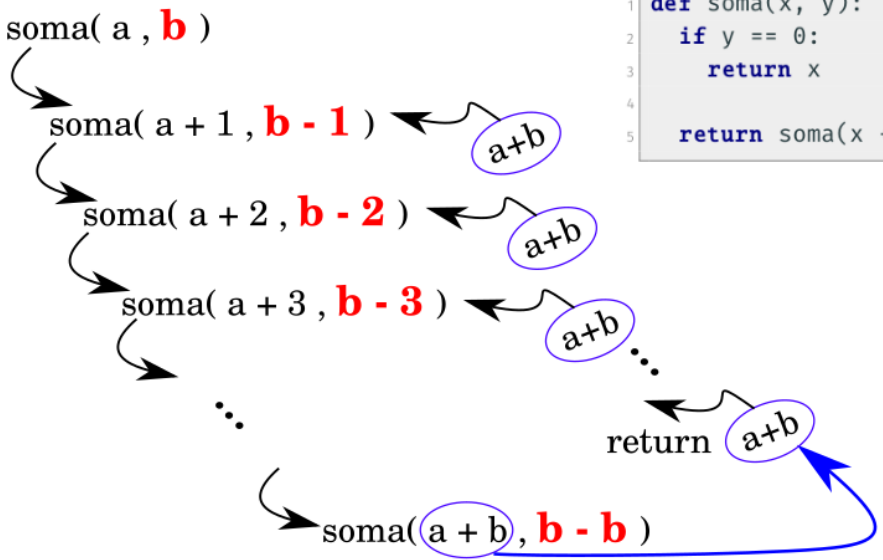


```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

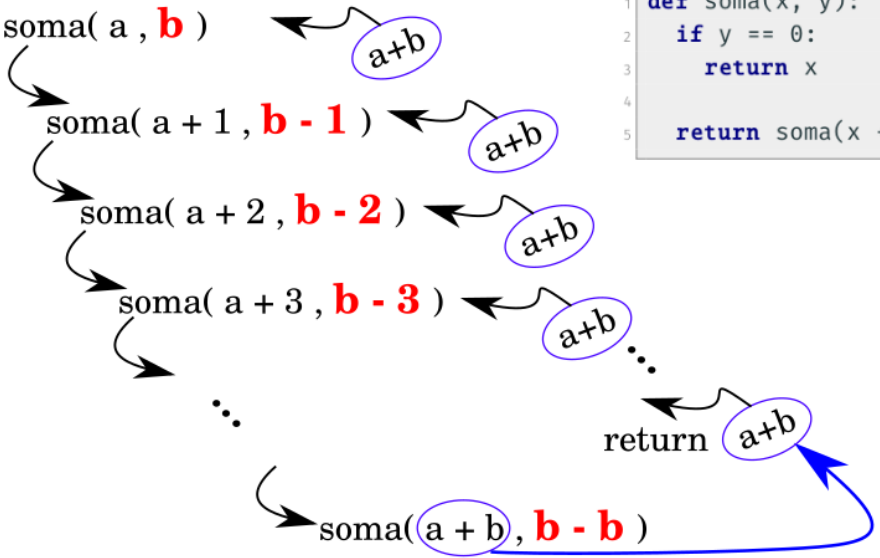


```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x + 1, y - 1)
```

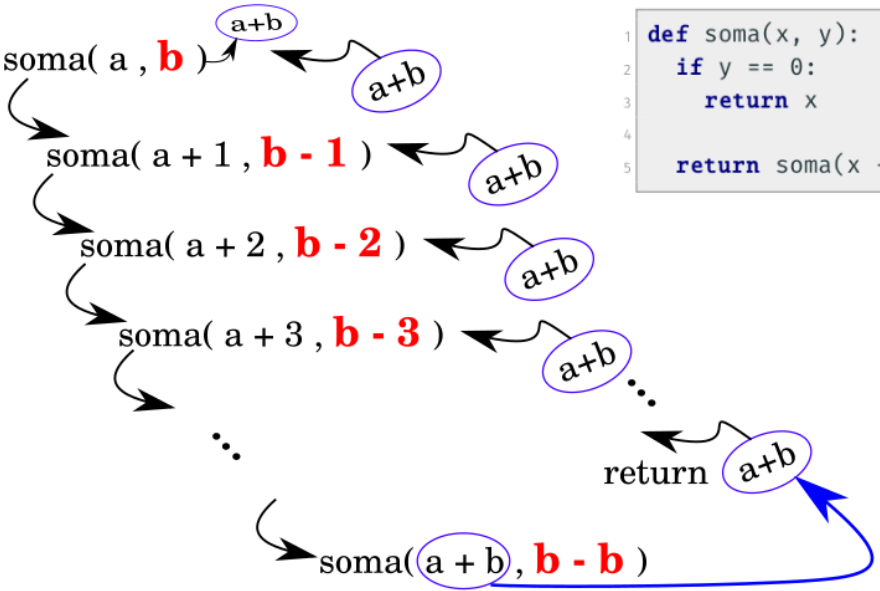





