Рекурсии

Python

Что такое рекурсия?

В программировании рекурсия, или же рекурсивная функция — это такая функция, которая вызывает саму себя.

Бесконечная рекурсия – проблема

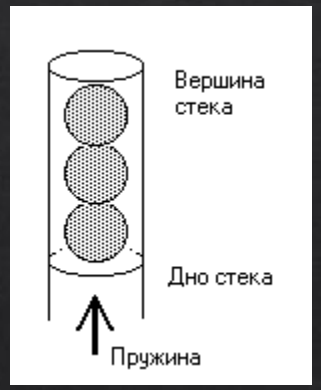
```
def recursion():
    recursion()

recursion()
```

```
RecursionError: maximum recursion depth exceeded
```

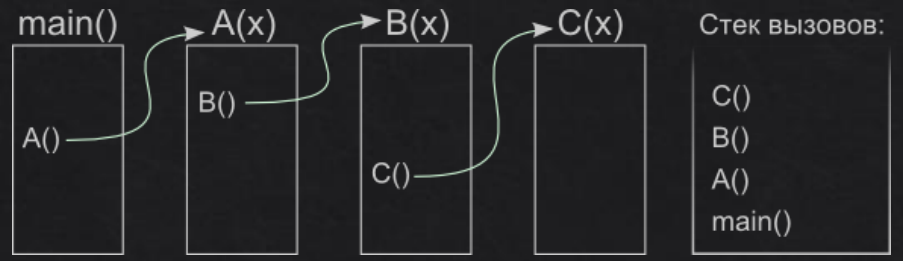
Почему при бесконечном цикле интерпретатор не выдаёт исключение, а при бесконечной рекурсии – выдаёт RecursionError?

Стек вызова

- Стек – структура данных, которая работает по принципу LIFO (Last In, First Out).



- Любой вызов функции заносится в стек для того чтобы иметь возможность вернуться после выполнения к месту, откуда она была вызвана.



Диапазон чисел

- Реализуем рекурсивную функцию для вывода ряда чисел от 1 до n, где n будет аргументом функции.

Код:

```
def numbers(n):
    if n > 1:
        numbers(n - 1)
    print(n, end=' ')

numbers(10)
```

Стек вызовов:

```
main()
- numbers(3)
-- numbers(2)
--- numbers(1)
---- print(1)
--- print(2)
-- print(3)
```

Немного изменим функцию

- Что выведет функция с прошлого слайда, если поменять порядок действий следующим образом и почему так происходит?

```
def numbers(n):
    print(n, end=' ')
    if n > 1:
        numbers(n - 1)

numbers(5)
```

Возвращаем значение

- Рекурсивные функции могут возвращать значения. В качестве примера рассмотрим задачу разворота числа.

```
def reverse_num(n, t = 0):
    if n != 0:
        return reverse_num(n // 10, t * 10 + n % 10)
    return t

print(reverse_num(12345))
```

Отладка рекурсии

- Отладка рекурсии может представлять из себя сложную задачу. Для простых рекурсий основной целью отладки является отслеживание порядка вызова функций, что можно сделать через вывод информации в коде или используя инструменты IDE.

```python
def fib(n: int) -> int:
    print(f'fib({n})')
    if n == 1 or n == 2:
        return 1
    return fib(n-1) + fib(n-2)

print(fib(5))
```