



Рекурсия

Преподаватель Маркова АВ



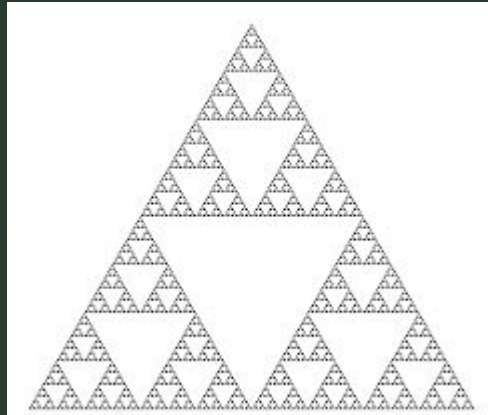
Рекурсия – способ определения множества объектов через само это множество на основе заданных простых базовых случаев.

Числа Фибоначчи

1) $F_1 = F_2 = 1;$

2) $F_n = F_{n-1} + F_{n-2}$

для $n > 2$



Примеры рекурсий:

1. Фракталы - треугольник Серпинского, фигуры обладающие свойствами самоподобия

У попа была собака, он её любил,
Она съела кусок мяса, он её убил,
В землю закопал,
Надпись написал:

У попа была собака, он её любил,
Она съела кусок мяса, он её убил,
В землю закопал,
Надпись написал:

...

▶ Рекурсивная процедура (функция) – это процедура (функция), которая вызывает сама себя напрямую или через другие процедуры и функции.

```
def df(n):  
    if n == 0:  
        return 1  
    else:  
        df(n // 2)  
        print(n % 2, end = " ")
```

```
df(int(input()))
```

Рекурсия заменяет цикл

Задача: Вычислите сумму всех цифр числа

```
def df(n):  
    s = n % 10  
    if n >= 10:  
        s += df(n // 10)  
    return s
```

```
print(df(89))
```

df(89)

9 + df(8)

9 + 8

Как работает рекурсия?

Факториал:

```
def Fact(N):  
    print ( "->", N )  
    if N <= 1: F = 1  
    else:  
        F = N * Fact ( N - 1 )  
    print ( "<-" , N )  
    return F
```

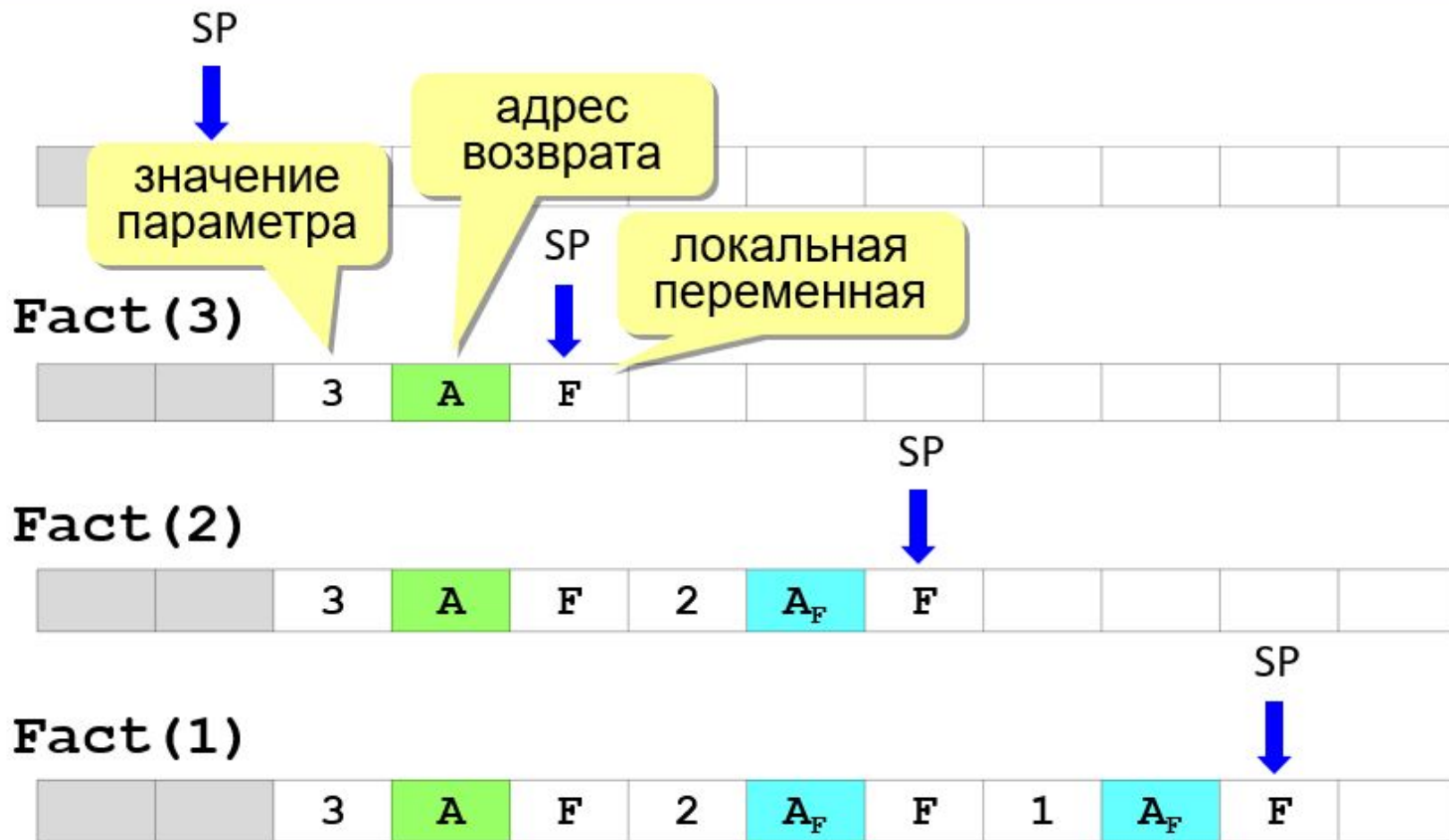
```
-> N = 3  
    -> N = 2  
        -> N = 1  
            <- N = 1  
        <- N = 2  
    <- N = 3
```



Как сохранить состояние функции перед рекурсивным вызовом?

Стек

Стек – область памяти, в которой хранятся локальные переменные и адреса возврата.



Рекурсия – «за» и «против»

- с каждым новым вызовом расходуется память в стеке (возможно переполнение стека)
- затраты на выполнение служебных операций при рекурсивном вызове



▪ программа становится более короткой и понятной



- возможно переполнение стека
- замедление работы



Любой рекурсивный алгоритм можно заменить итерационным!

итерационный
алгоритм

```
def Fact ( n ) :  
    f = 1  
    for i in range ( 2 , n + 1 ) :  
        f *= i  
    return f
```