

Лекция 4

в которой мы в первый раз встретимся с Фибоначчи
и узнаем про обобщения и функции высших порядков

Пример с итерациями

Последовательность Фибоначчи

Последовательность Фибоначчи



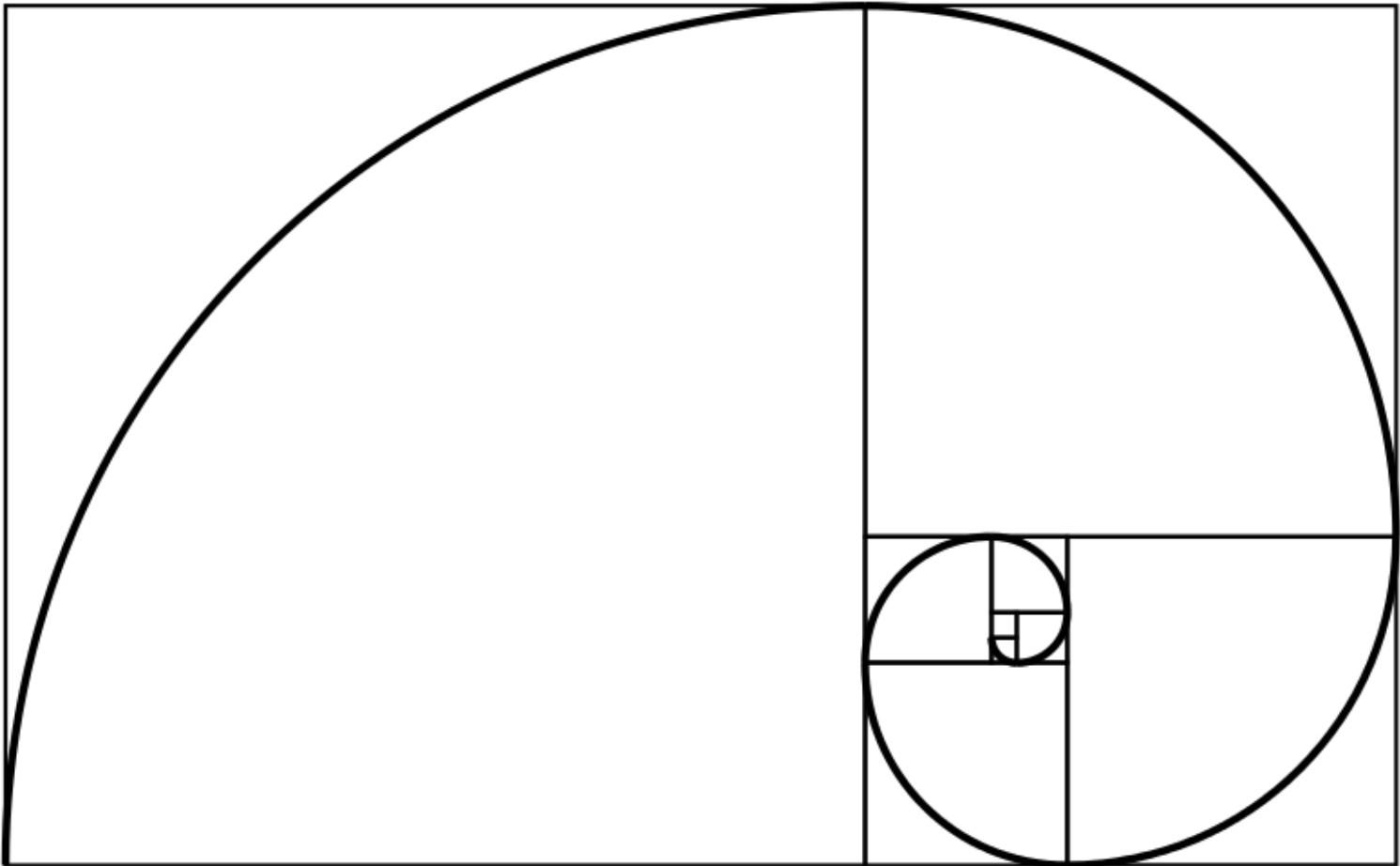
Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



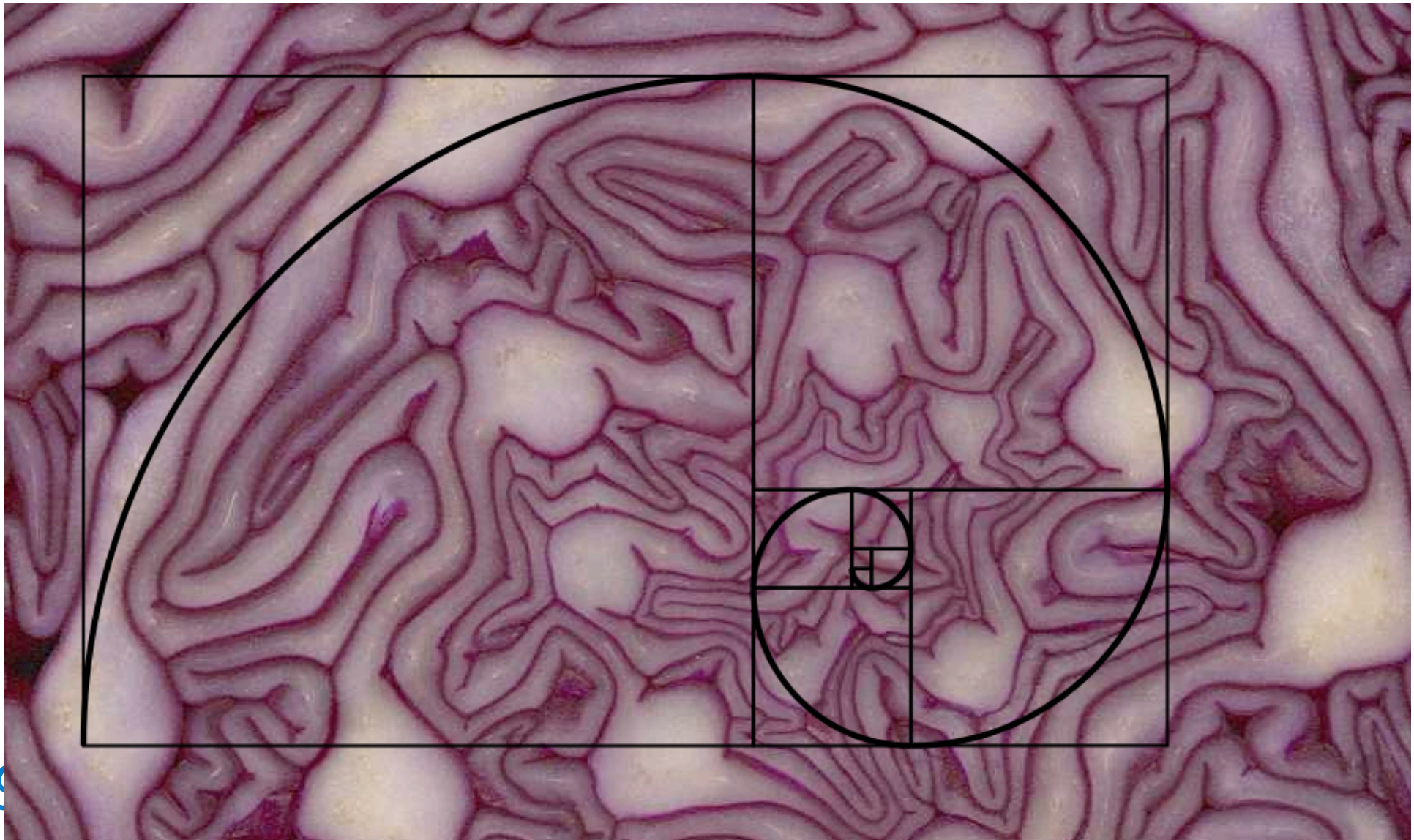
Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



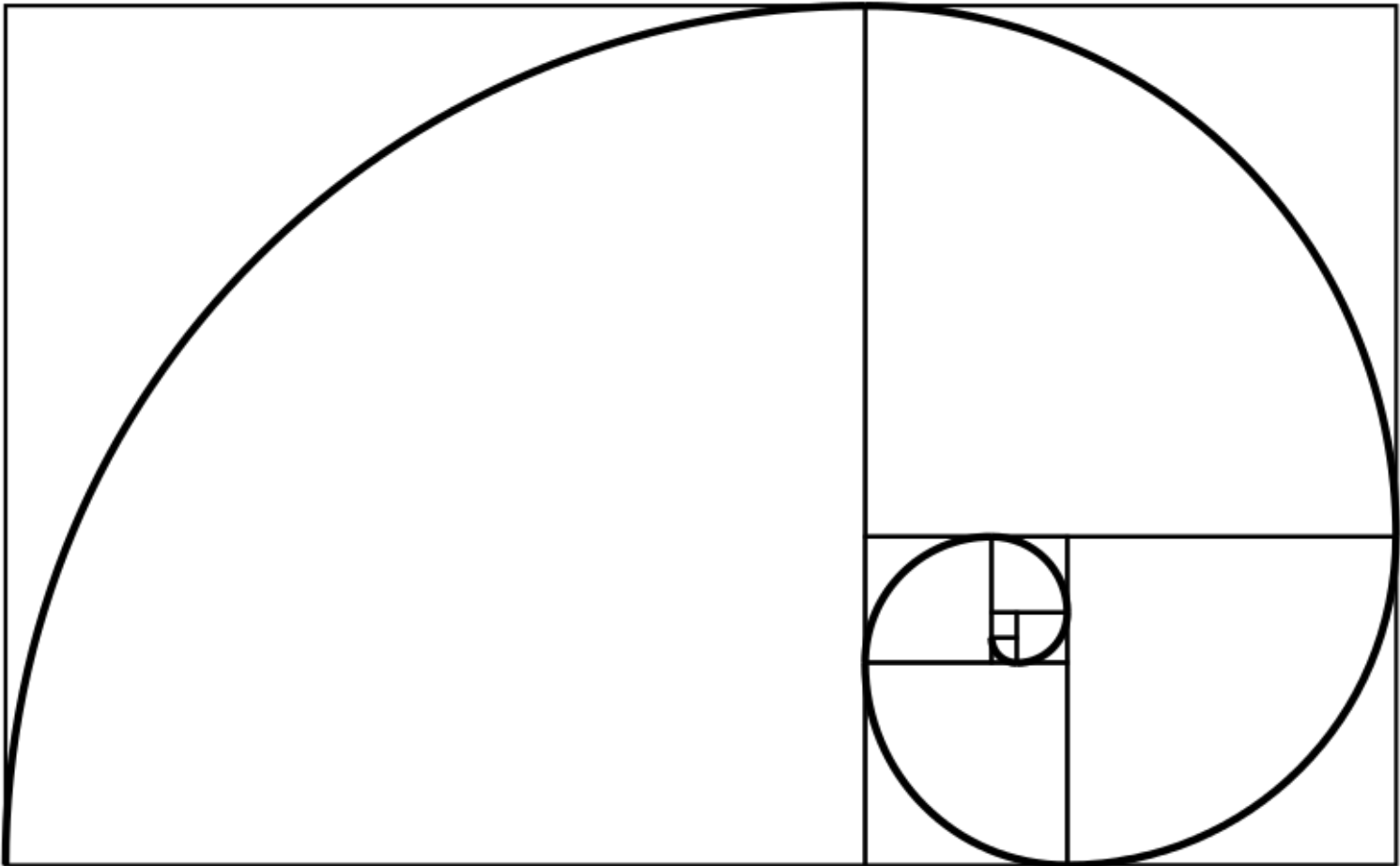
Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



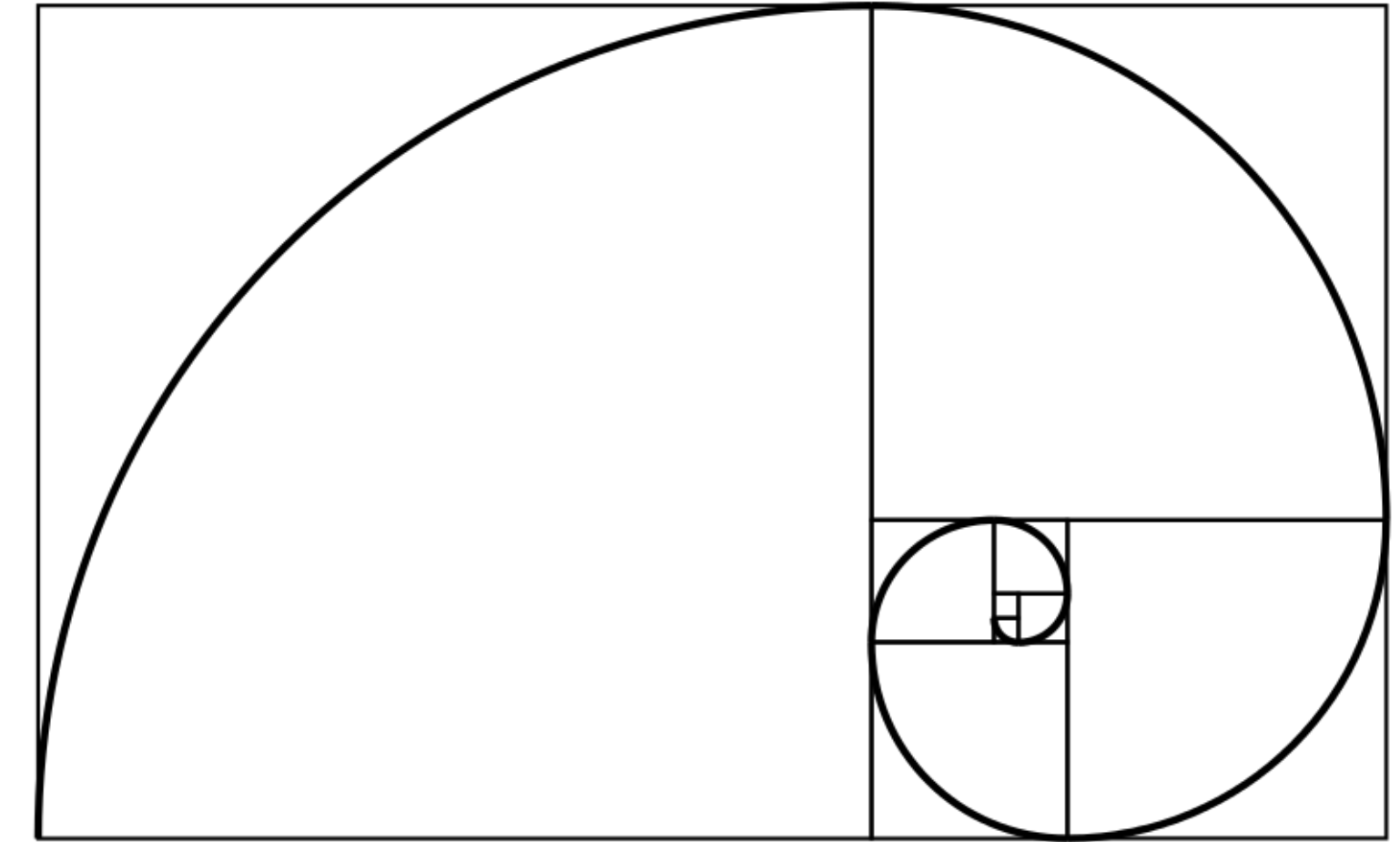
Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987

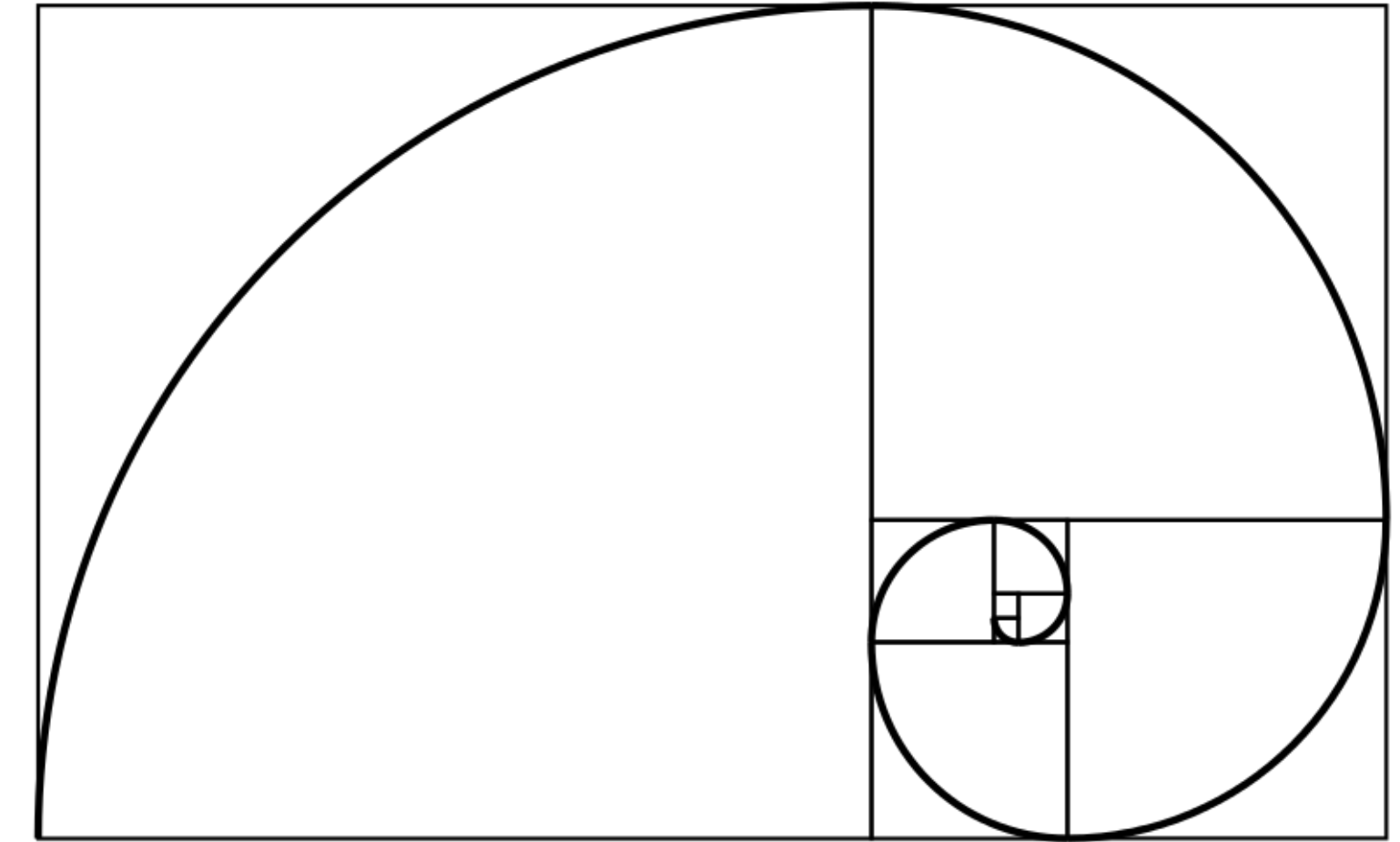


```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr — k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



Последовательность Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987

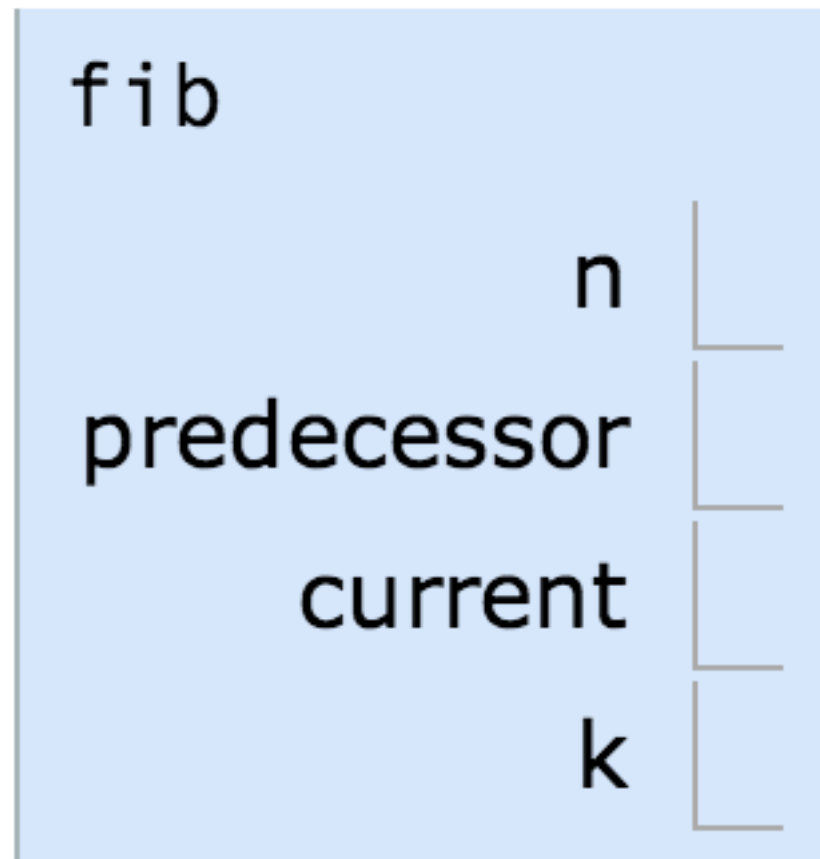


```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr — k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```

