

# Программирование на Python: графика

1. [Простые программы](#)
2. [Процедуры](#)
3. [Циклы](#)
4. [Штриховка](#)
5. [Закрашивание областей](#)
6. [Построение графиков функций](#)
7. [Анимация](#)
8. [Игры](#)

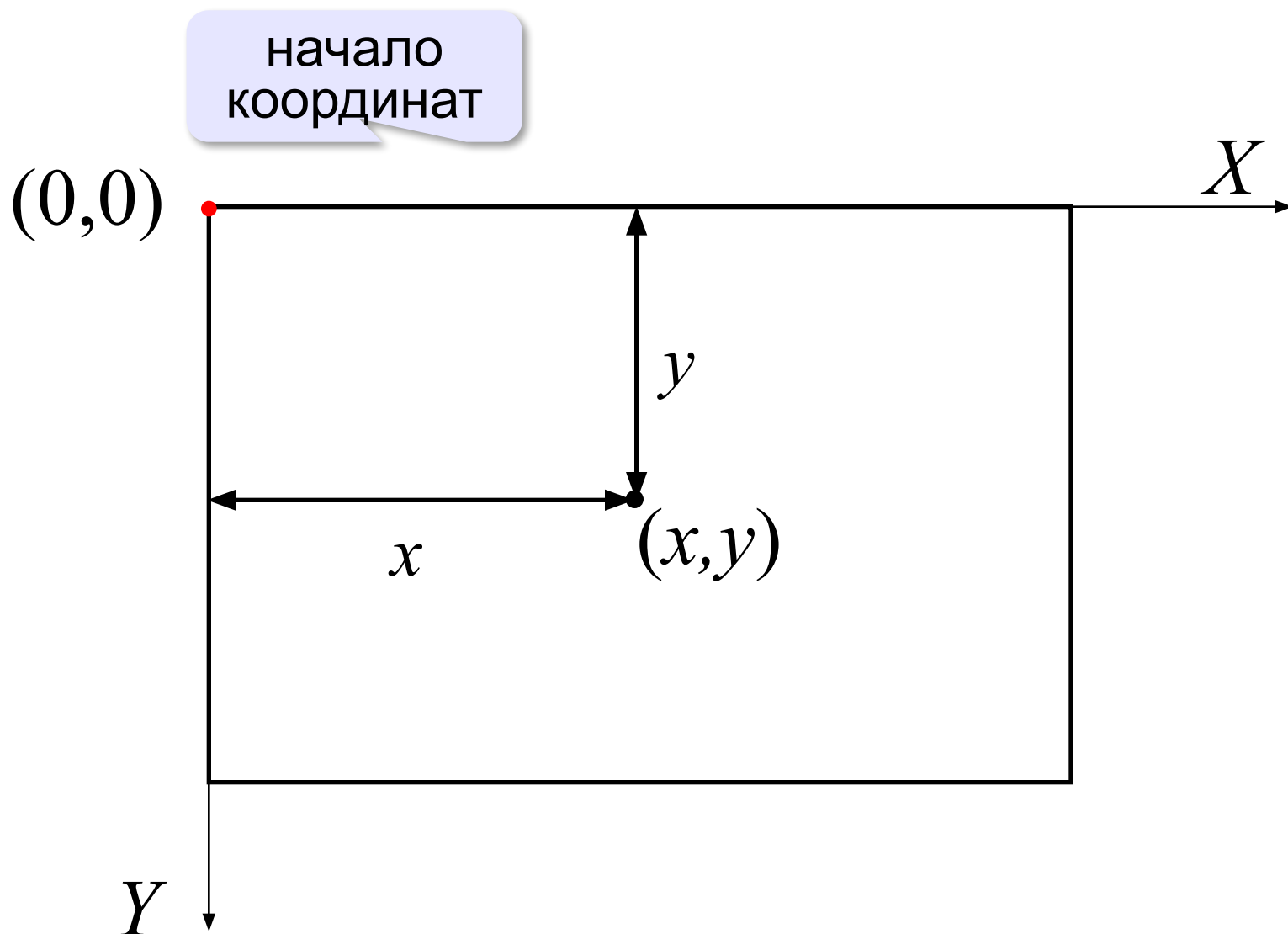
Модуль `graph.py`:

<http://kpolyakov.spb.ru/download/graph.py>

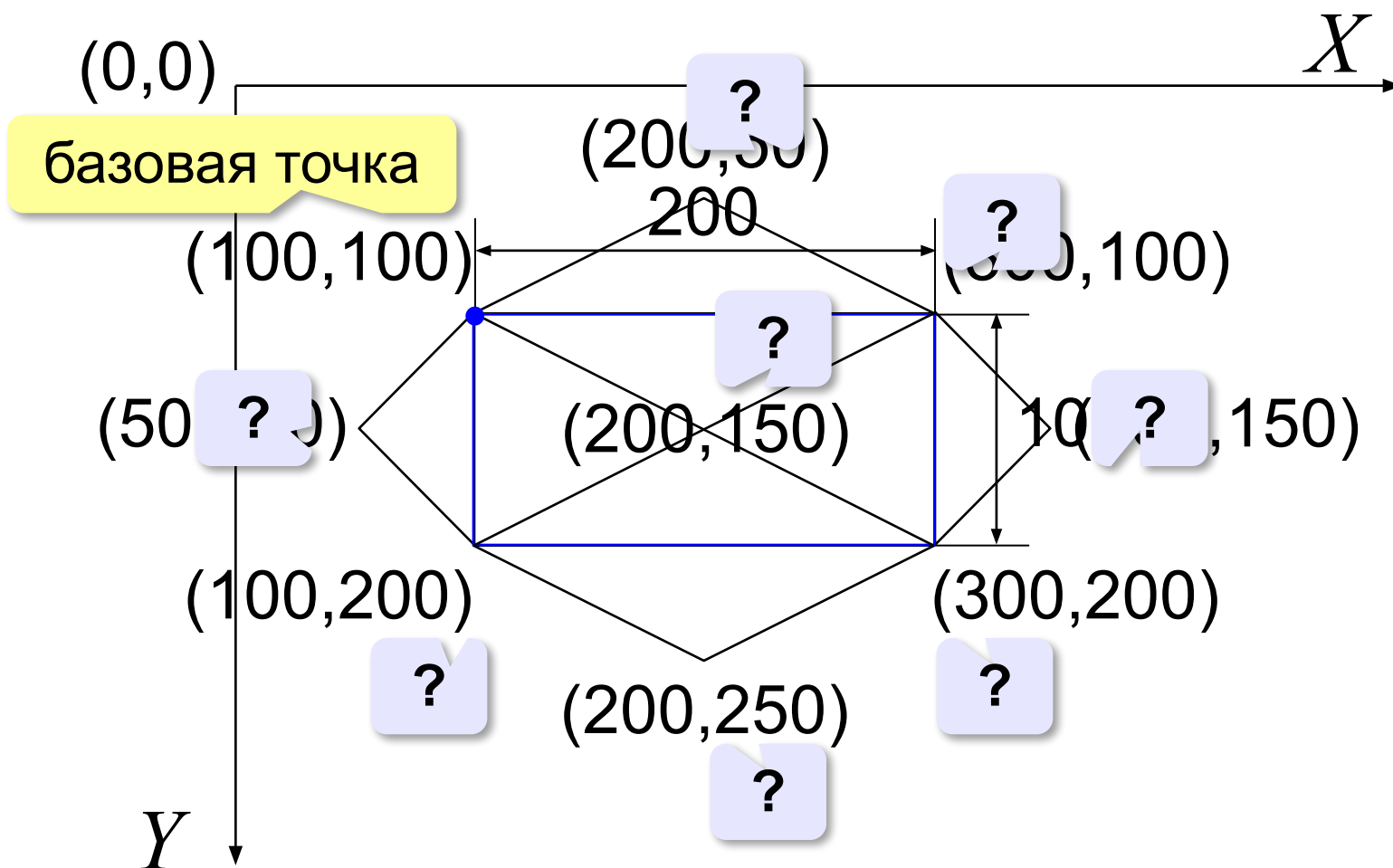
# Программирование на Python: графика

## 1. Простые программы

# Система координат



# Определение координат



# Управление цветом

---

## Подключение графического модуля:

```
from graph import *
```

подключить все функции модуля graph

## Цвет линий:

```
penColor ( "red" )
```

white, black, gray, navy, blue,  
cyan, green, yellow, red, orange,  
brown, maroon, violet, purple, ...