```
1  def soma(x, y):
2    if y == 0:
3      return x
4
5    return soma(x + 1, y - 1)
```




```
1  def soma(x, y):
2    if y == 0:
3      return x
4
5    return soma(x + 1, y - 1)
```




```
1  def soma(x, y):
2    if y == 0:
3      return x
4
5    return soma(x + 1, y - 1)
```

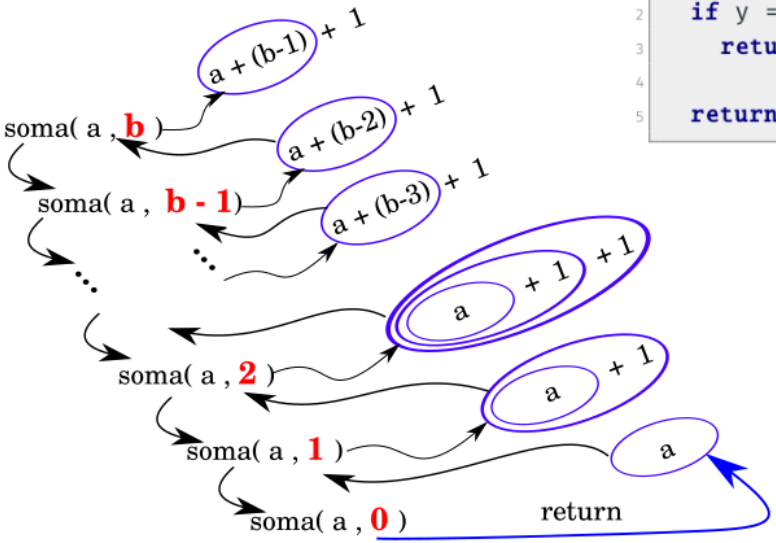
Exemplo 4

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Soma de dois números inteiros não negativos, x e y, usando apenas incrementos e decrementos unitários.

```
1 def soma(x, y):
2   if y == 0:
3     return x
4
5   return soma(x, y - 1) + 1
```

```python
def soma(x, y):
  if y == 0:
    return x

  return soma(x, y - 1) + 1
```



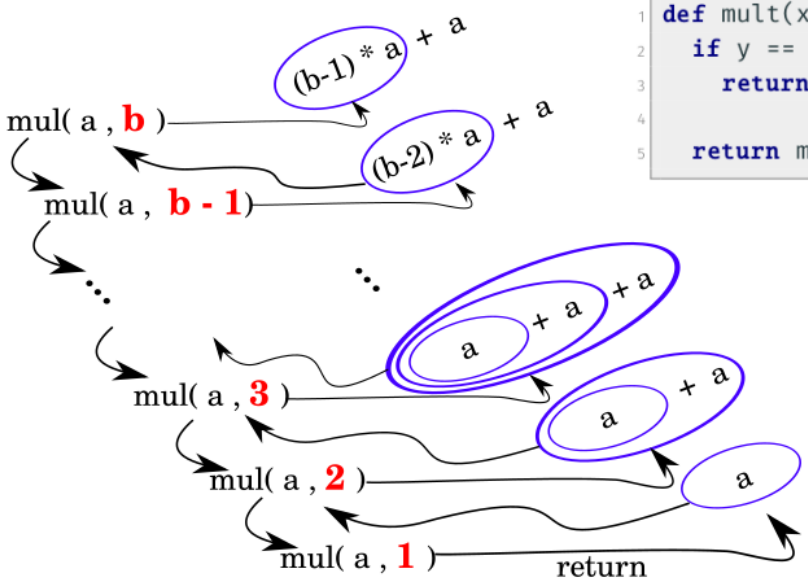
Exemplo 5

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Multiplicação de dois números inteiros positivos, x e y, usando apenas somas e subtrações.

```
1 def mult(x, y):
2   if y == 1:
3     return x
4
5   return mult(x, y - 1) + x
```

```python
def mult(x, y):
  if y == 1:
    return x

  return mult(x, y - 1) + x
```



Exemplo 6

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Soma de todos os inteiros positivos pares menores ou iguais a um valor inteiro *n*.

```
1 def soma_par(n):
2   if n <= 0:
3     return 0
4   else:
5     if n % 2 == 0:
6       return soma_par(n - 2) + n
7     else:
8       return soma_par(n - 1)
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Soma de todos os inteiros positivos pares menores ou iguais a um valor inteiro *n*.

```
1 def soma_par(n):
2   if n <= 0:
3     return 0
4
5   if n % 2 == 0:
6     return soma_par(n - 2) + n
7   else:
8     return soma_par(n - 1)
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Soma de todos os inteiros positivos pares menores ou iguais a um valor inteiro *n*.