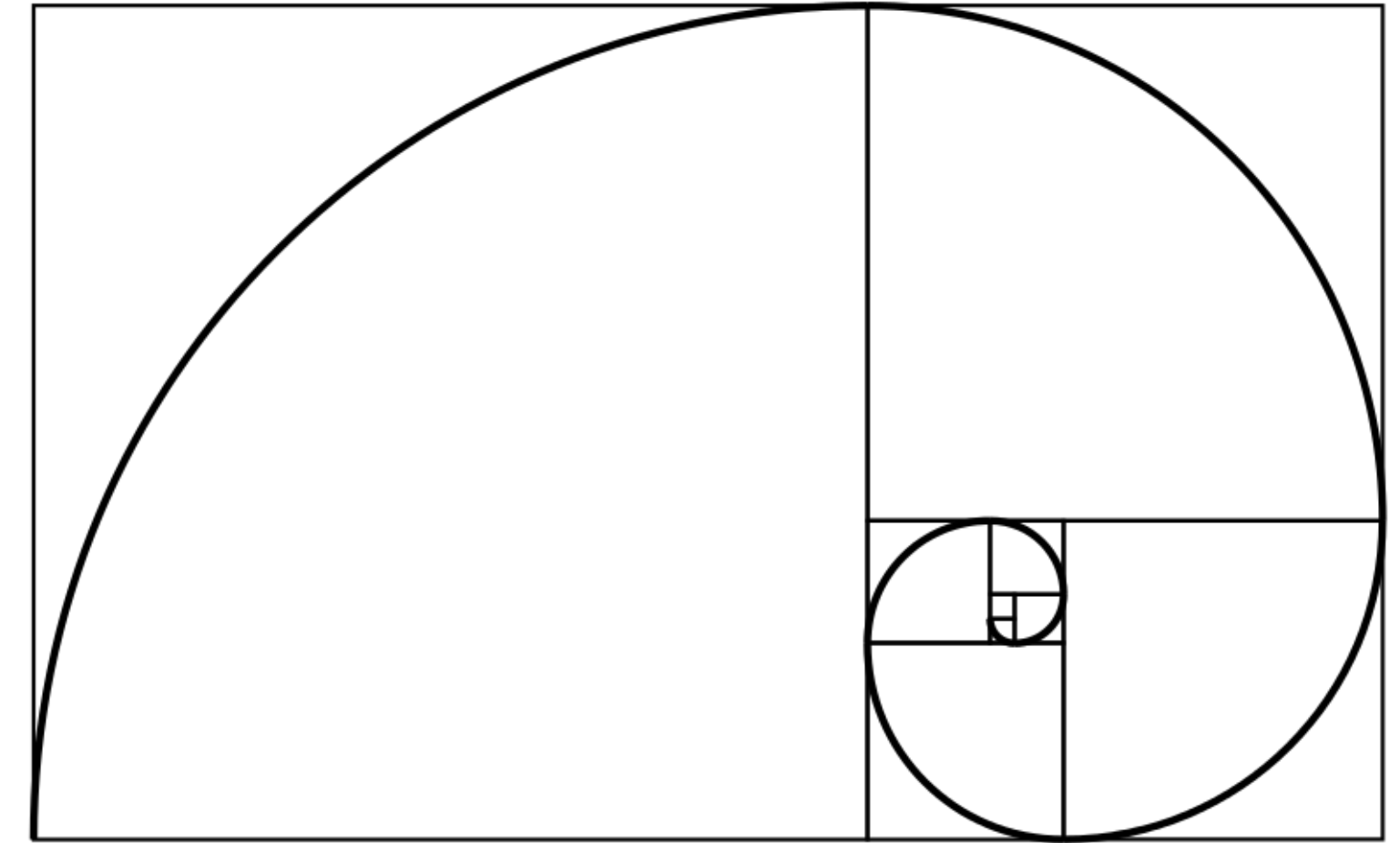
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987

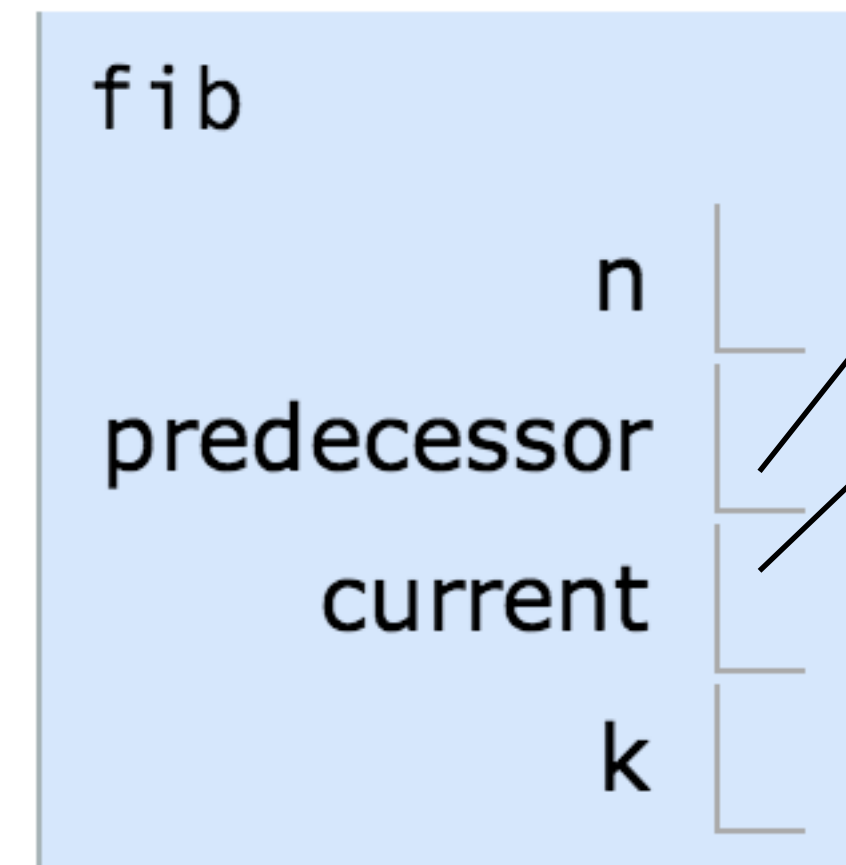


```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr — k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```

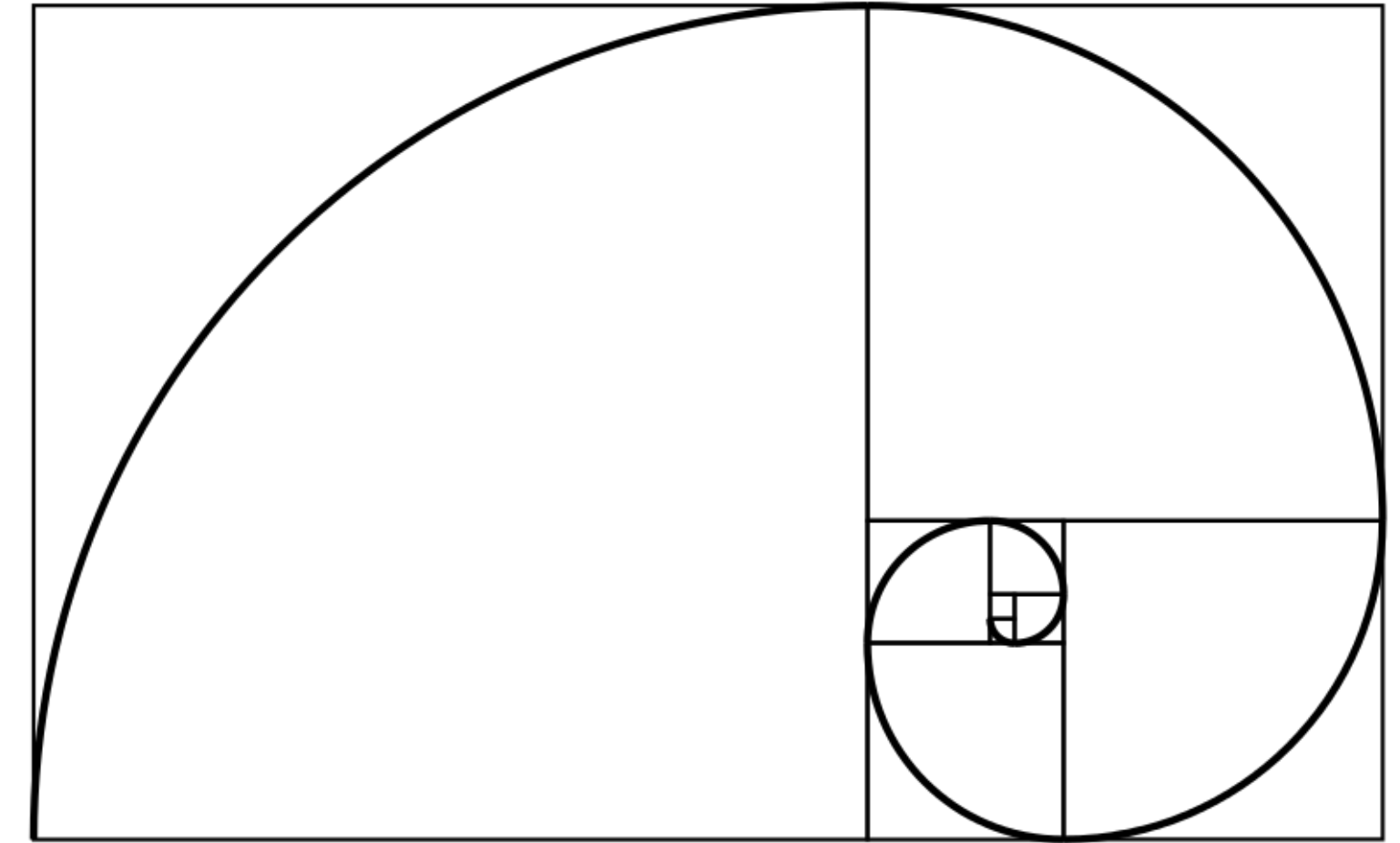
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987

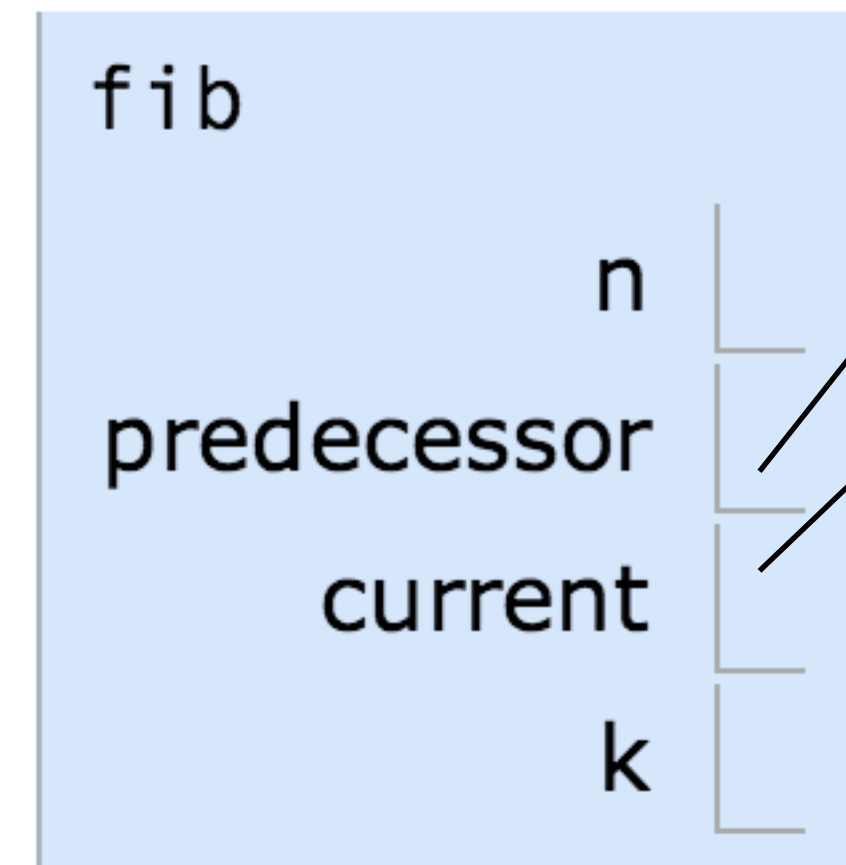


```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```

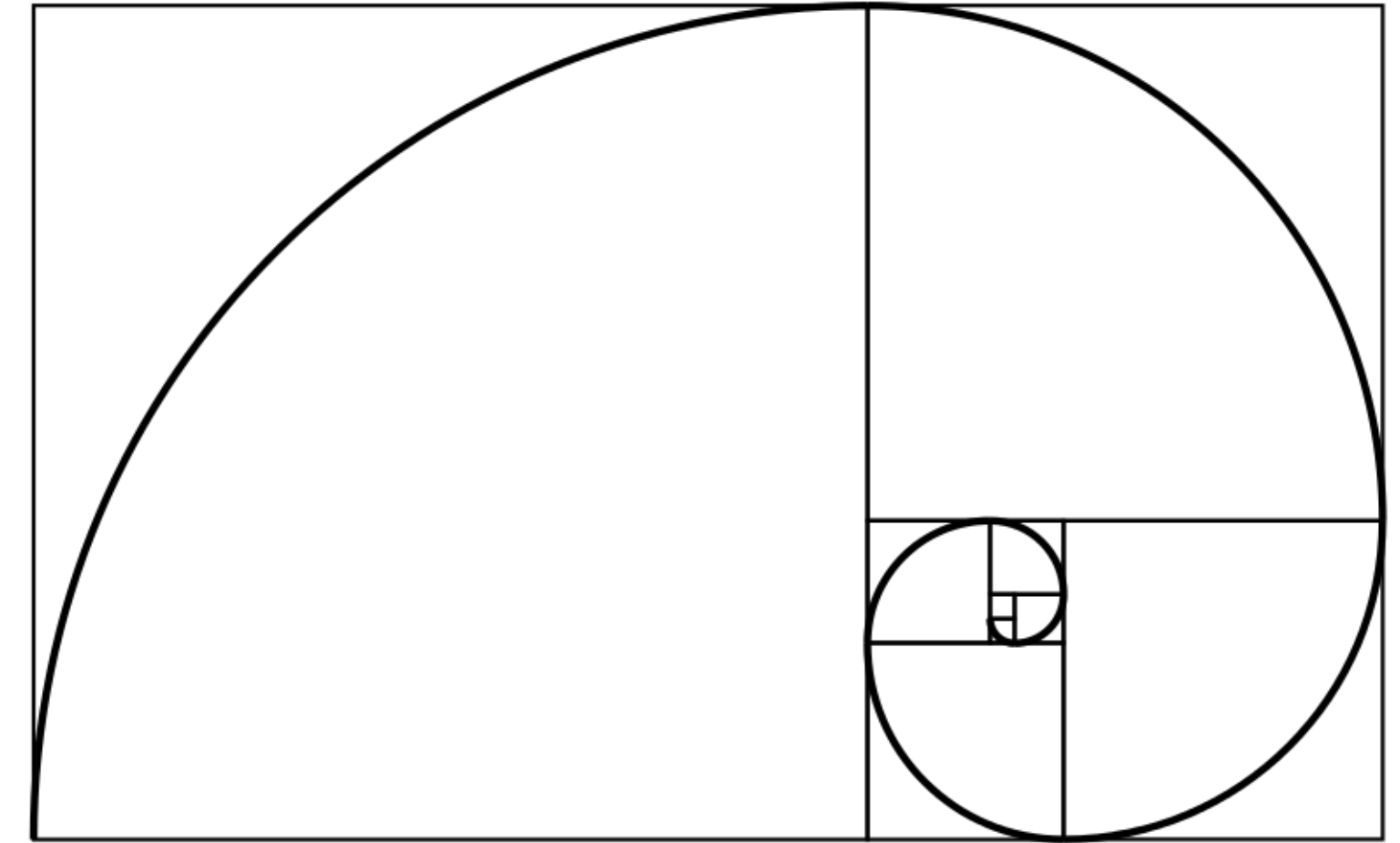
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



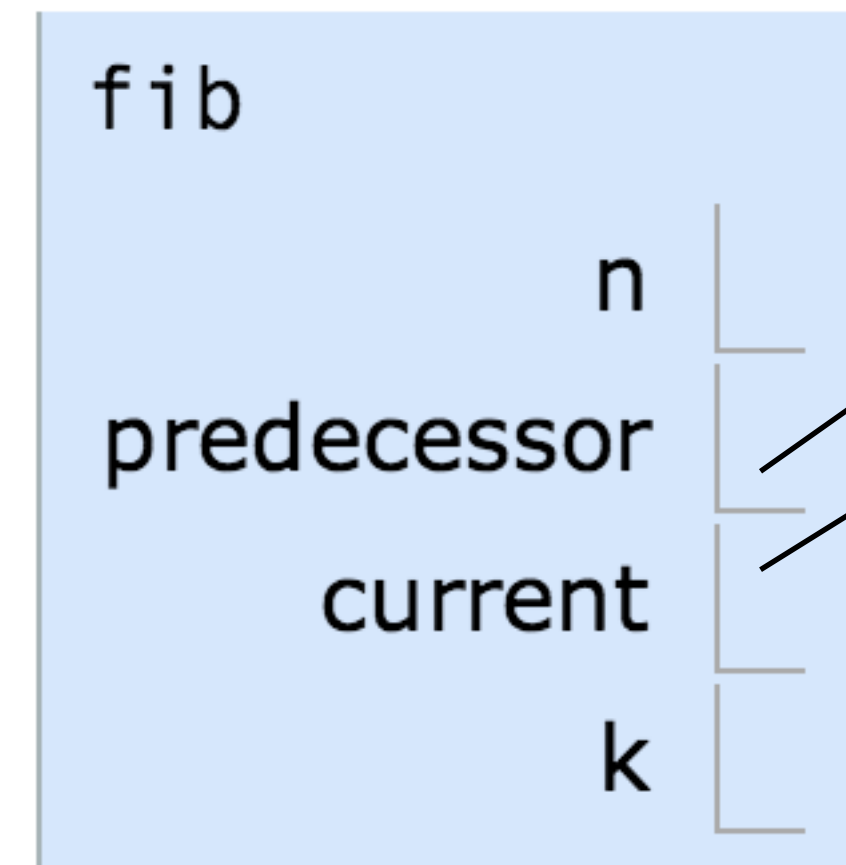
```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



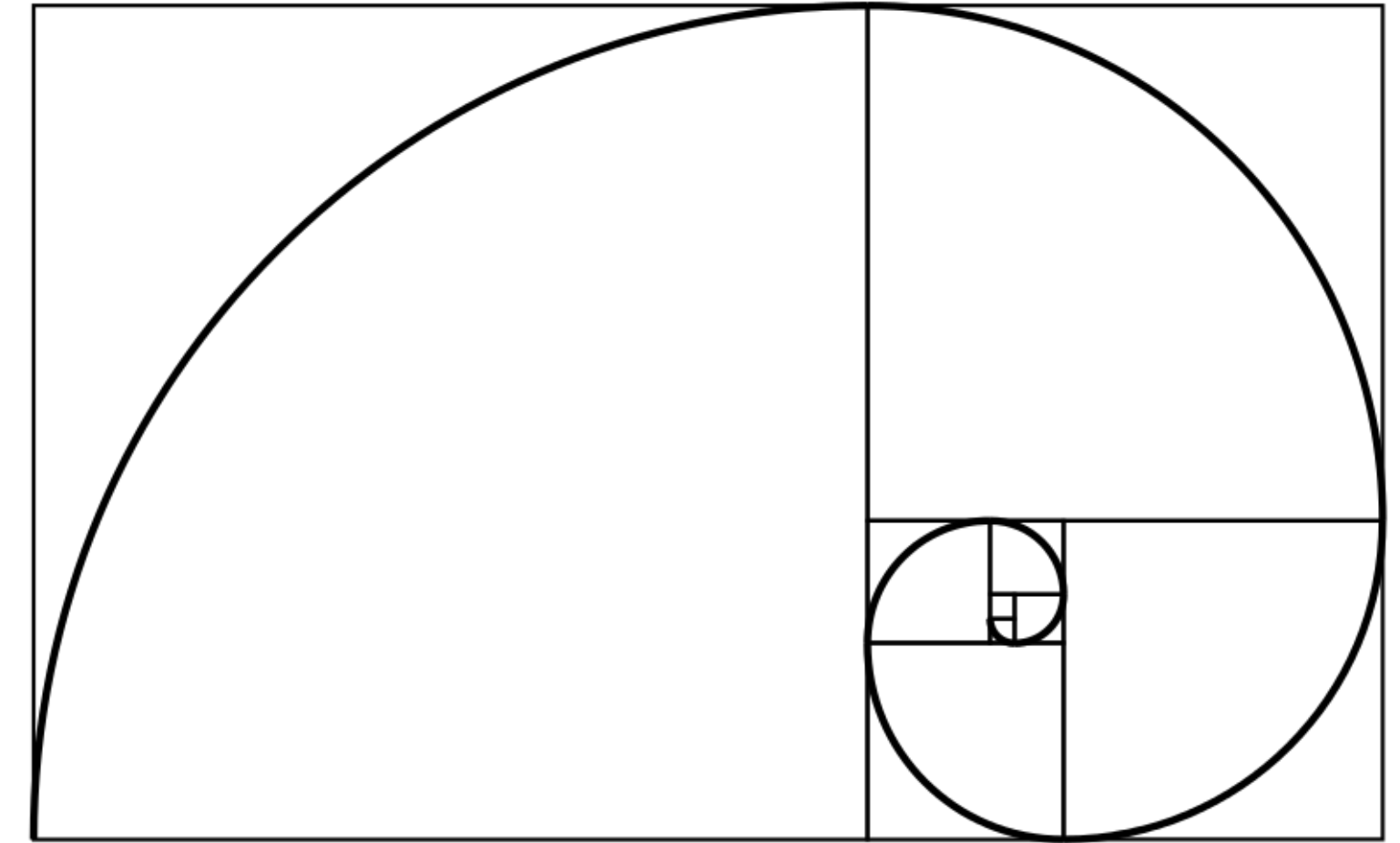
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



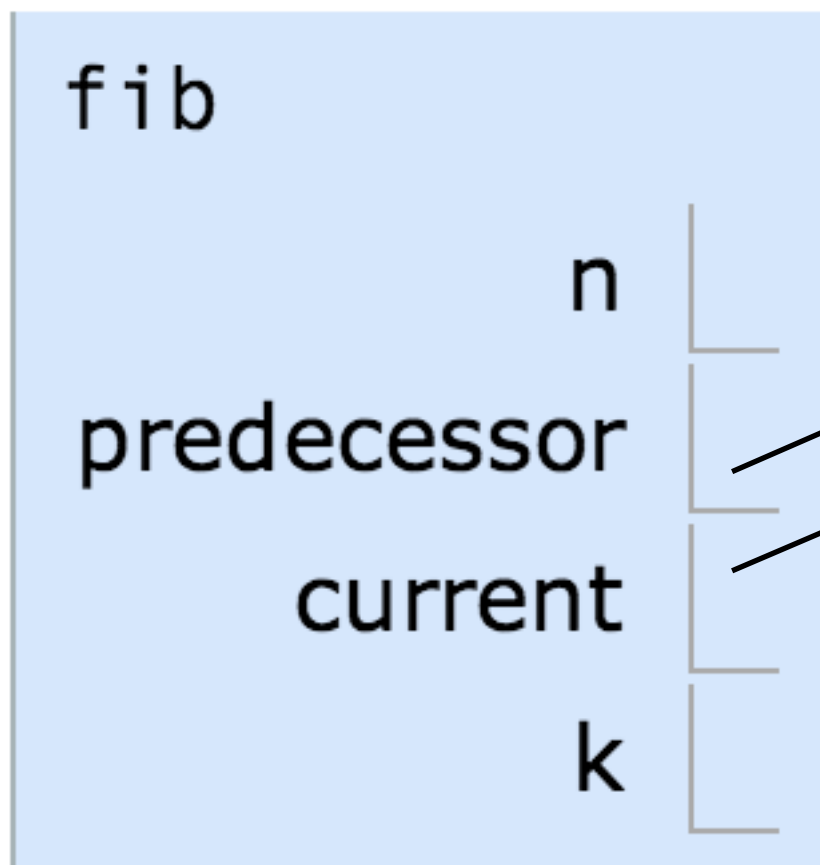
```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



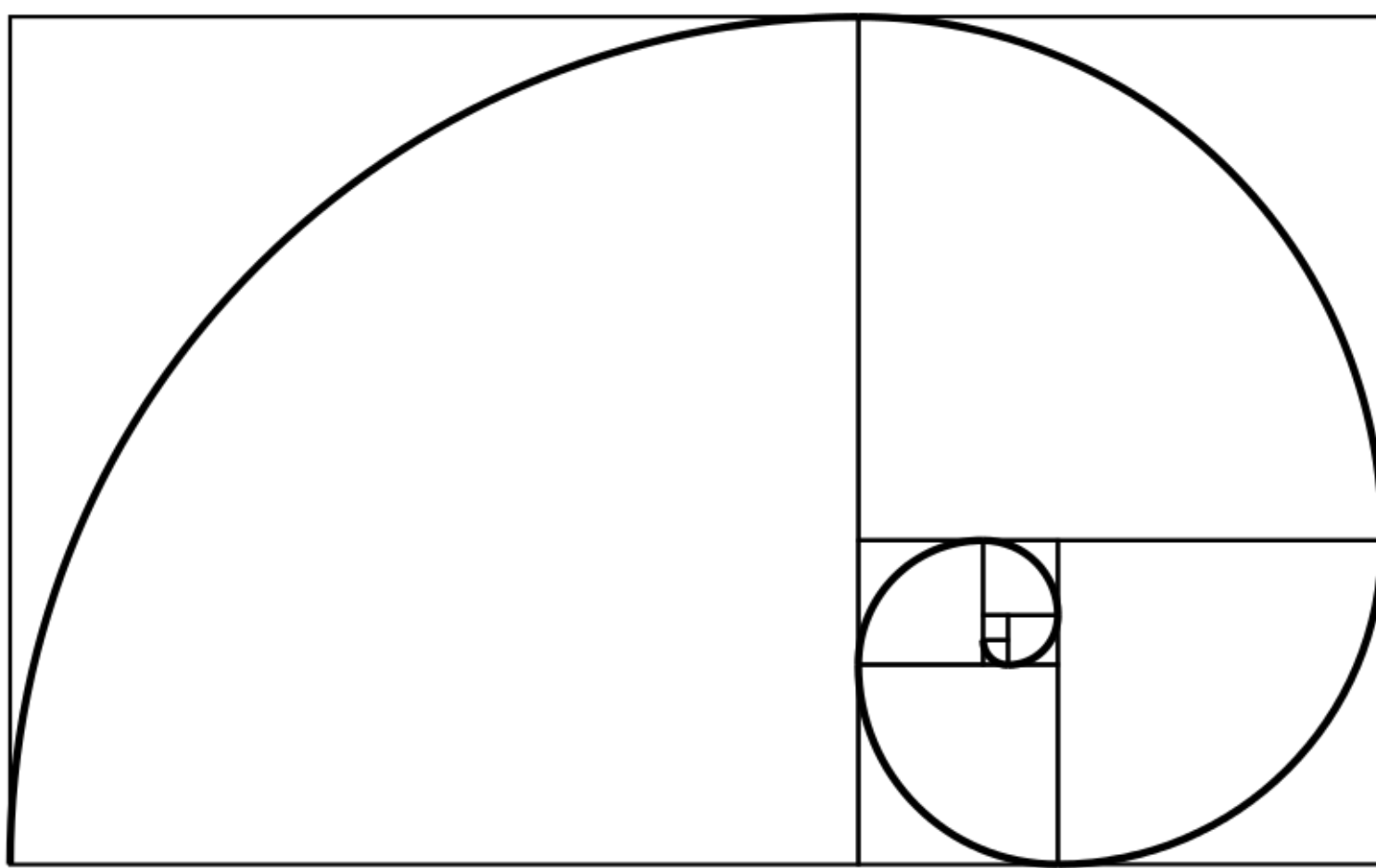
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



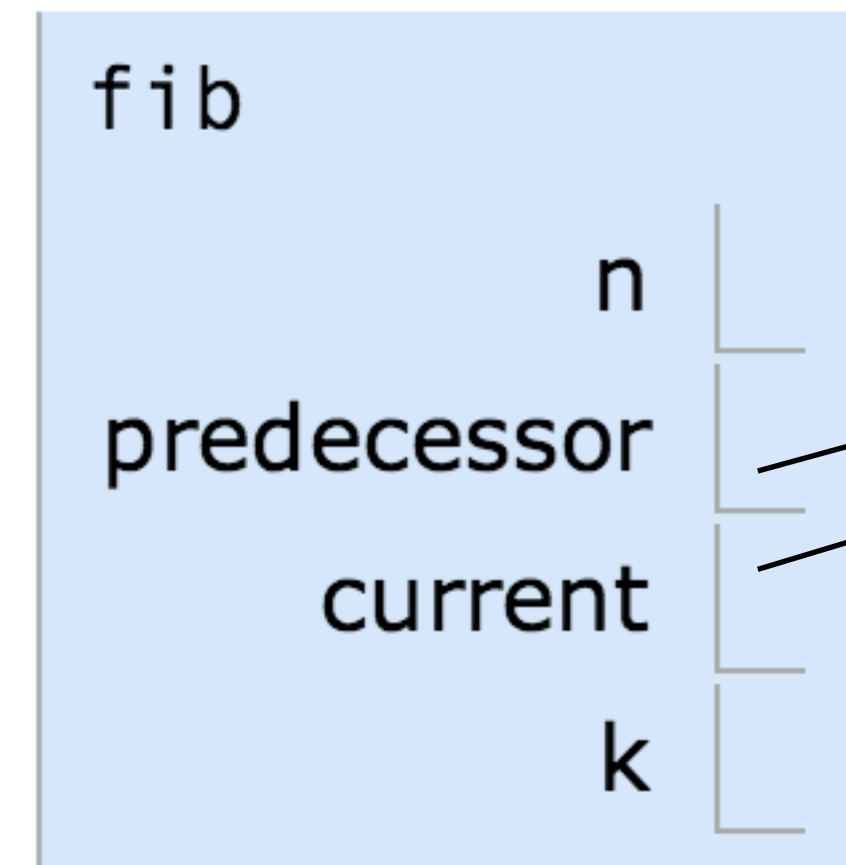
```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



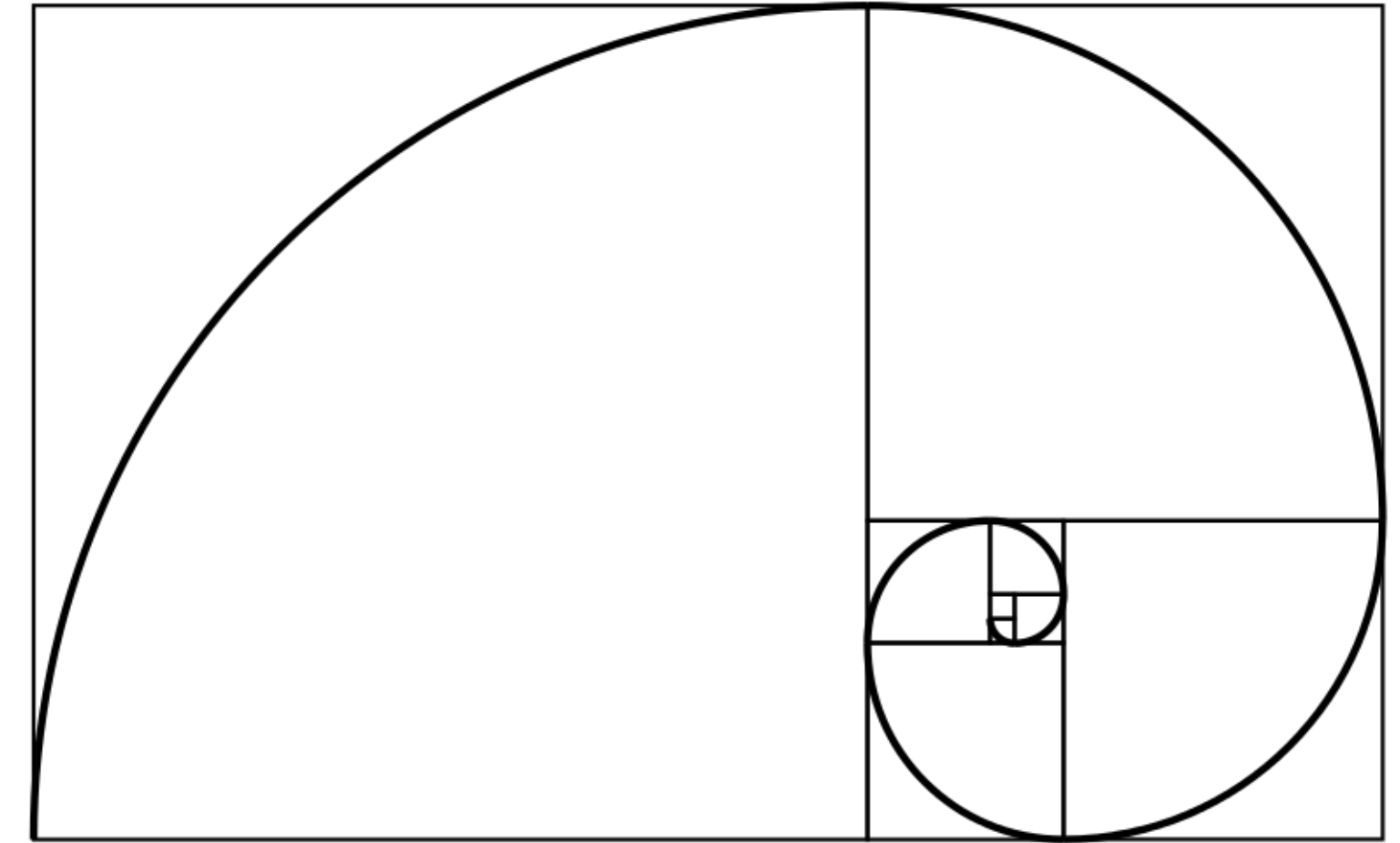
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



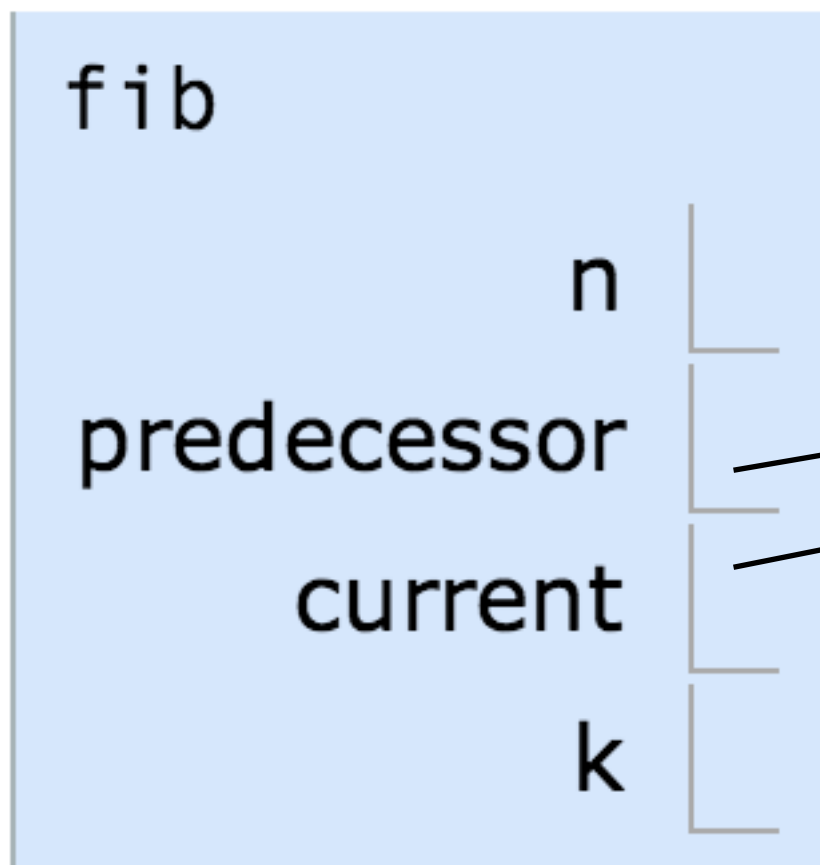
```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



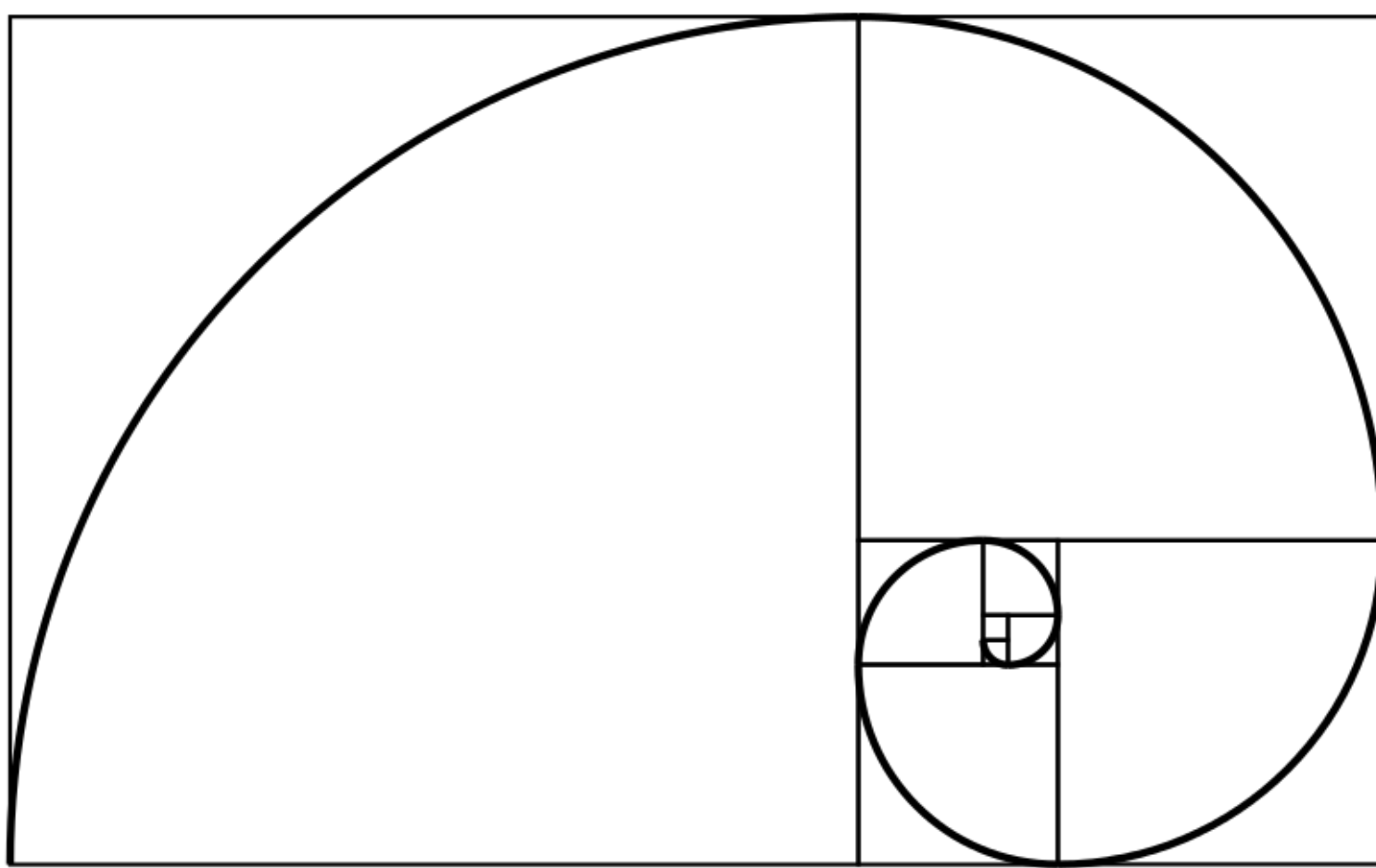
Следующее число Фибоначчи является суммой текущего и предыдущего



Последовательность Фибоначчи



0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""  
    pred, curr = 0, 1    # нулевое и первое числа Фибоначчи  
    k = 1                # curr - k-ое число Фибоначчи  
    while k < n:  
        pred, curr = curr, pred + curr  
        k = k + 1  
    return curr
```



Следующее число Фибоначчи является суммой текущего и предыдущего



Вопрос 1

Что вычисляет эта функция?



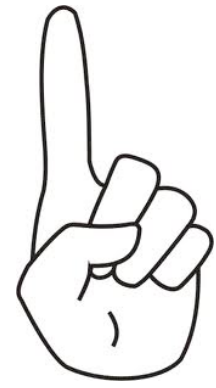
Вопрос 1

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



$$n^2$$



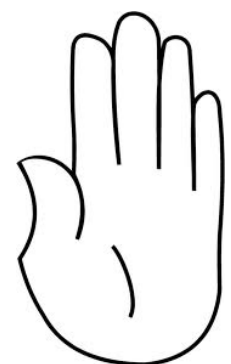
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



$$n^2 + 1$$



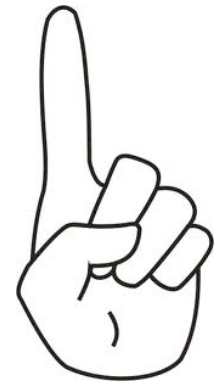
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



$$n^2$$



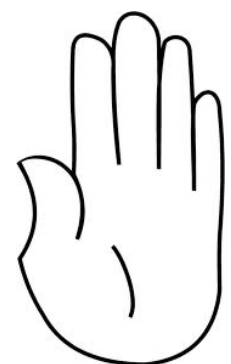
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



$$n^2 + 1$$



$$n \cdot (n + 1)$$

Что вычисляет эта функция?

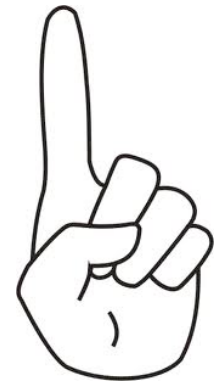
```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```

a

b



Вопрос 1



$$n^2$$



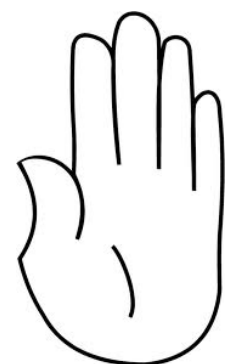
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



$$n^2 + 1$$



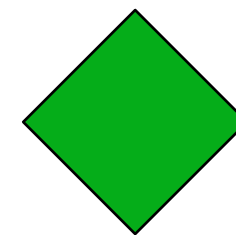
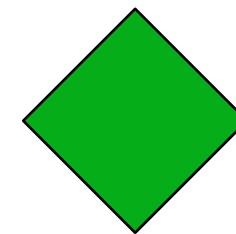
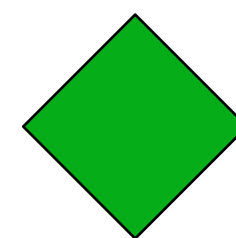
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