```
1  def soma_par(n):
2    if n <= 1:
3      return 0
4
5    if n % 2 == 0:
6      return soma_par(n - 2) + n
7    else:
8      return soma_par(n - 1)
```

Exemplo 7

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Cálculo do produtório $\prod_{i=m}^{n} i = m \times (m+1) \times (m+2) \times \cdots \times n$, tal que m e n são inteiros tais que $m \leq n$.

```
1 def produtorio(m, n):
2   if m == n:
3     return m
4   else:
5     return produtorio(m, n - 1) * n
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Cálculo do produtório $\prod_{i=m}^{n} i = m \times (m+1) \times (m+2) \times \cdots \times n$, tal que m e n são inteiros tais que $m \leq n$.

```
1 def produtorio(m, n):
2   if m == n:
3     return m
4   else:
5     return m * produtorio(m + 1, n)
```

Exemplo 8

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Cálculo do valor de k^n, onde n é um número inteiro não negativo.

$$k^n = \begin{cases} 1, & \text{se } n = 0 \\ k \times k^{n-1}, & \text{caso contrário} \end{cases}$$

```
1 def potencia(k, n):
2   if n == 0:
3     return 1
4   else:
5     return k * potencia(k, n - 1)
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Cálculo do valor de k^n, onde n é um número inteiro não negativo.

$$k^n = \begin{cases} 1, & \text{se } n = 0 \\ k \times k^{n-1}, & \text{caso contrário} \end{cases}$$

```
1 def potencia(k, n):
2   pot = 1
3   for i in range(n):
4     pot = pot * k
5   return pot
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Podemos redefinir k^n (sendo n um inteiro não negativo) da seguinte forma:

$$k^n = \begin{cases} 1, & \text{se } n = 0 \\ k^{n/2} \times k^{n/2}, & \text{se } n \text{ for positivo e par} \\ k \times k^{\lfloor n/2 \rfloor} \times k^{\lfloor n/2 \rfloor}, & \text{se } n \text{ for positivo e ímpar} \end{cases}$$

- Note que definimos a solução do caso mais complexo em termos de casos mais simples. Usando esta definição, podemos implementar uma função iterativa ou recursiva, ambas mais eficientes que as versões anteriores.

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

```
1 def potencia(k, n):
2   if n == 0:
3     return 1
4   else:
5     aux = potencia(k, n // 2)
6     if n % 2 == 0:
7       return aux * aux
8     else:
9       return k * aux * aux
```

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

- Na nova versão do algoritmo, a cada chamada recursiva, o valor de n é dividido por 2. Ou seja, a cada chamada recursiva, o valor de n é reduzido, pelo menos, para a metade.
- Usando divisões inteiras, faremos no máximo $\lfloor \log_2 n \rfloor + 2$ chamadas recursivas.
- Por outro lado, a função iterativa original executa o laço n vezes.

Perguntas ....

Referências

- Zanoni Dias, MC102, Algoritmos e Programação de Computadores, IC/UNICAMP, 2021. https://ic.unicamp.br/~mc102/

  Aula Introdutória [slides] [vídeo]
  Primeira Aula de Laboratório [slides] [vídeo]
  Python Básico: Tipos, Variáveis, Operadores, Entrada e Saída [slides] [vídeo]
  Comandos Condicionais [slides] [vídeo]
  Comandos de Repetição [slides] [vídeo]
  Listas e Tuplas [slides] [vídeo]
  Strings [slides] [vídeo]
  Dicionários [slides] [vídeo]
  Funções [slides] [vídeo]
  Objetos Multidimensionais [slides] [vídeo]
  Algoritmos de Ordenação [slides] [vídeo]
  Algoritmos de Busca [slides] [vídeo]
  Recursão [slides] [vídeo]
  Algoritmos de Ordenação Recursivos [slides] [vídeo]
  Arquivos [slides] [vídeo]
  Expressões Regulares [slides] [vídeo]
  Execução de Testes no Google Cloud Shell [slides] [vídeo]
  Numpy [slides] [vídeo]
  Pandas [slides] [vídeo]

- Panda - Cursos de Computação em Python (IME -USP) https://panda.ime.usp.br/

  Como Pensar Como um Cientista da Computação https://panda.ime.usp.br/pensepy/static/pensepy/
  Aulas de Introdução à Computação em Python https://panda.ime.usp.br/aulasPython/static/aulasPython/

- Fabio Kon, Introdução à Ciência da Computação com Python http://bit.ly/FabioKon/
- Socratica, Python Programming Tutorials http://bit.ly/SocraticaPython/
- Google - online editor for cloud-native applications (Python programming) https://shell.cloud.google.com/
- w3schools - Python Tutorial https://www.w3schools.com/python/
- Outros, citados nos Slides.