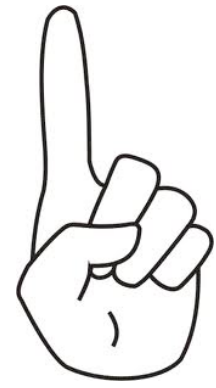
```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```

a

b



Вопрос 1



$$n^2$$



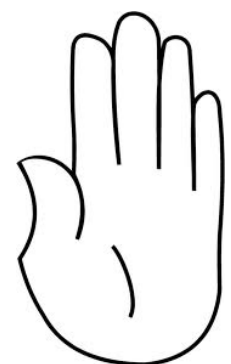
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



$$n^2 + 1$$



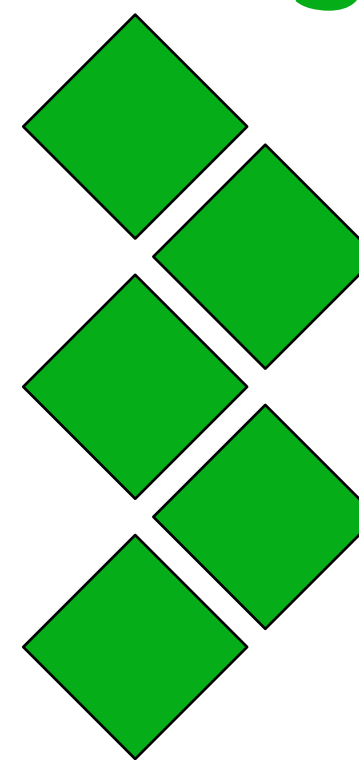
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```

a

b



Вопрос 1



$$n^2$$



$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



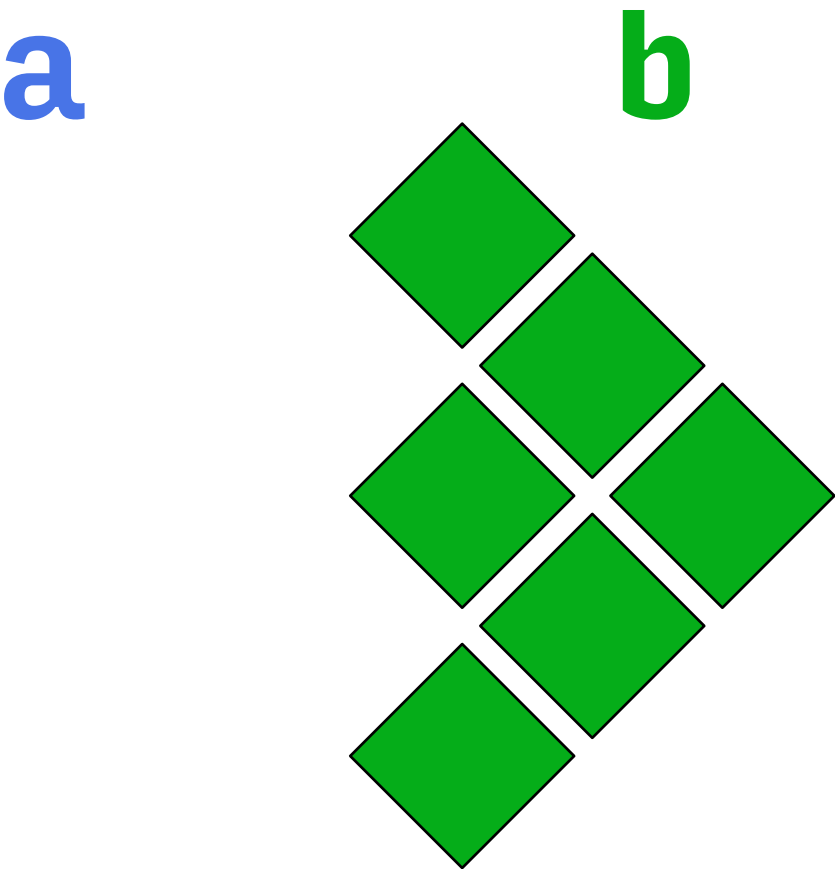
$$n^2 + 1$$



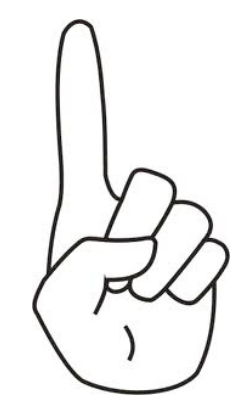
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



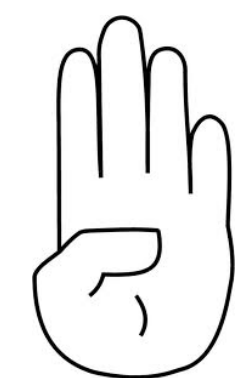
$$n^2$$



$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



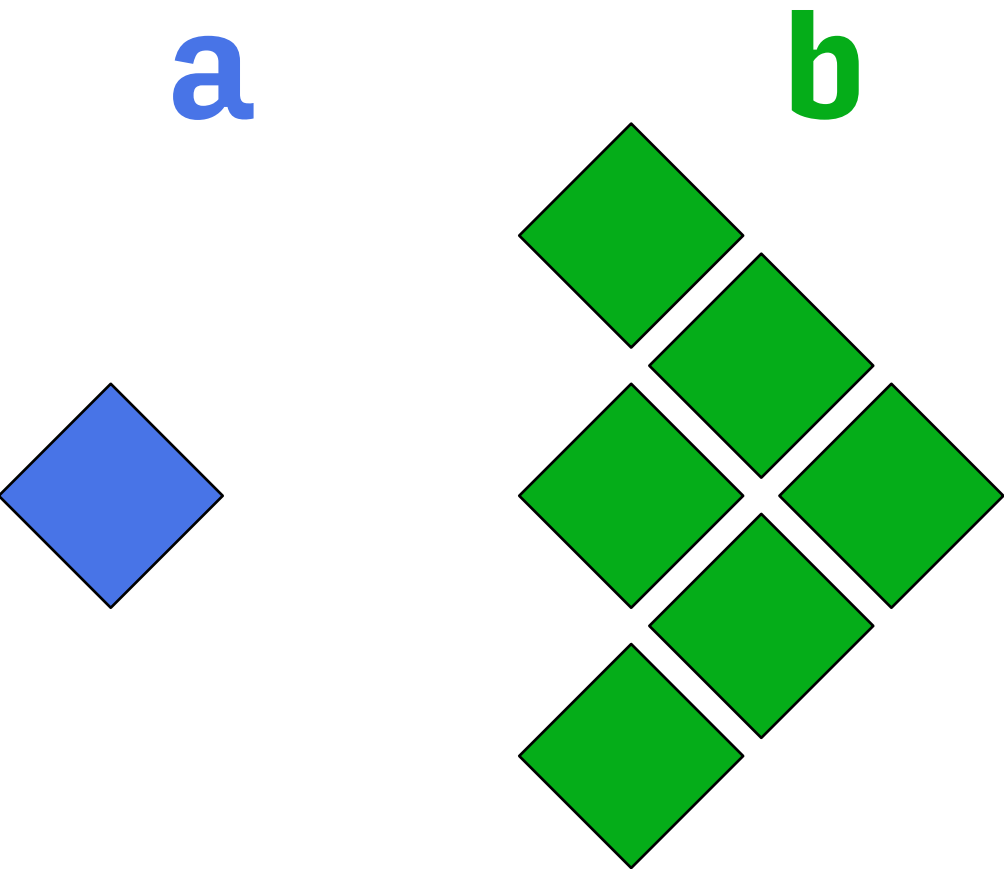
$$n^2 + 1$$



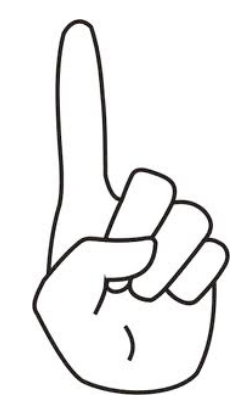
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



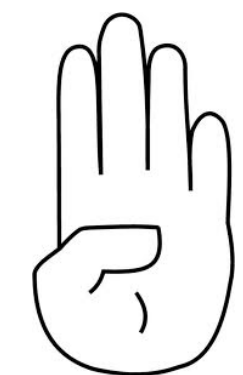
$$n^2$$



$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



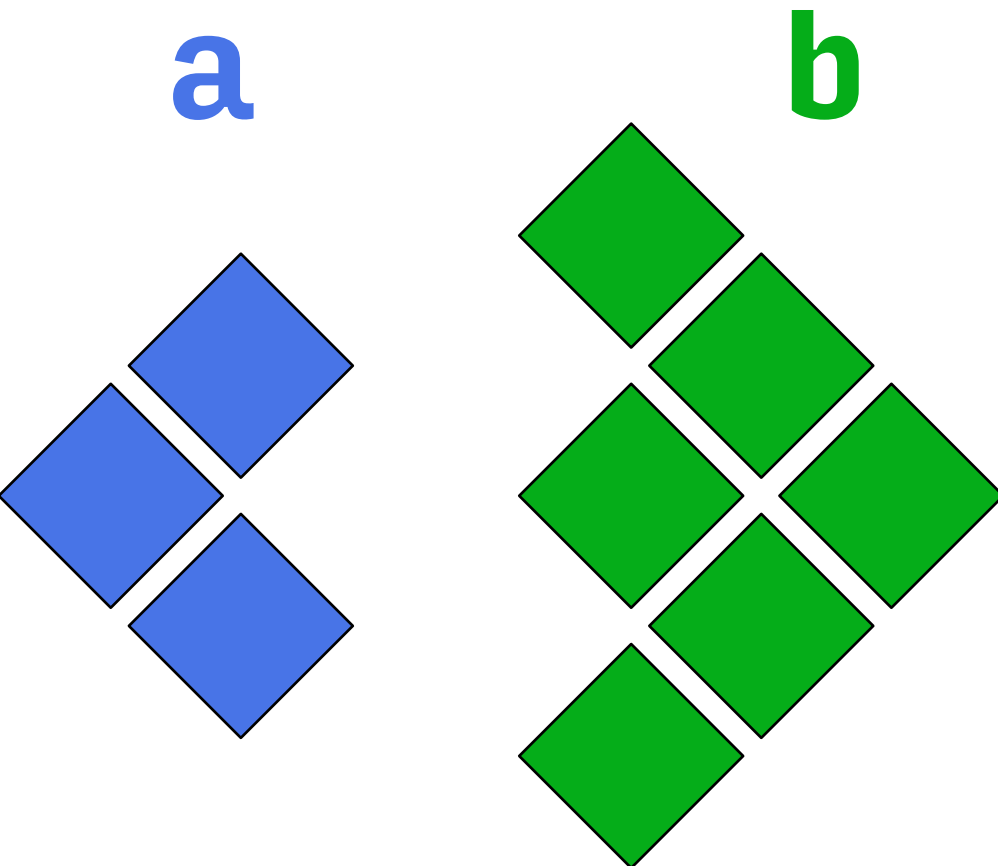
$$n^2 + 1$$



$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



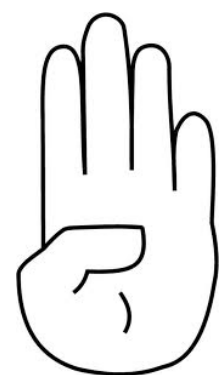
$$n^2$$



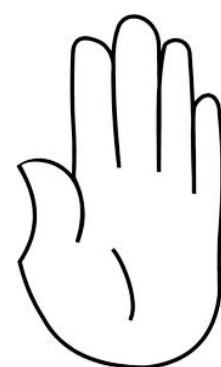
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



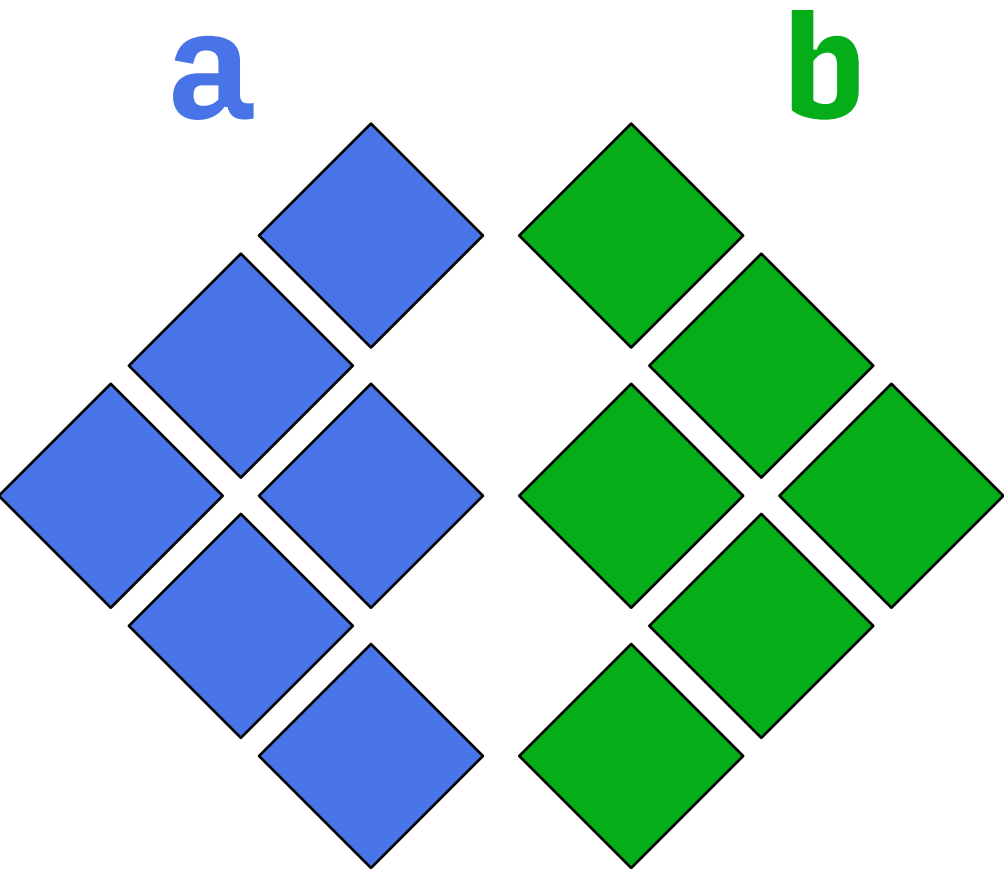
$$n^2 + 1$$



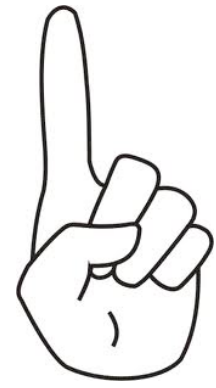
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



$$n^2$$



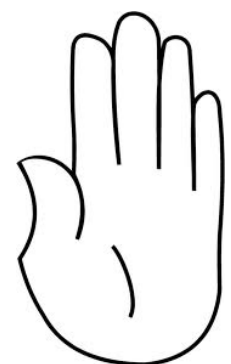
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



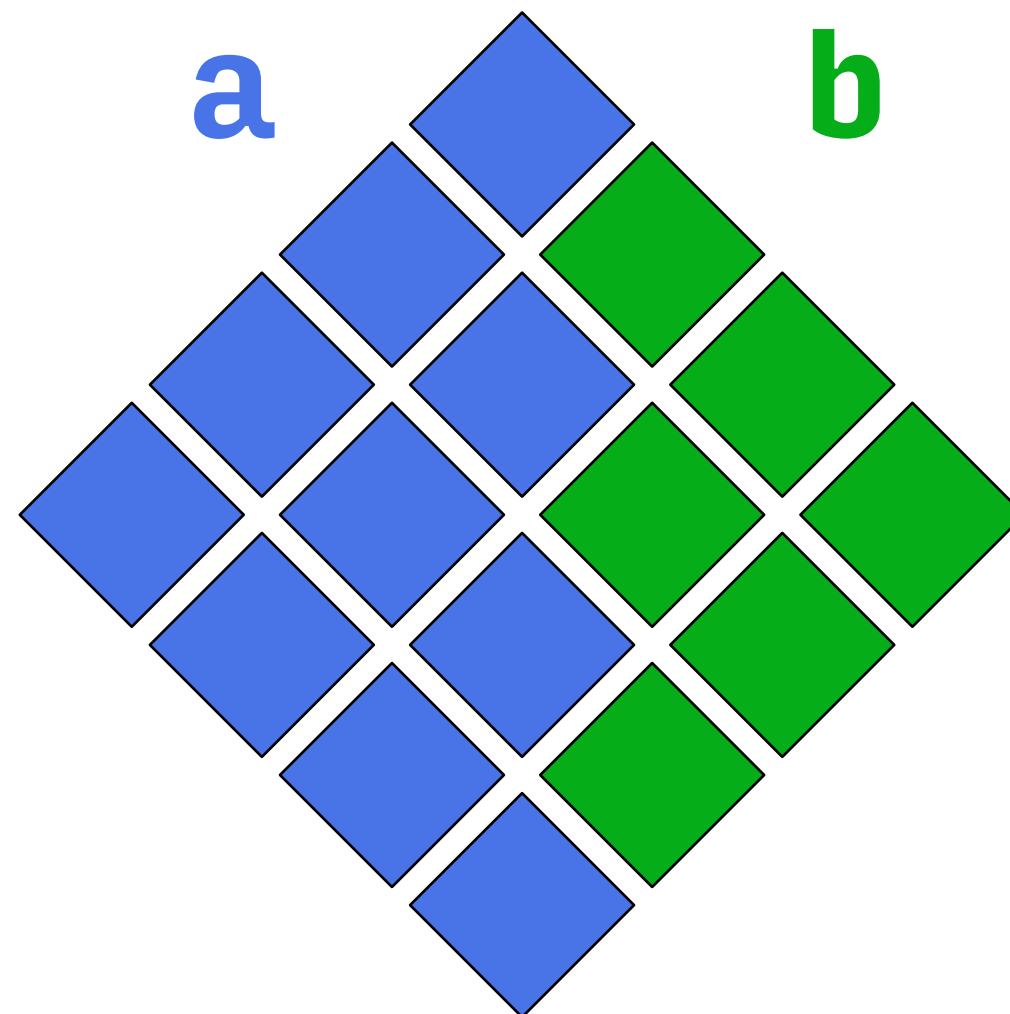
$$n^2 + 1$$



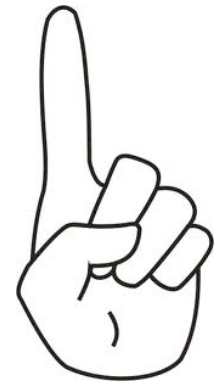
$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



$$n^2$$



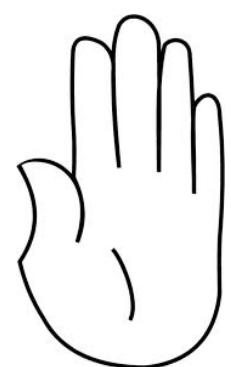
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



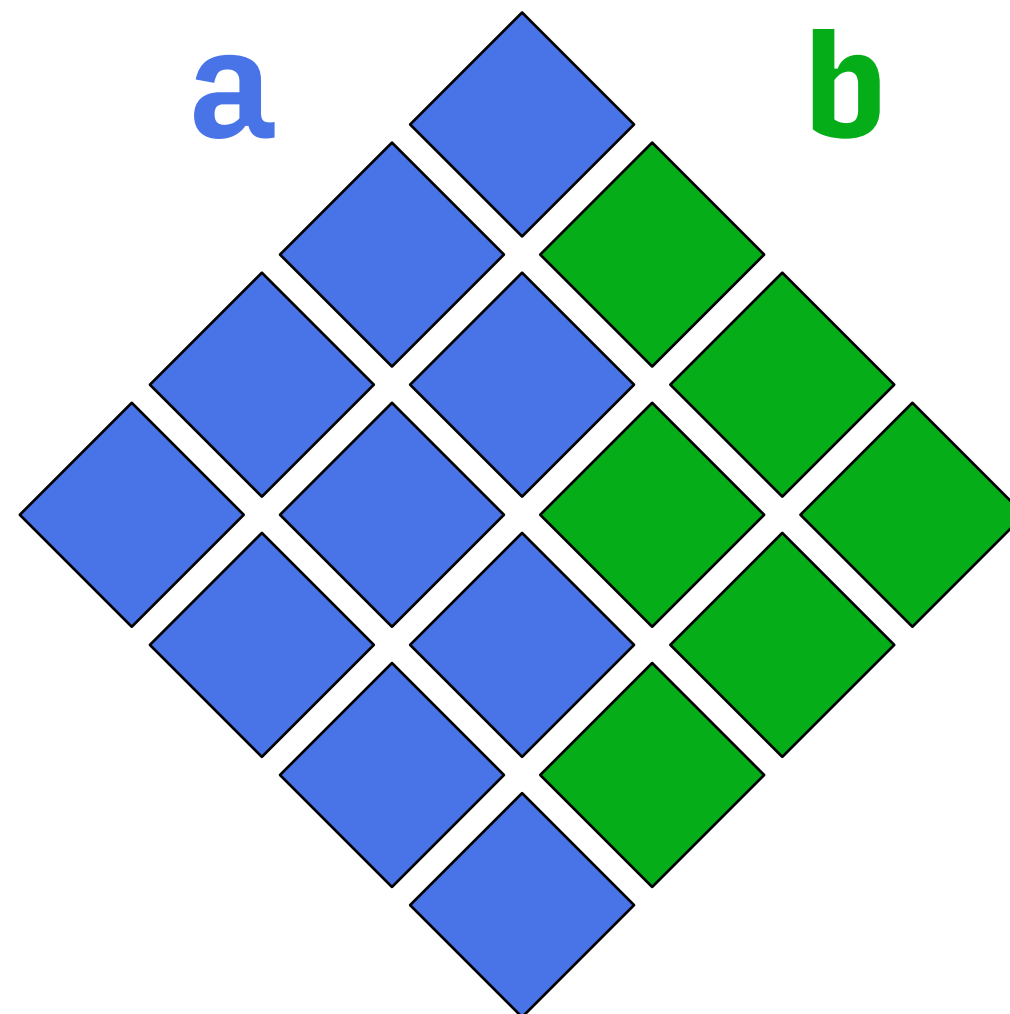
$$n^2 + 1$$



$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



$$n^2$$



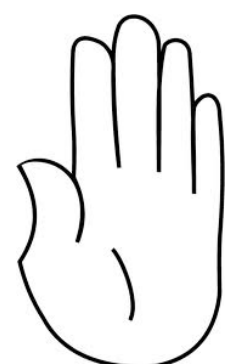
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



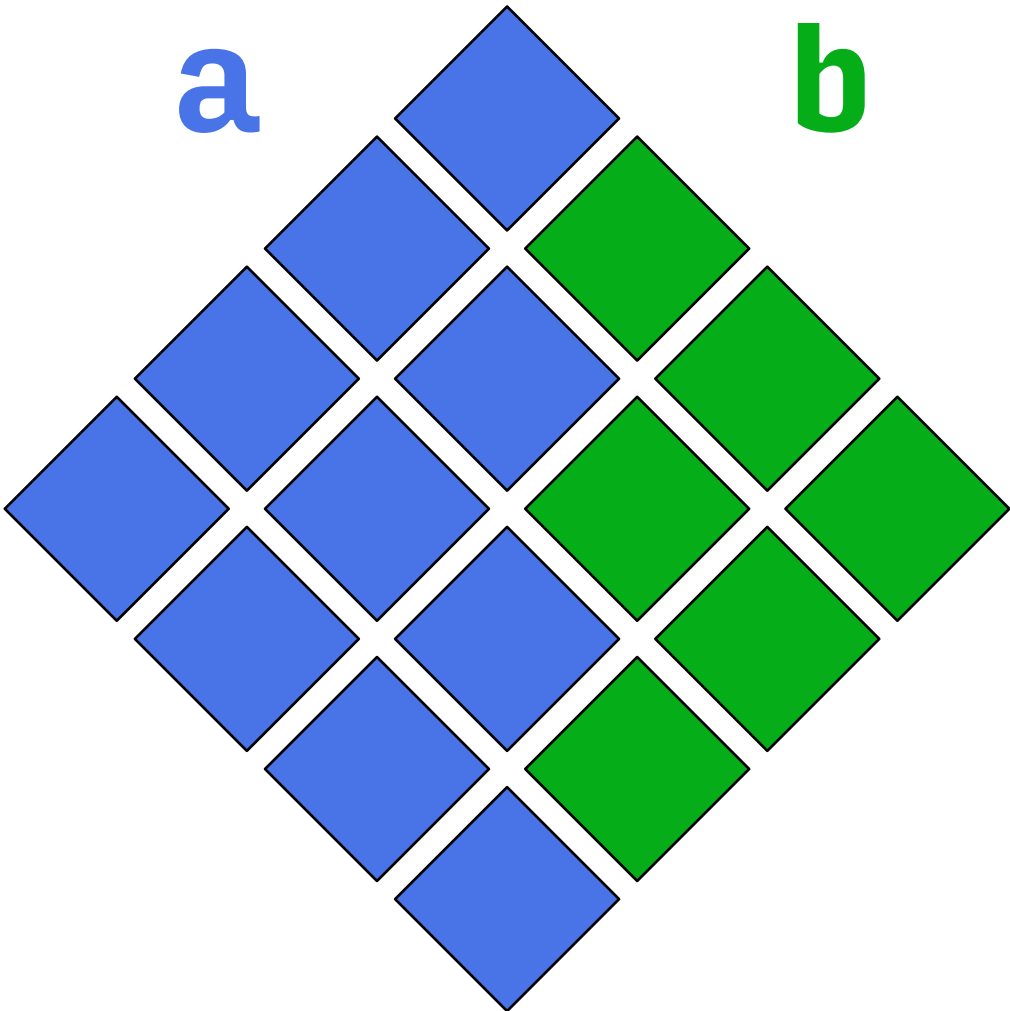
$$n^2 + 1$$



$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Вопрос 1



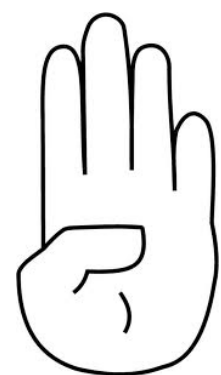
$$n^2$$



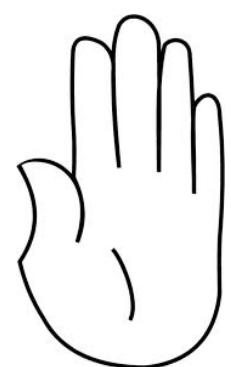
$$(n + 1)^2$$



$$2 \cdot (n + 1)$$



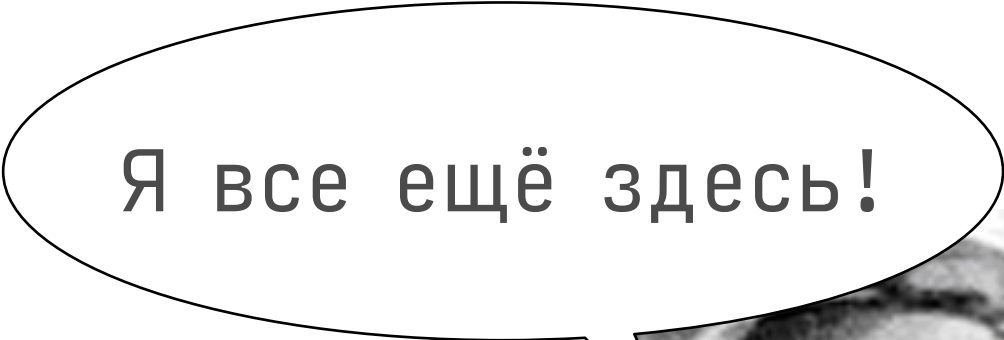
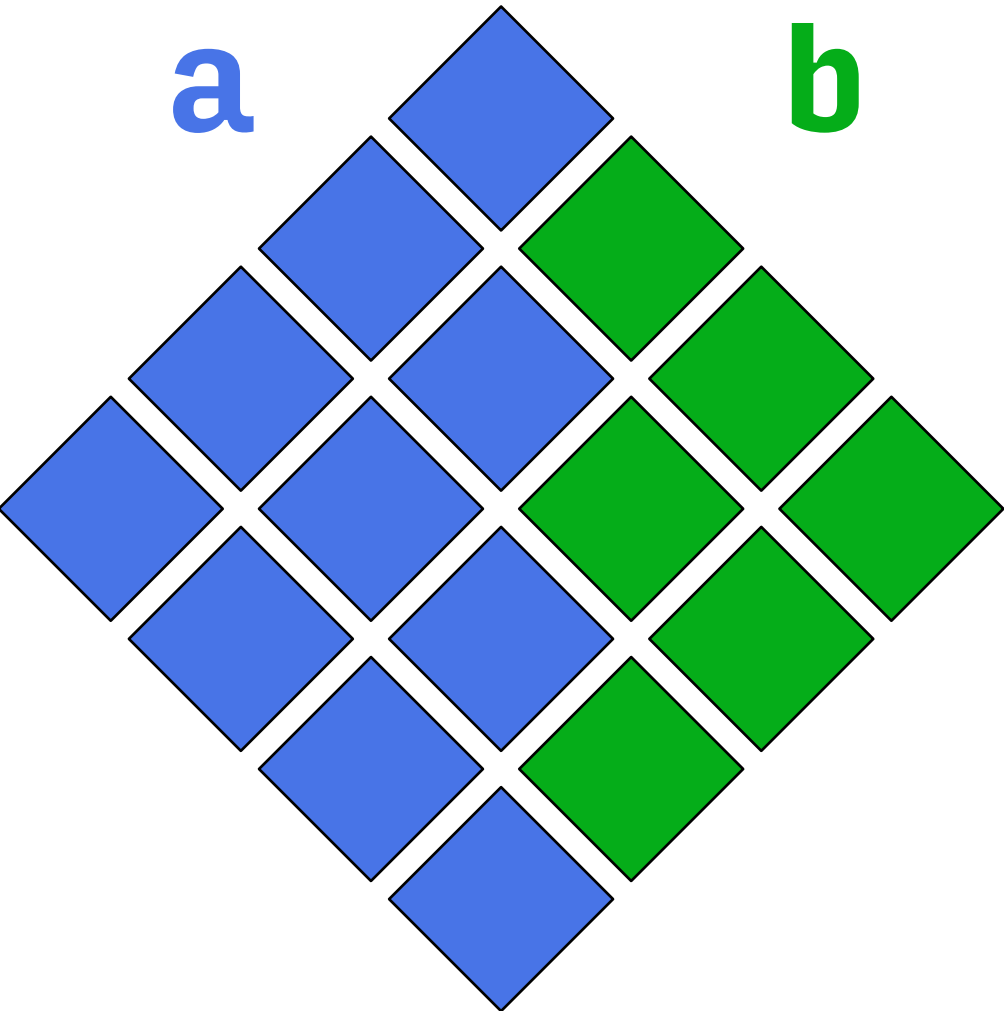
$$n^2 + 1$$



$$n \cdot (n + 1)$$

Что вычисляет эта функция?

```
def pyramid(n):  
    a, b, total = 0, n, 0  
    while b:  
        a, b = a + 1, b - 1  
        total = total + a + b  
    return total
```



Проектирование функций

Характеристики функций

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

Область значений функции (range) — множество возможных выходных значений.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

Область значений функции (range) — множество возможных выходных значений.

Поведение чистой функции — связь между входом и выходом.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает x*x."""
```

Область значений функции (range) — множество возможных выходных значений.

Поведение чистой функции — связь между входом и выходом.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает x*x."""
```

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""
```

Область значений функции (range) — множество возможных выходных значений.

Поведение чистой функции — связь между входом и выходом.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает  $x*x$ ."""
```

x — действительное число

Область значений функции (range) — множество возможных выходных значений.

```
def fib(n):  
    """Вычисляет  $n$ -ое число Фибоначчи, для  $n \geq 1$ ."""
```

Поведение чистой функции — связь между входом и выходом.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает  $x*x$ ."""
```

x — действительное число

Область значений функции (range) — множество возможных выходных значений.

*возвращает
неотрицательное
действительное число*

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для  $n \geq 1$ ."""
```

Поведение чистой функции — связь между входом и выходом.

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает  $x*x$ ."""
```

x — действительное число

Область значений функции (range) — множество возможных выходных значений.

*возвращает
неотрицательное
действительное число*

Поведение чистой функции — связь между входом и выходом.

*возвращаемое значение
является квадратом
входного значения*

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для  $n \geq 1$ ."""
```

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает  $x*x$ ."""
```

x — действительное число

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для  $n \geq 1$ ."""
```

n — целое число, большее либо равное 1

Область значений функции (range) — множество возможных выходных значений.

*возвращает
неотрицательное
действительное число*

Поведение чистой функции — связь между входом и выходом.

*возвращаемое значение
является квадратом
входного значения*

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает  $x*x$ ."""
```

x — действительное число

Область значений функции (range) — множество возможных выходных значений.

*возвращает
неотрицательное
действительное число*

Поведение чистой функции — связь между входом и выходом.

*возвращаемое значение
является квадратом
входного значения*

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для  $n \geq 1$ ."""
```

n — целое число, большее либо равное 1

возвращает число Фибоначчи

Характеристики функций

Область определения функции (domain) — множество возможных аргументов.

```
def square(x):  
    """Возвращает x*x."""
```

x — действительное число

Область значений функции (range) — множество возможных выходных значений.

*возвращает
неотрицательное
действительное число*

Поведение чистой функции — связь между входом и выходом.

*возвращаемое значение
является квадратом
входного значения*

```
def fib(n):  
    """Вычисляет n-ое число Фибоначчи, для n >= 1."""
```

n — целое число, большее либо равное 1

возвращает число Фибоначчи

возвращает значение n-го числа Фибоначчи

Руководство по проектированию функций

Руководство по проектированию функций

У одной функции одна задача.

Руководство по проектированию функций

У одной функции одна задача.



Руководство по проектированию функций

У одной функции одна задача.



не



Руководство по проектированию функций

У одной функции одна задача.



не



Don't repeat yourself (DRY) — не повторяйся.

Напиши один раз, вызывай много раз.

Руководство по проектированию функций

У одной функции одна задача.

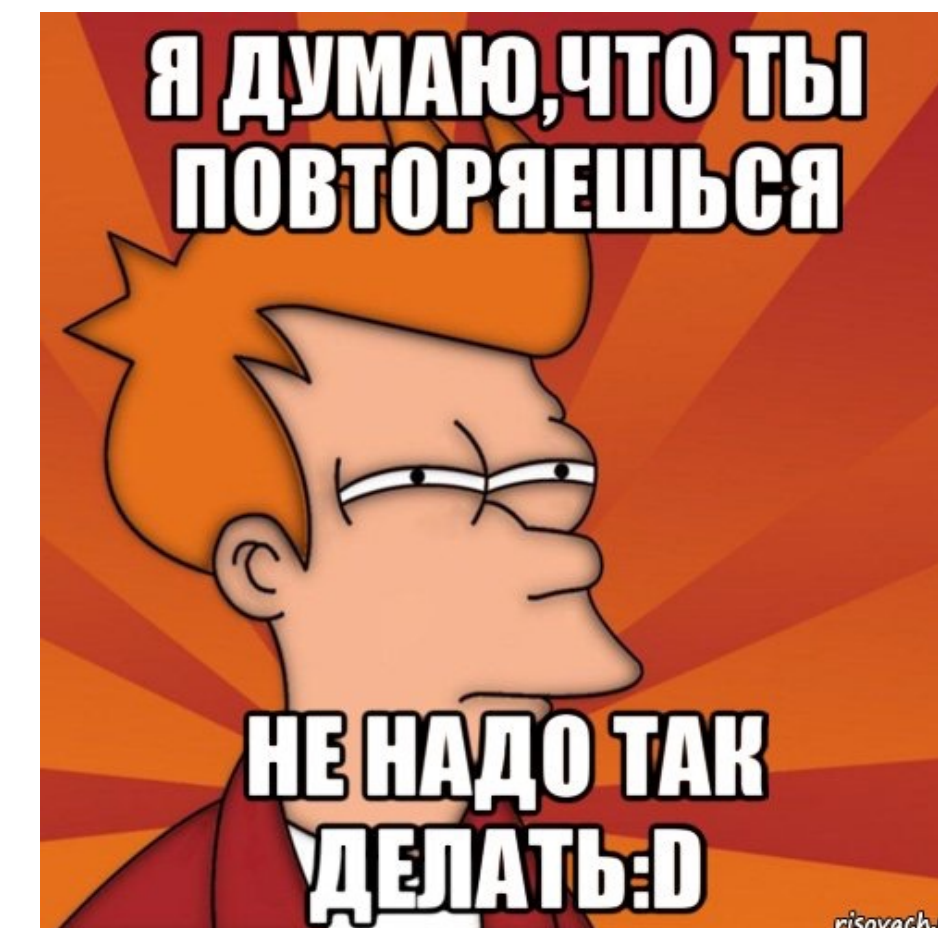


не



Don't repeat yourself (DRY) — не повторяйся.

Напиши один раз, вызывай много раз.



Руководство по проектированию функций

У одной функции одна задача.

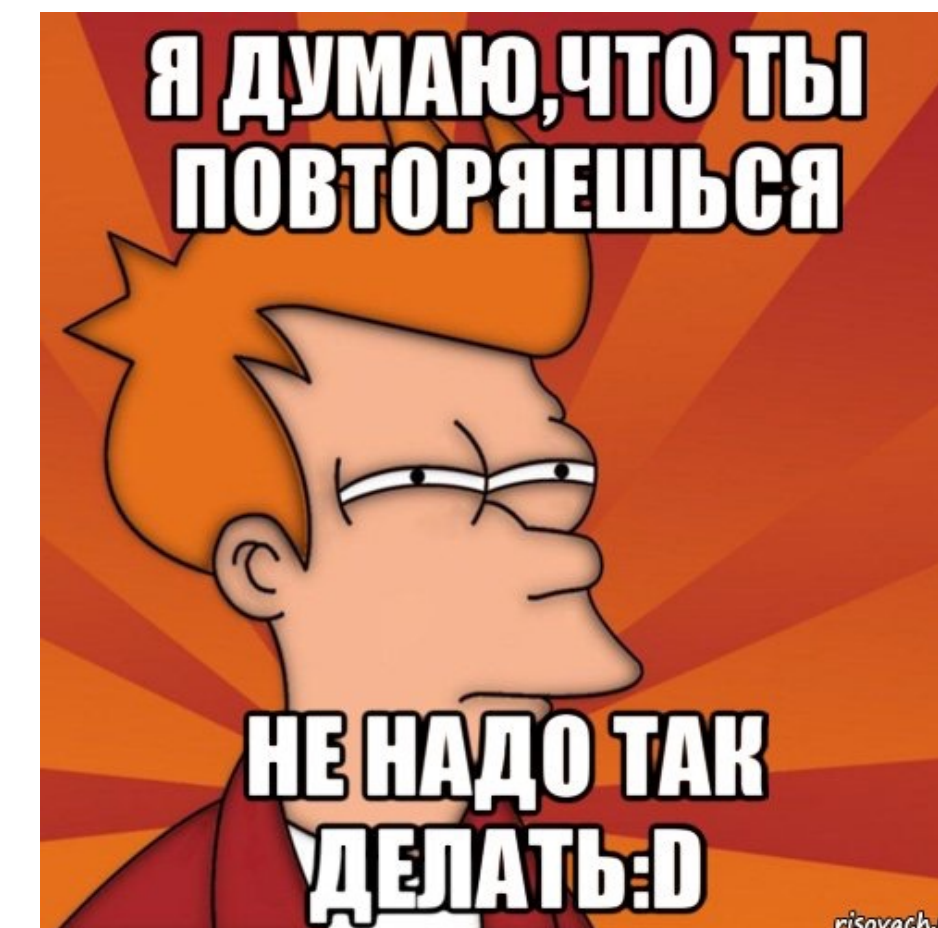


не



Don't repeat yourself (DRY) — не повторяйся.

Напиши один раз, вызывай много раз.



Определяй функции в общем виде.

Руководство по проектированию функций

У одной функции одна задача.



не



Don't repeat yourself (DRY) — не повторяйся.

Напиши один раз, вызывай много раз.



Определяй функции в общем виде.



Обобщение

Обобщение через аргументы

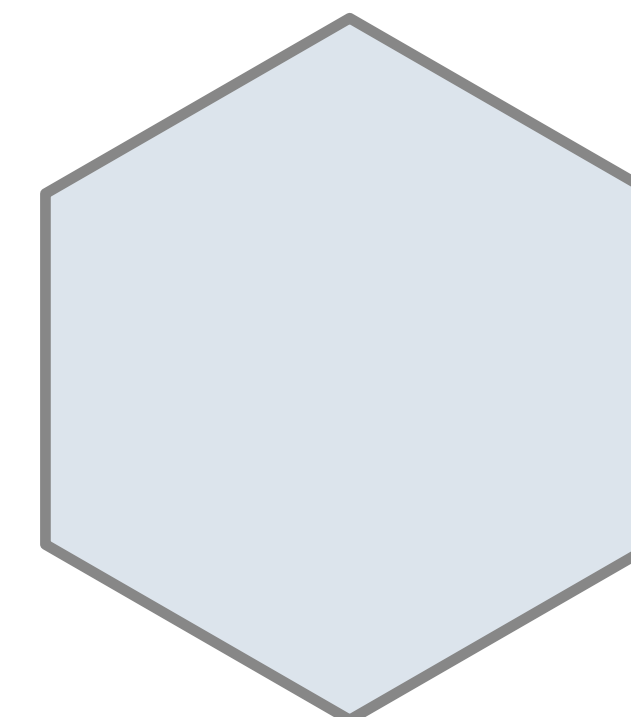
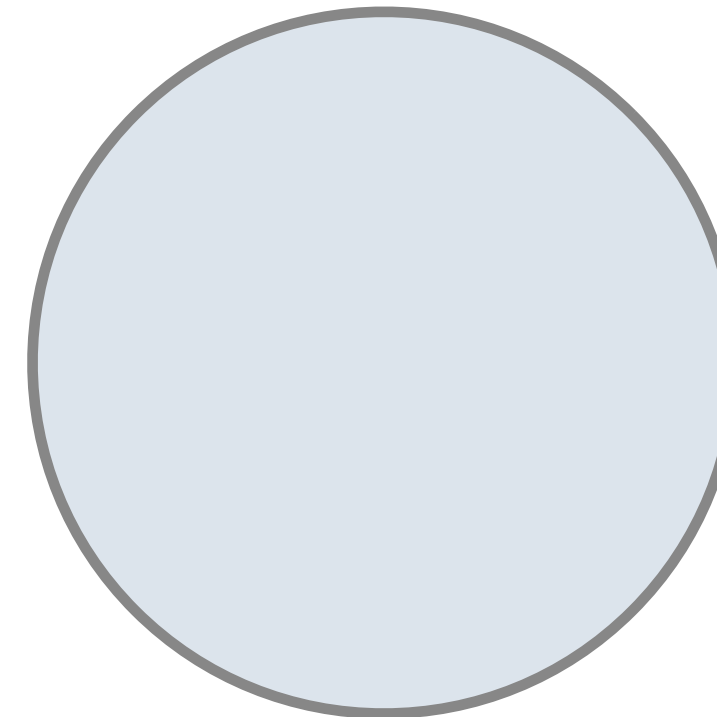
Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

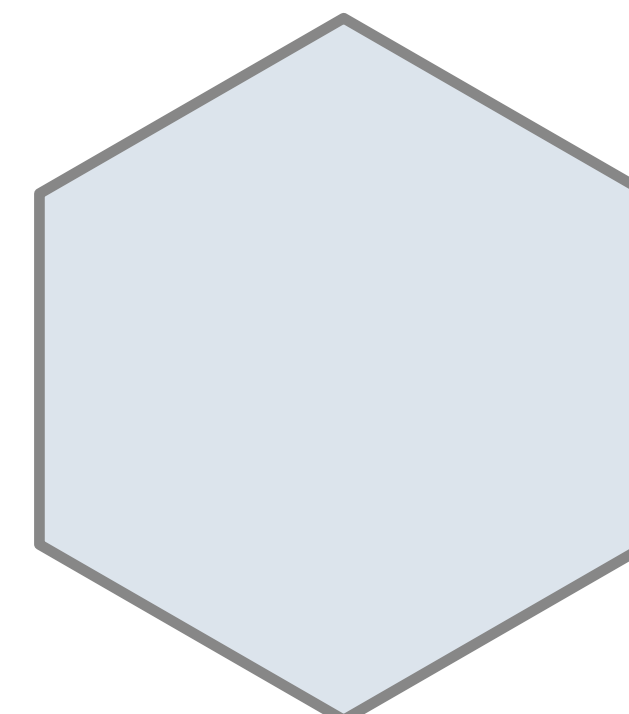
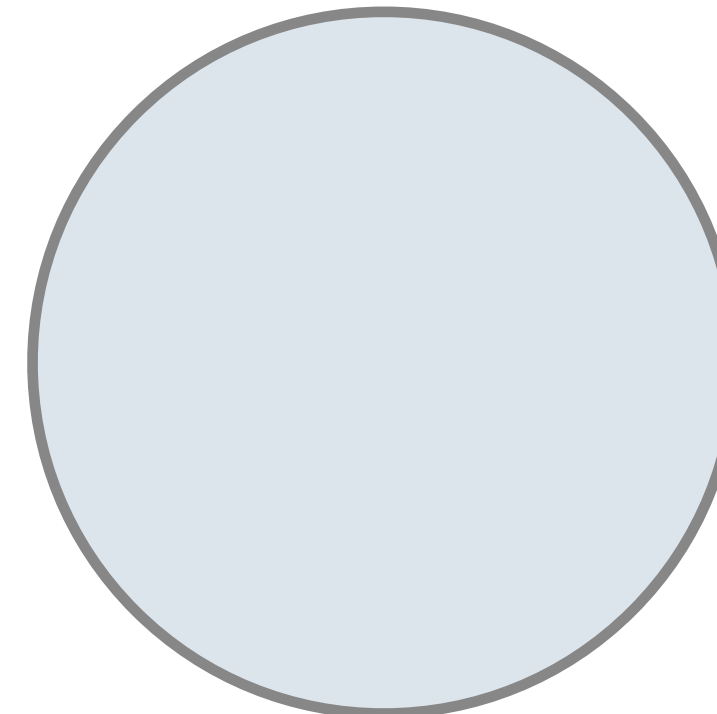
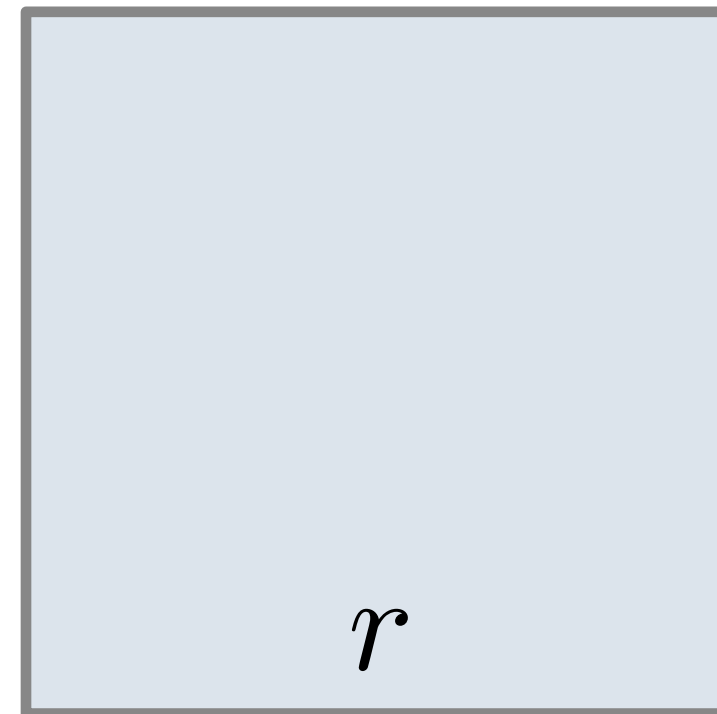
Фигура:



Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

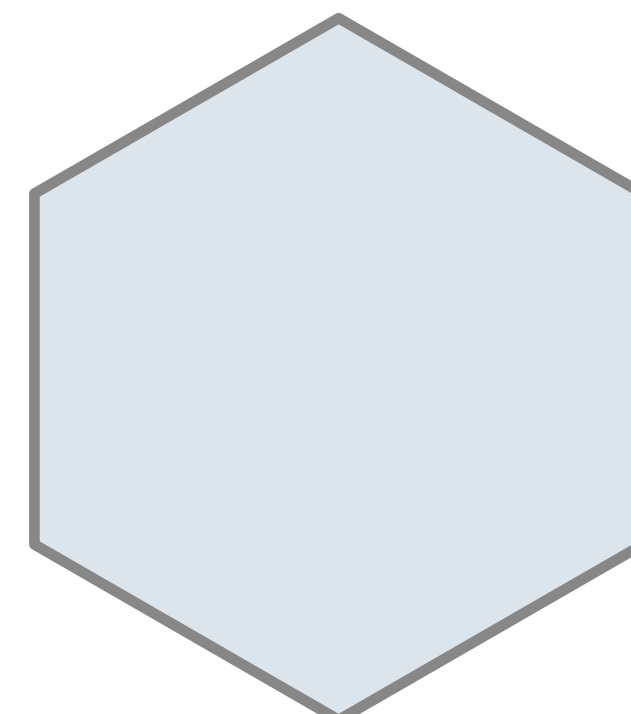
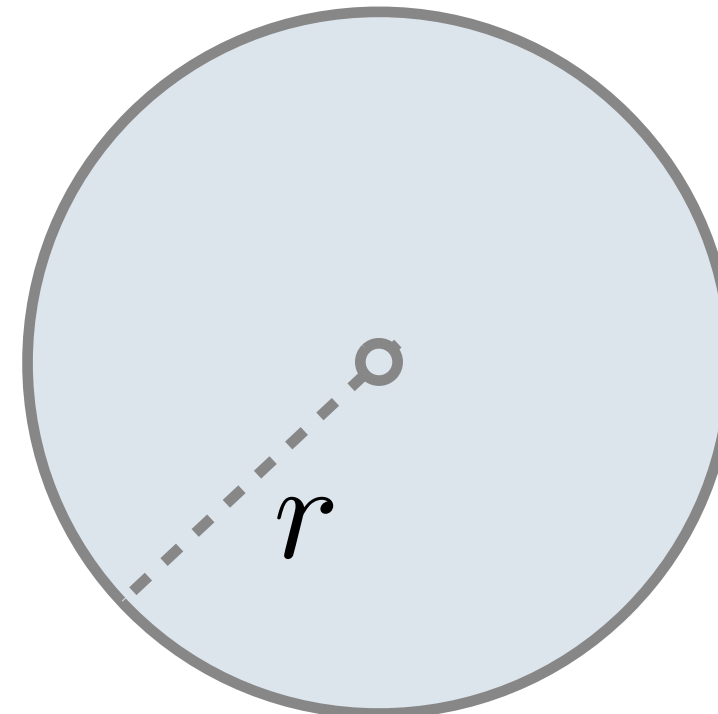
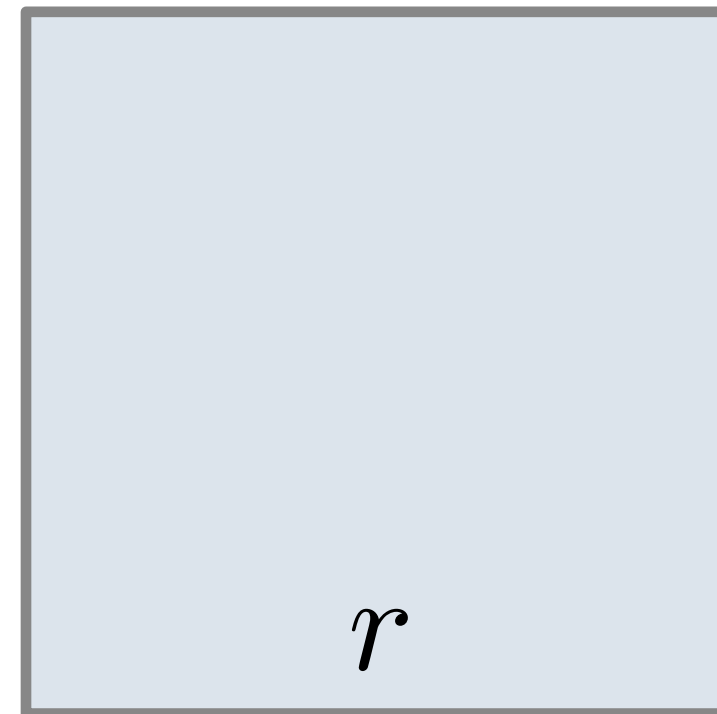
Фигура:



Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

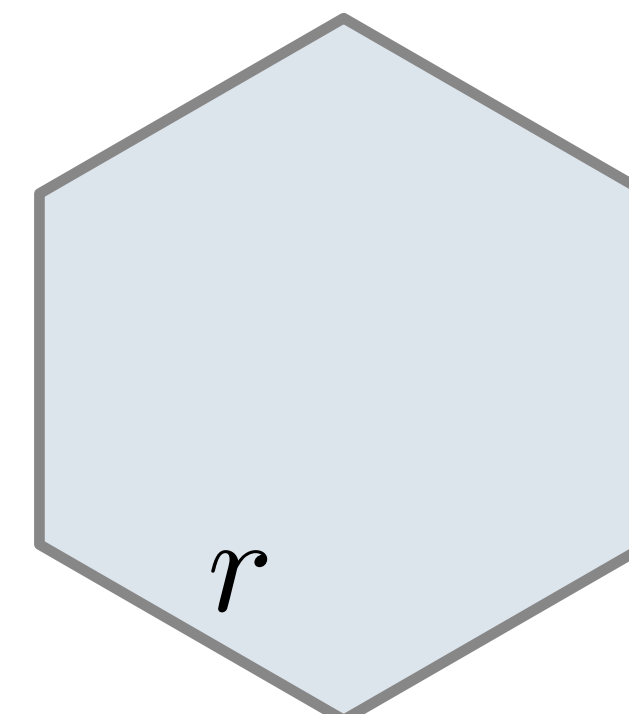
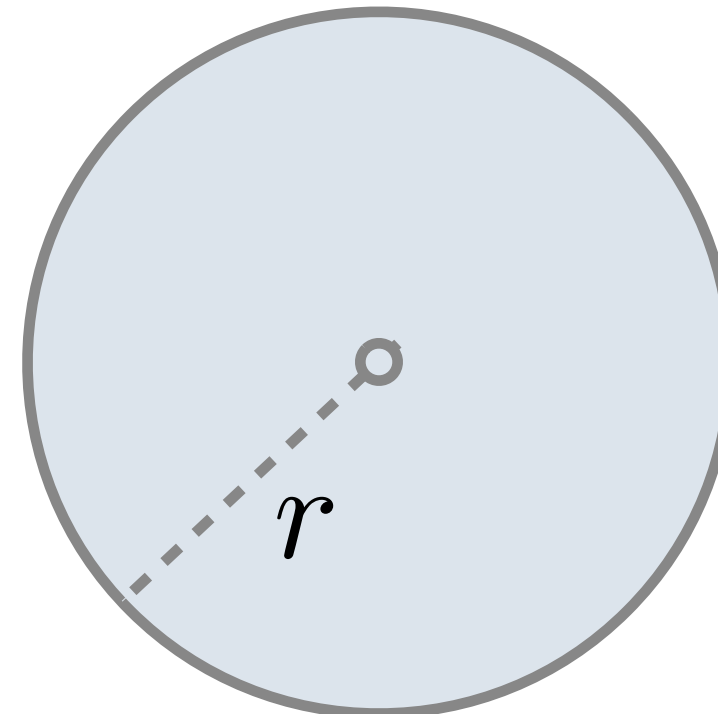
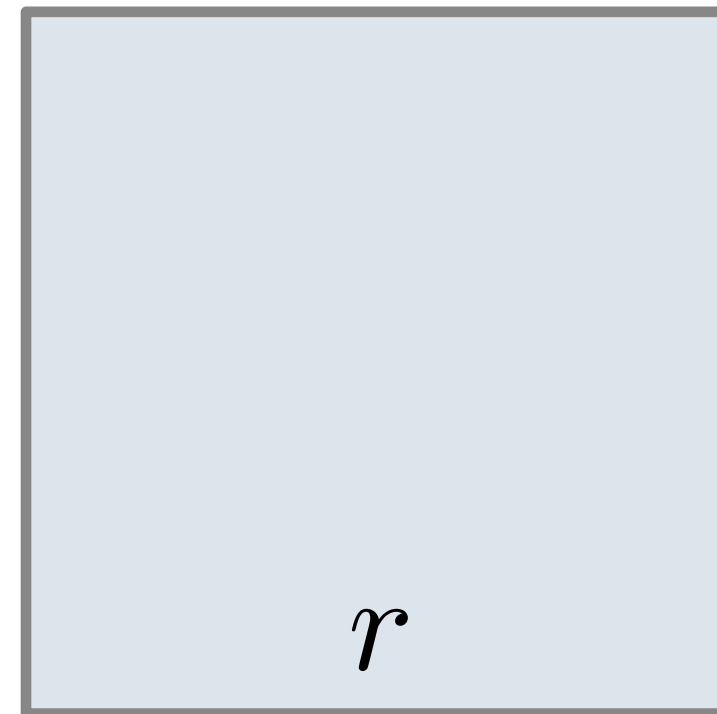
Фигура:



Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

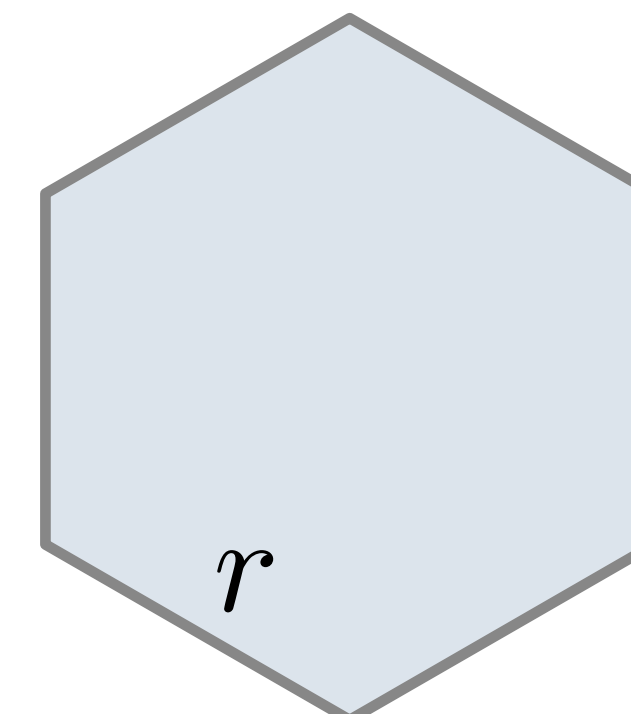
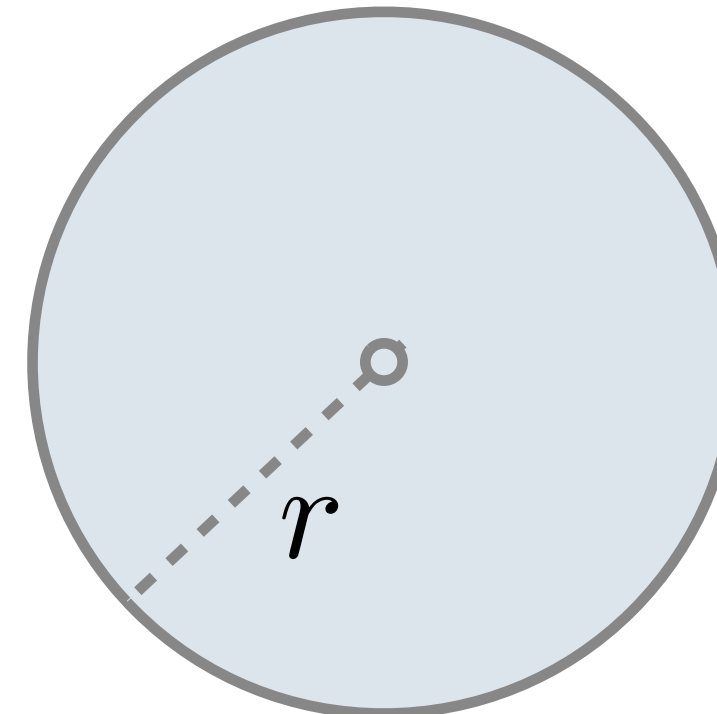
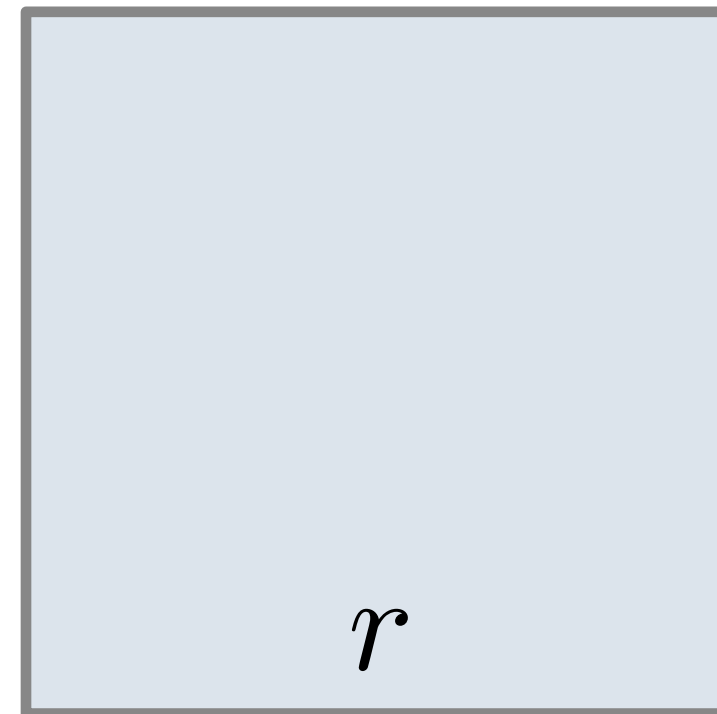
Фигура:



Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:

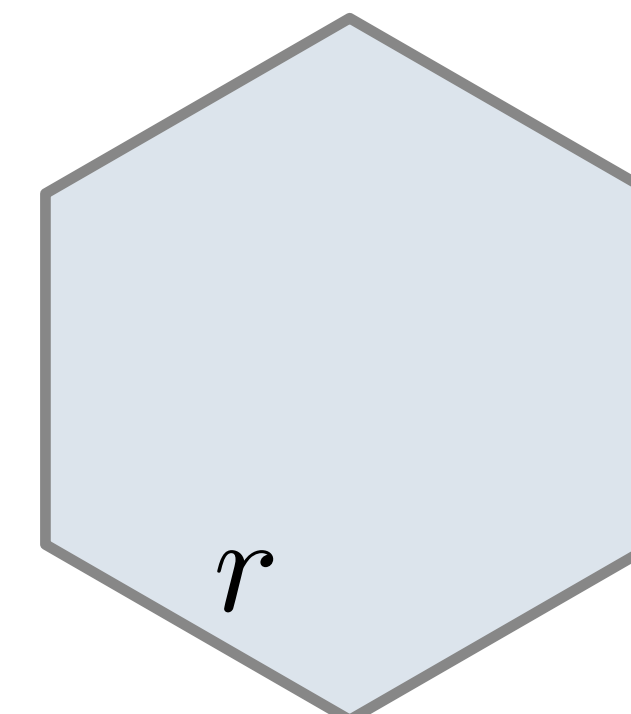
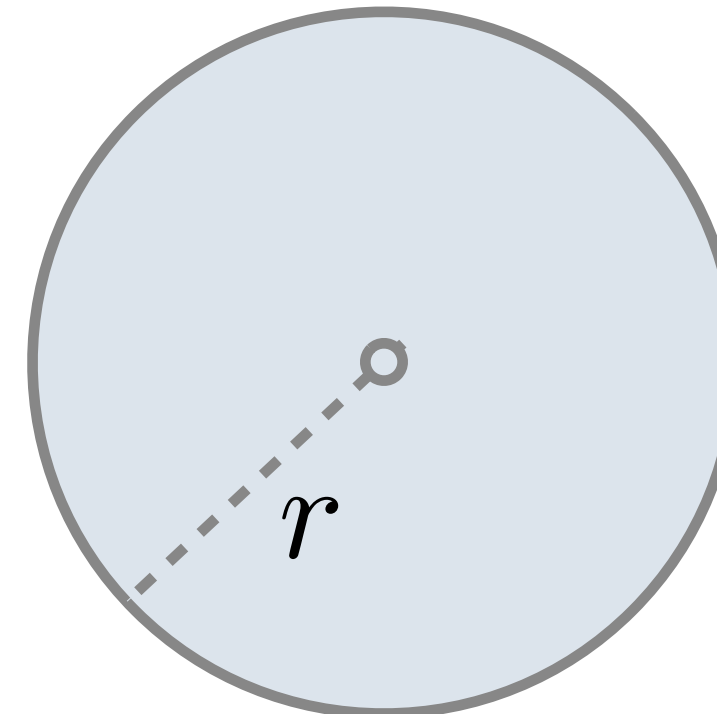
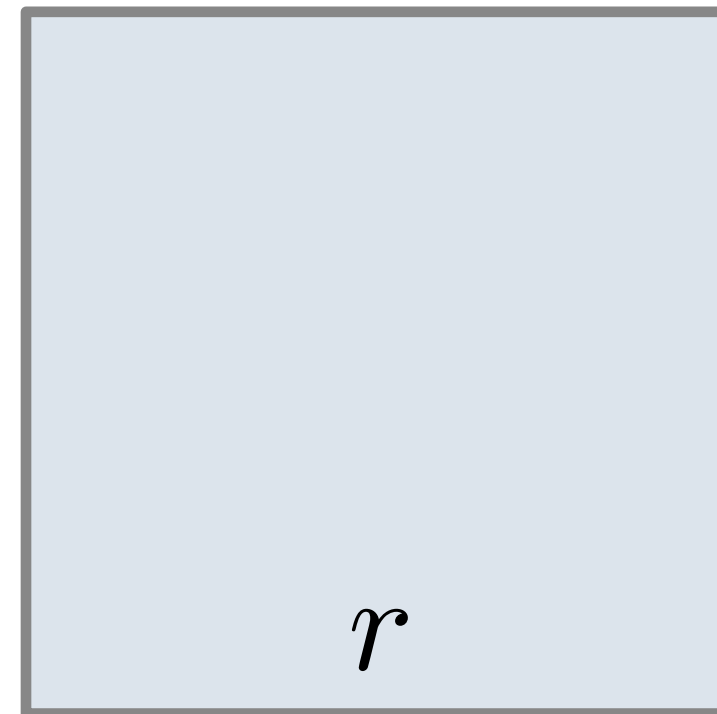


Площадь:

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



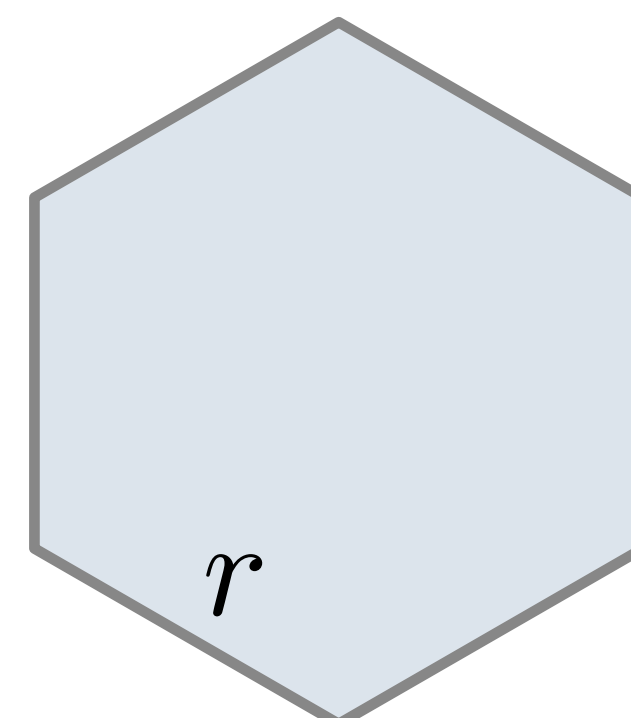
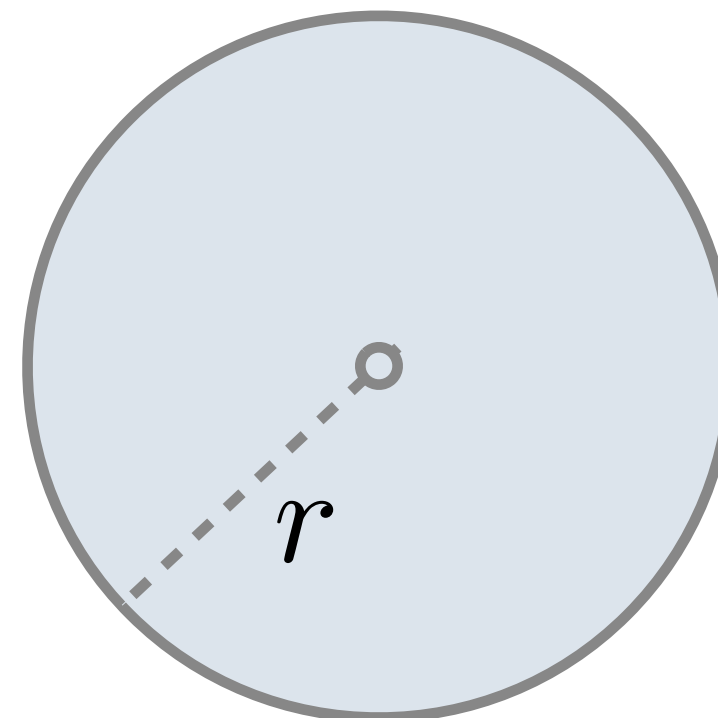
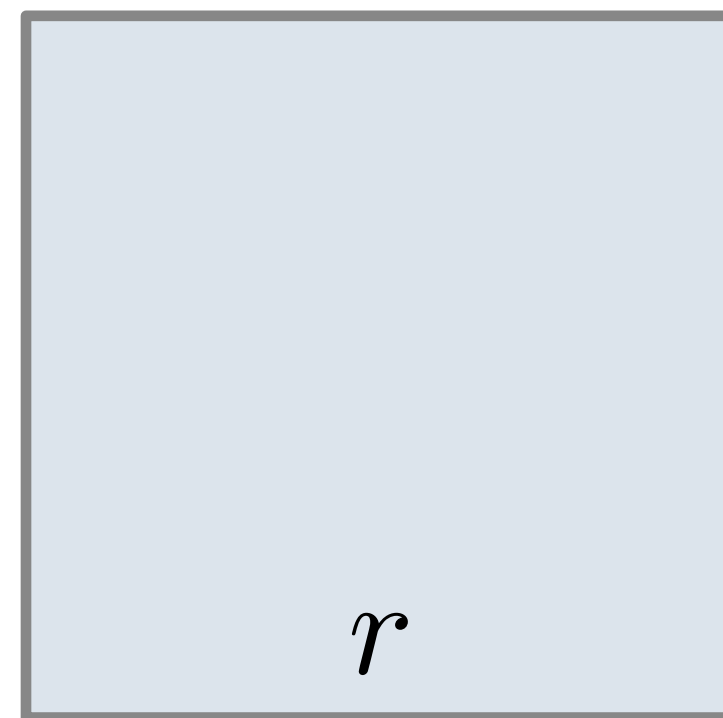
Площадь:

$$r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

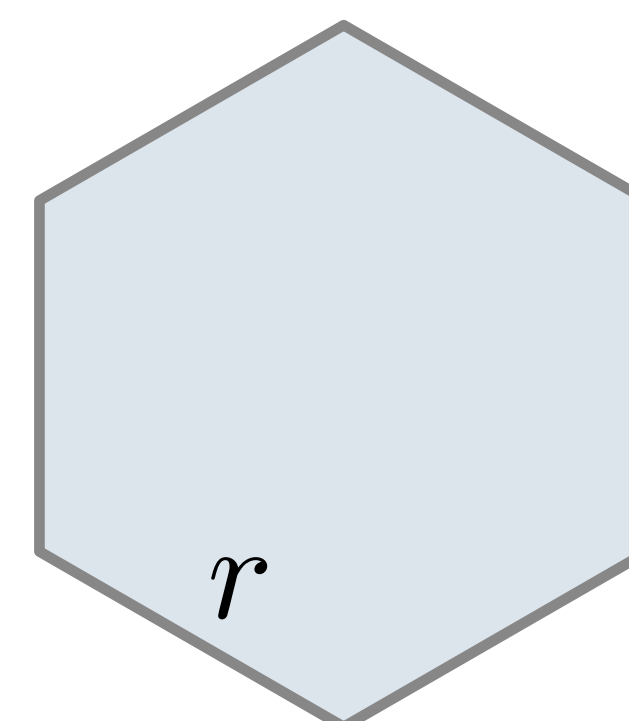
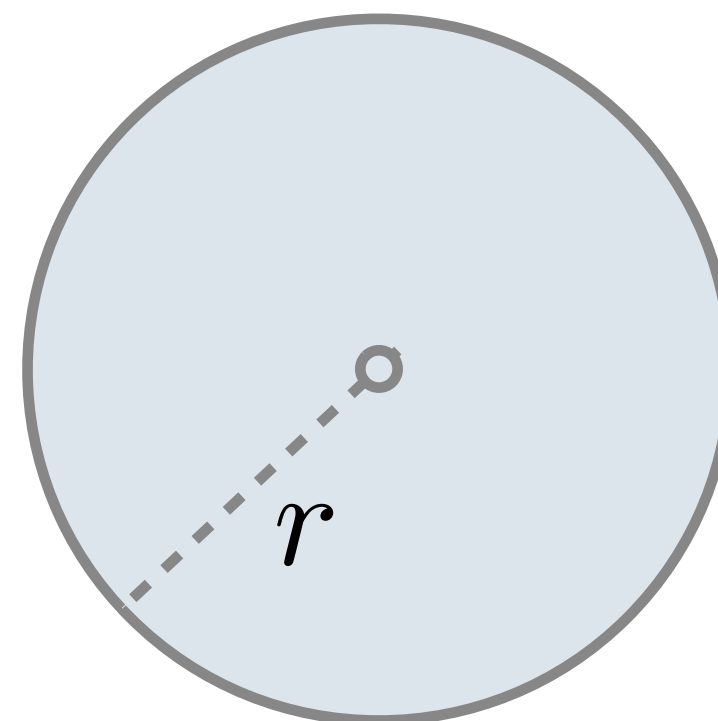
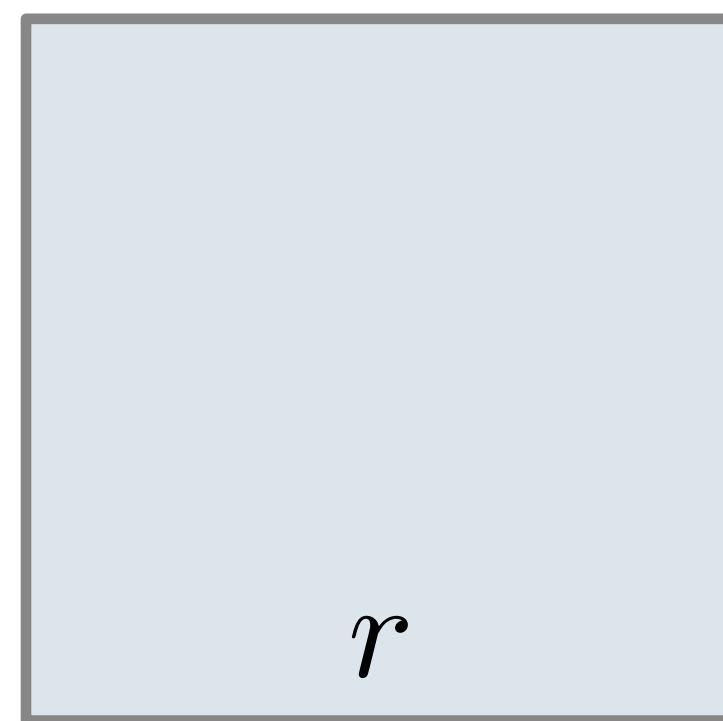
$$r^2$$

$$\pi \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$r^2$$

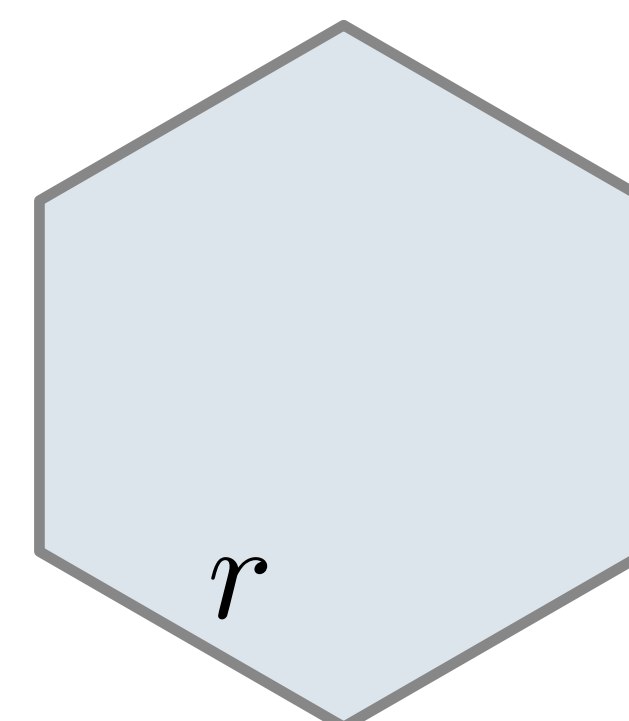
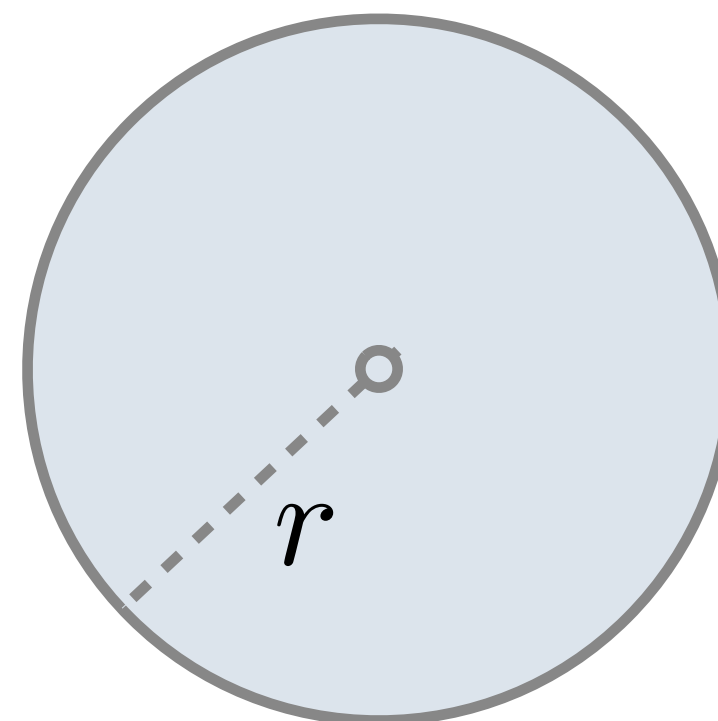
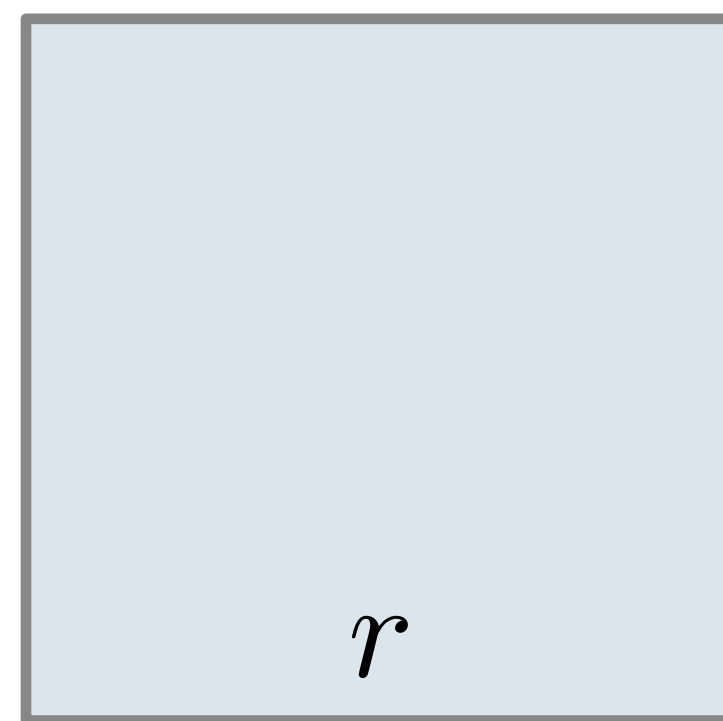
$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$1 \cdot r^2$$

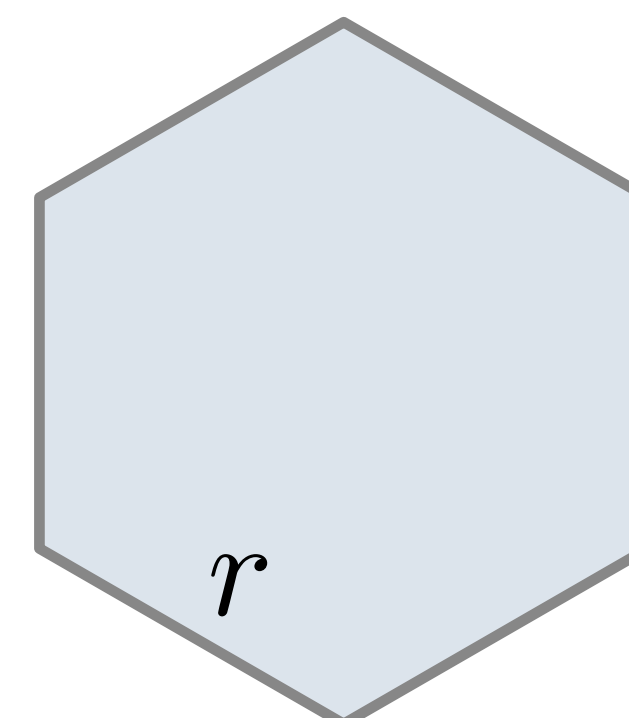
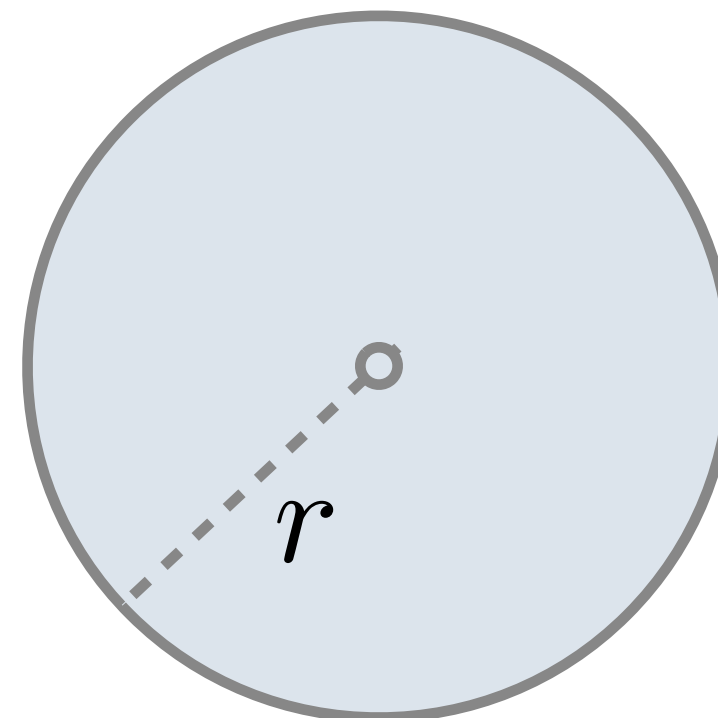
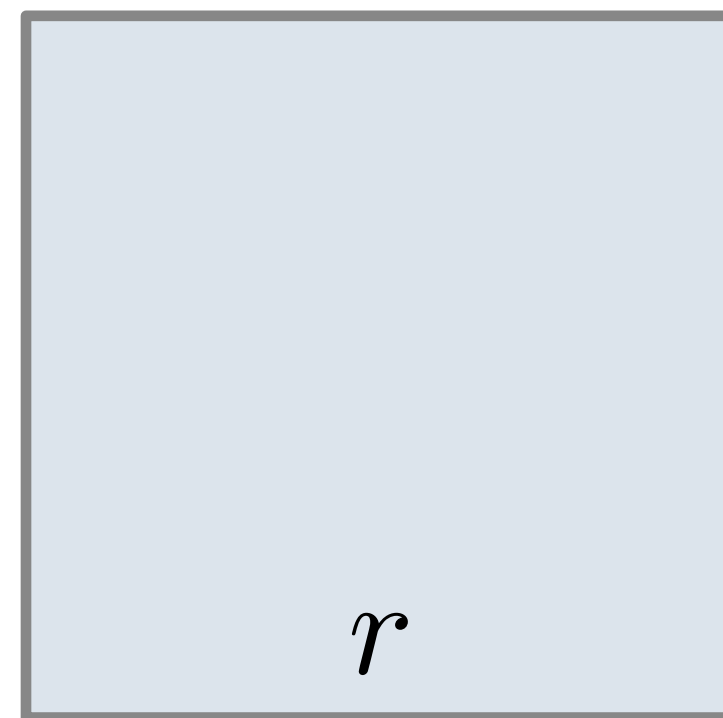
$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$\boxed{1} \cdot r^2$$

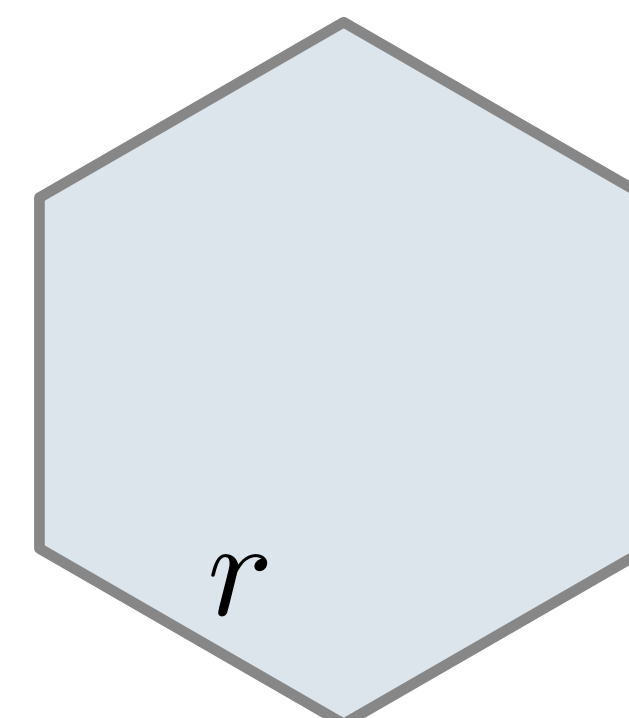
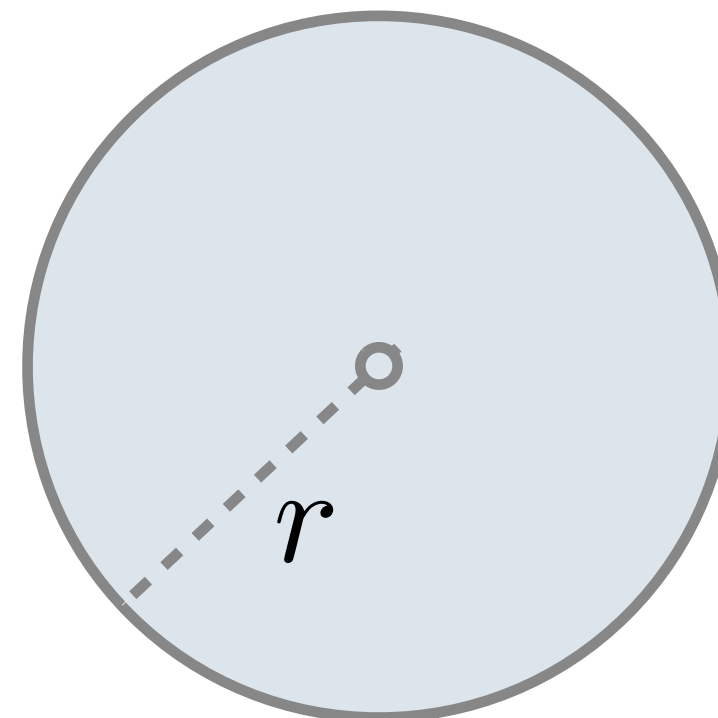
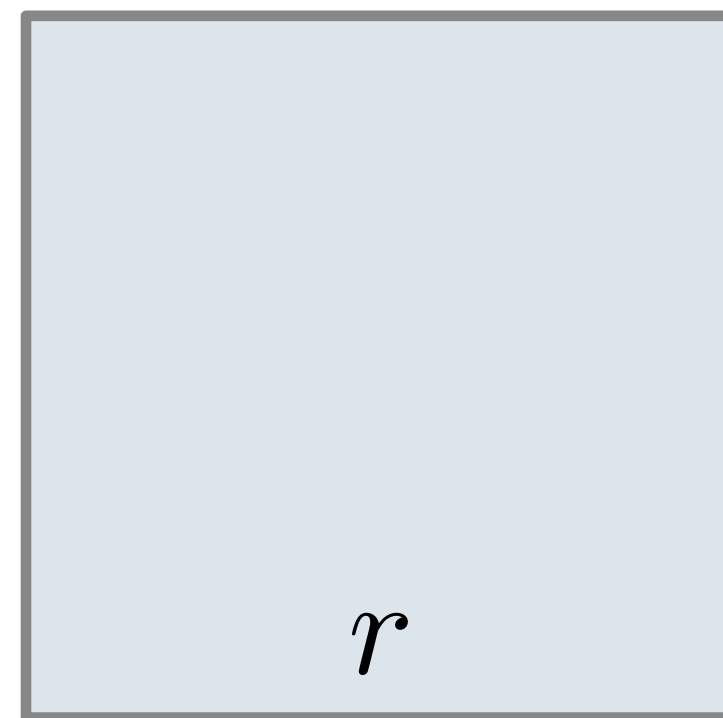
$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$1 \cdot r^2$$

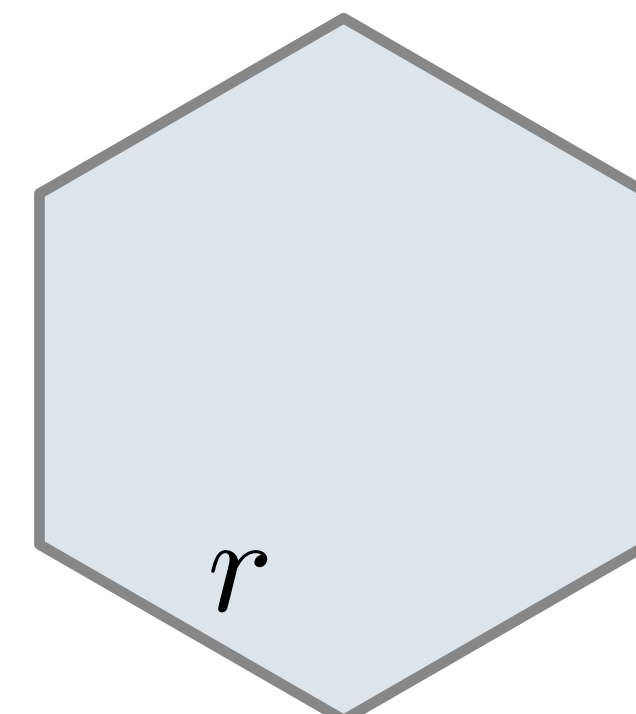
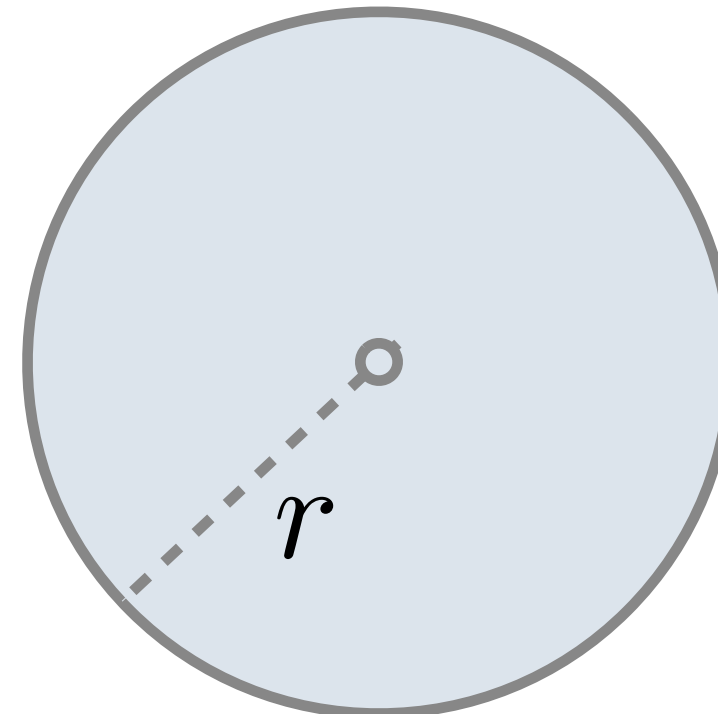
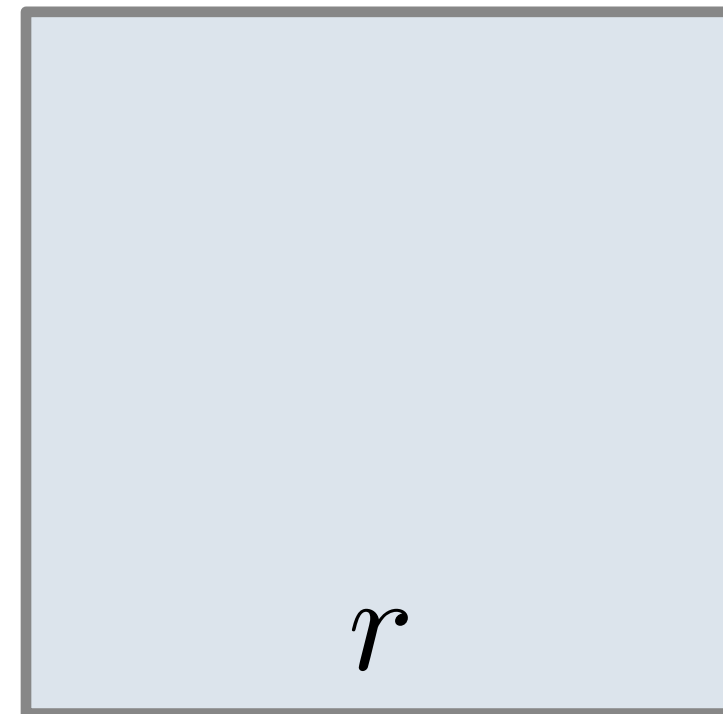
$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$1 \cdot r^2$$

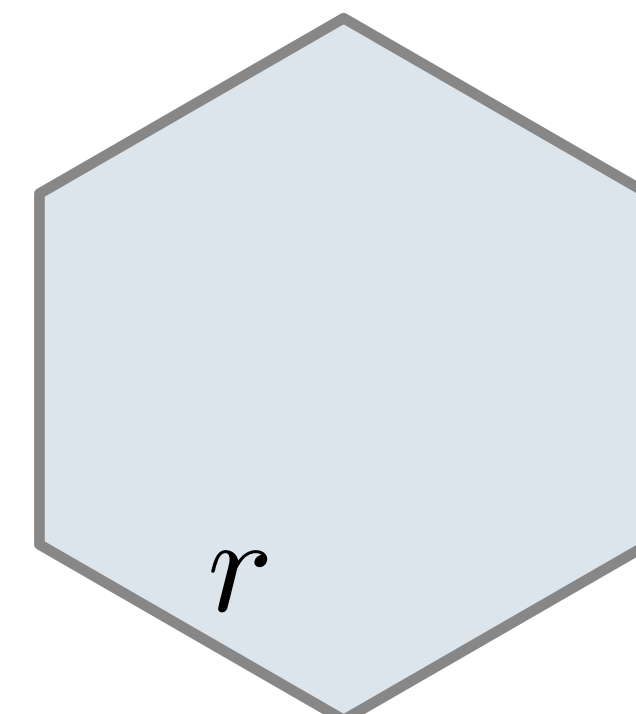
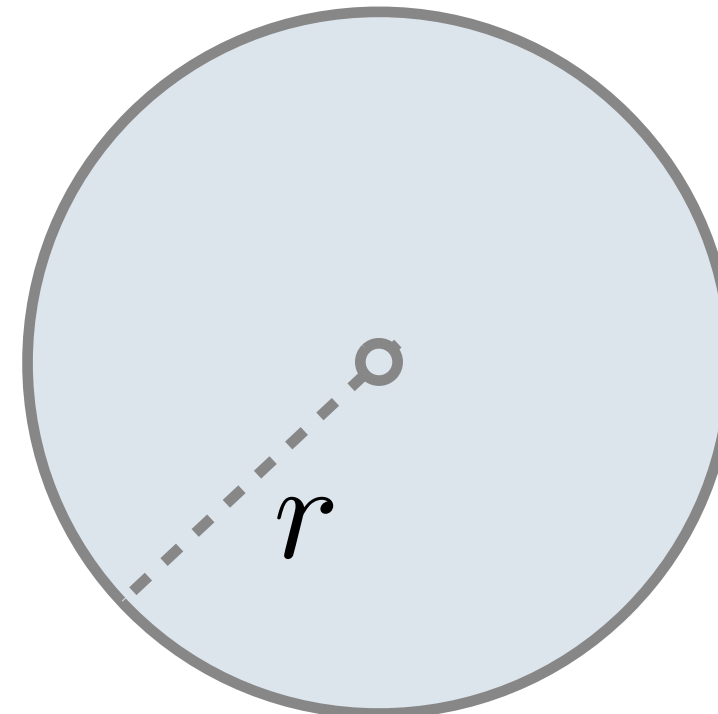
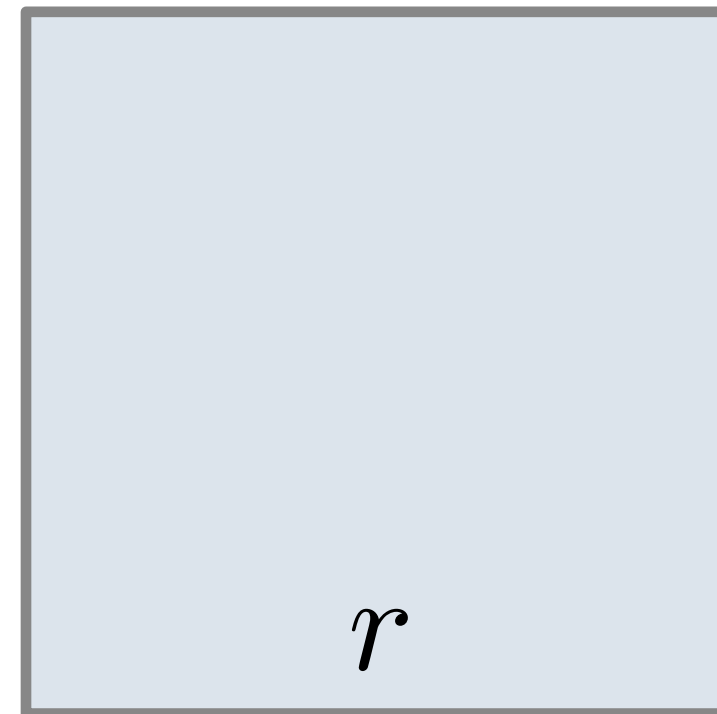
$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$1 \cdot r^2$$

$$\pi \cdot r^2$$

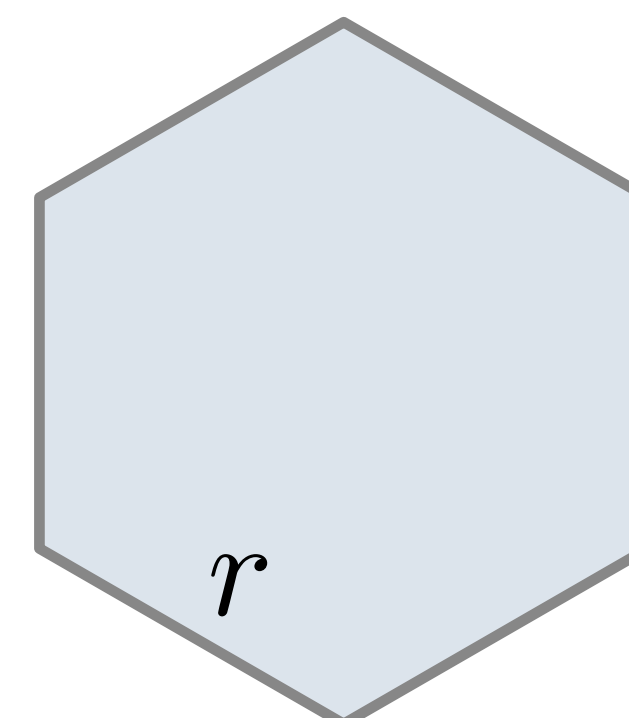
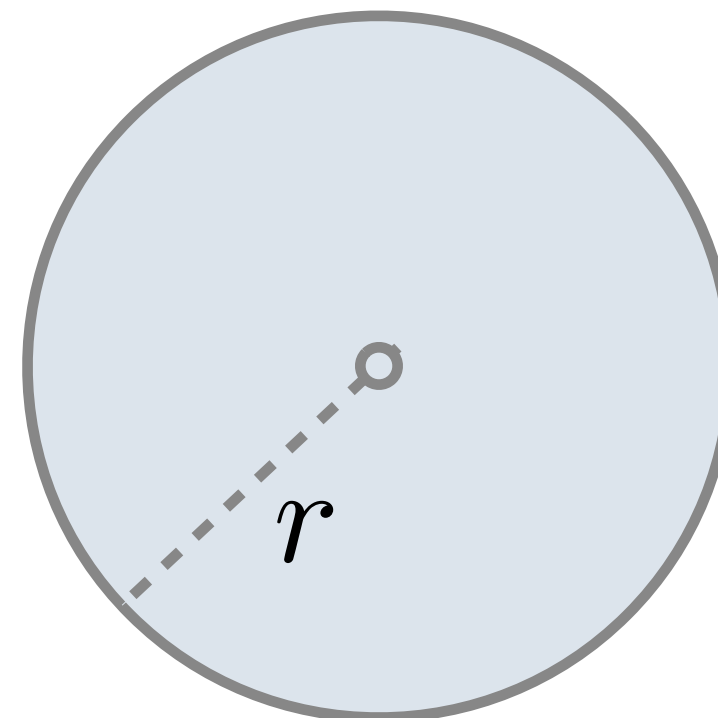
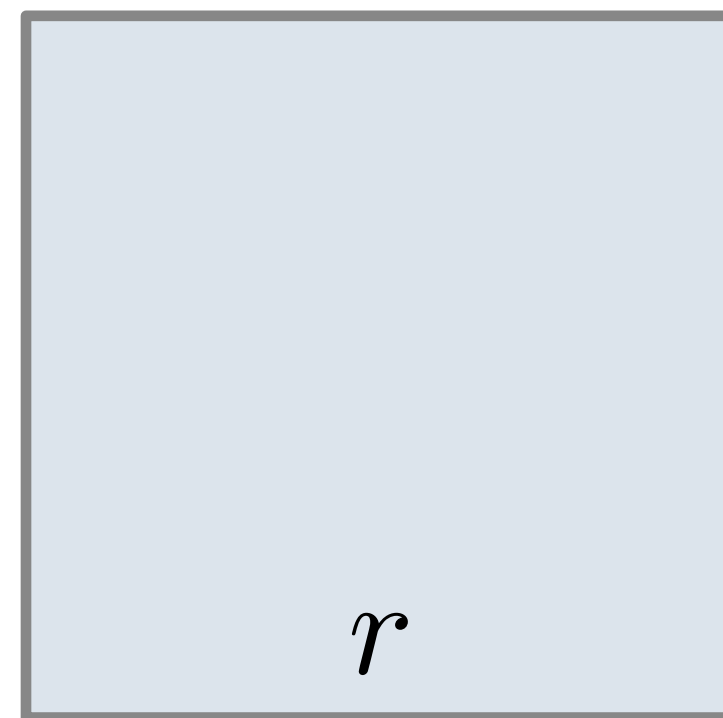
$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Нахождение общей структуры позволяет сделать общую реализацию.

Обобщение через аргументы

Правильные фигуры имеют связь между размером и площадью.

Фигура:



Площадь:

$$1 \cdot r^2$$

$$\pi \cdot r^2$$

$$\frac{3\sqrt{3}}{2} \cdot r^2$$

Нахождение общей структуры позволяет сделать общую реализацию.

(Пример)

Функции высшего порядка

Обобщение через процессы вычисления

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

$$\sum_{k=1}^5 k = 1 + 2 + 3 + 4 + 5 = 15$$

$$\sum_{k=1}^5 k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225$$

$$\sum_{k=1}^5 \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04$$

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

$$\sum_{k=1}^5 k = 1 + 2 + 3 + 4 + 5 = 15$$

$$\sum_{k=1}^5 k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225$$

$$\sum_{k=1}^5 \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04$$

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

$$\sum_{k=1}^5 k = 1 + 2 + 3 + 4 + 5 = 15$$

$$\sum_{k=1}^5 k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225$$

$$\sum_{k=1}^5 \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04$$

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

$$\sum_{k=1}^5 k = 1 + 2 + 3 + 4 + 5 = 15$$

$$\sum_{k=1}^5 k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225$$

$$\sum_{k=1}^5 \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04$$

Обобщение через процессы вычисления

Среди функций общей структурой может быть процесс вычисления, а не число.

$$\sum_{k=1}^5 k = 1 + 2 + 3 + 4 + 5 = 15$$

$$\sum_{k=1}^5 k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225$$

$$\sum_{k=1}^5 \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04$$

(Пример)

Пример со сложением

```
def cube(k):  
    return pow(k, 3)  
  
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.  
  
    >>> summation(5, cube)  
    225  
    """  
    total, k = 0, 1  
    while k <= n:  
        total, k = total + term(k), k + 1  
    return total
```

Пример со сложением

```
def cube(k):  
    return pow(k, 3)
```

Функция одного аргумента
(не называется «term»)

```
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.  
  
    >>> summation(5, cube)  
    225  
    """  
    total, k = 0, 1  
    while k <= n:  
        total, k = total + term(k), k + 1  
    return total
```


Пример со сложением

```
def cube(k):  
    return pow(k, 3)
```

Функция одного аргумента
(не называется «term»)

```
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.  
  
    >>> summation(5, cube)  
    225  
    """""  
    total, k = 0, 1  
    while k <= n:  
        total, k = total + term(k), k + 1  
    return total
```

Формальный параметр, который будет
связан с этой функцией

Пример со сложением

```
def cube(k):  
    return pow(k, 3)
```

Функция одного аргумента
(не называется «term»)

```
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.
```

Формальный параметр, который будет
связан с этой функцией

```
>>> summation(5, cube)
```

```
225
```

```
"""
```

```
total, k = 0, 1
```

```
while k <= n:
```

```
    total, k = total + term(k), k + 1
```

```
return total
```

Здесь вызывается функция,
связанная с «term»

Пример со сложением

```
def cube(k):  
    return pow(k, 3)
```

Функция одного аргумента
(не называется «term»)

```
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.
```

Формальный параметр, который будет
связан с этой функцией

```
>>> summation(5, cube)
```

```
225
```

```
"""
```

```
total, k = 0, 1
```

```
while k <= n:
```

```
    total, k = total + term(k), k + 1
```

```
return total
```

Функция «cube» передается в
качестве аргумента

Здесь вызывается функция,
связанная с «term»

Пример со сложением

```
def cube(k):  
    return pow(k, 3)
```

Функция одного аргумента
(не называется «term»)

```
def summation(n, term):  
    """Сумма первых n членов (terms) последовательности.
```

Формальный параметр, который будет
связан с этой функцией

```
>>> summation(5, cube)
```

```
225
```

```
"""
```

```
total, k = 0, 1
```

```
while k <= n:
```

```
    total, k = total + term(k), k + 1
```

```
return total
```

Функция «cube» передается в
качестве аргумента

Здесь вызывается функция,
связанная с «term»

0 + 1 + 8 + 27 + 64 + 125

Функции в качестве возвращаемых значений

(Пример)

Локально заданные функции

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.  
  
    >>> add_three = make_adder(3)  
    >>> add_three(4)  
    7  
    """  
    def adder(k):  
        return k + n  
    return adder
```

Локально заданные функции

Функция, заданная внутри тела другой функции, связывается с именем в локальном фрейме.

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.  
  
    >>> add_three = make_adder(3)  
    >>> add_three(4)  
    7  
    """  
    def adder(k):  
        return k + n  
    return adder
```

Локально заданные функции

Функция, заданная внутри тела другой функции, связывается с именем в локальном фрейме.

Функция возвращающая
функцию

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.  
  
    >>> add_three = make_adder(3)  
    >>> add_three(4)  
    7  
    """  
    def adder(k):  
        return k + n  
    return adder
```


Локально заданные функции

Функция, заданная внутри тела другой функции, связывается с именем в локальном фрейме.

Функция возвращающая
функцию

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.  
  
    >>> add_three = make_adder(3)  
    >>> add_three(4)  
    7  
    """""  
    def adder(k):  
        return k + n  
    return adder
```

Имя «add_three» связано с функцией

Локально заданные функции

Функция, заданная внутри тела другой функции, связывается с именем в локальном фрейме.

Функция возвращающая
функцию

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.
```

```
>>> add_three = make_adder(3)  
>>> add_three(4)  
7  
"""
```

Имя «add_three» связано с функцией

```
def adder(k):  
    return k + n  
return adder
```

Инструкция def внутри другой
инструкции def

Локально заданные функции

Функция, заданная внутри тела другой функции, связывается с именем в локальном фрейме.

Функция возвращающая
функцию

```
def make_adder(n):  
    """Возвращает функцию, принимающую один аргумент k и возвращающую k + n.  
  
    >>> add_three = make_adder(3)  
    >>> add_three(4)  
    7  
    """
```

Имя «add_three» связано с функцией

```
def adder(k):  
    return k + n  
return adder
```

Инструкция def внутри другой
инструкции def

Можно обращаться к именам,
определенным в содержащей функции