## Толщина линий:

<http://bit.ly/2mNrkoq>

```
penSize ( 2 )
```

## Цвет заливки:

```
brushColor ( "green" )
```

# Управление цветом (RGB)

Цвет в формате RGB:

"yellow"

```
penColor ( 255 , 255 , 0 )
```

**R**(red)  
0..255

**G**(green)  
0..255

**B**(blue)  
0..255

```
brushColor ( 255 , 0 , 255 )
```

"magenta"

```
penColor ( 0 , 255 , 255 )
```

"cyan"

```
brushColor ( 255 , 255 , 255 )
```


"white"

```
penColor ( 0 , 0 , 0 )
```

"black"

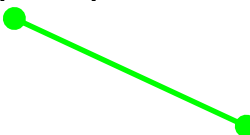
# Примитивы (простейшие фигуры)

$(x, y)$



```
penColor(0, 0, 255)  
point(x, y)
```

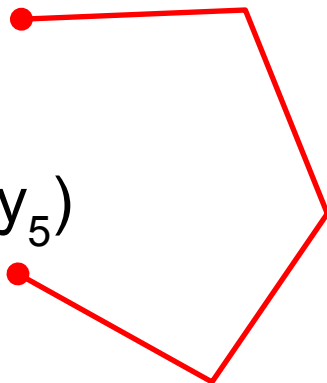
$(x_1, y_1)$



$(x_2, y_2)$

```
penColor(0, 255, 0)  
line(x1, y1, x2, y2)
```

$(x_1, y_1)$



$(x_2, y_2)$

$(x_3, y_3)$

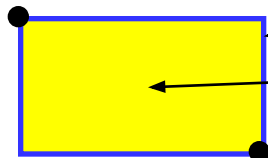
$(x_4, y_4)$

$(x_5, y_5)$

```
penColor(255, 0, 0)  
moveTo(x1, y1)  
lineTo(x2, y2)  
lineTo(x3, y3)  
lineTo(x4, y4)  
lineTo(x5, y5)
```

# Примитивы (простейшие фигуры)

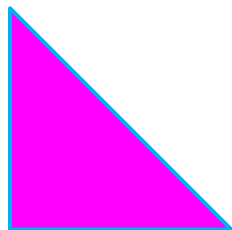
(10, 20)



(50, 40)

```
penColor("blue")  
brushColor("yellow")  
rectangle(10, 20, 50, 40)
```

(10, 10)

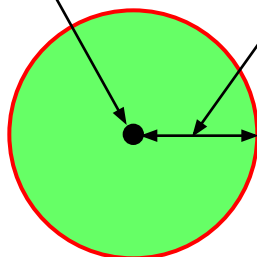


(10, 50)

(50, 50)

```
penColor("cyan")  
brushColor("magenta")  
polygon([ (10, 10), (50, 50),  
          (10, 50), (10, 10) ] )
```

(50, 30)

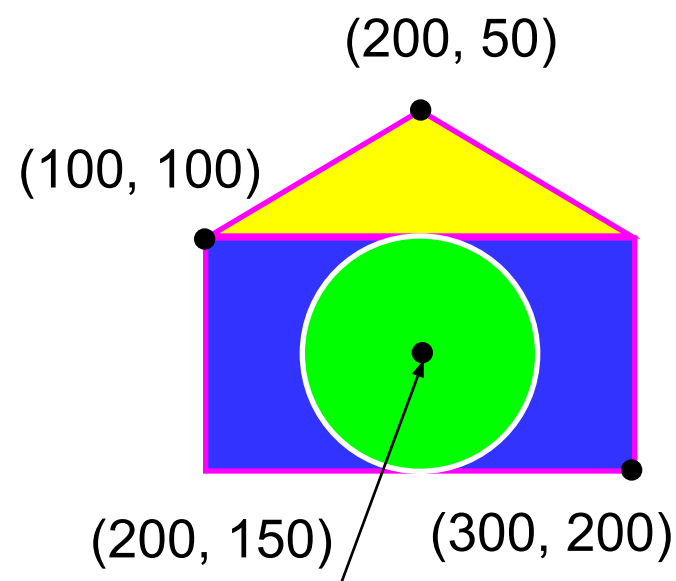


R=20

```
penColor("red")  
brushColor("green")  
circle(50, 30, 20)
```



# Пример

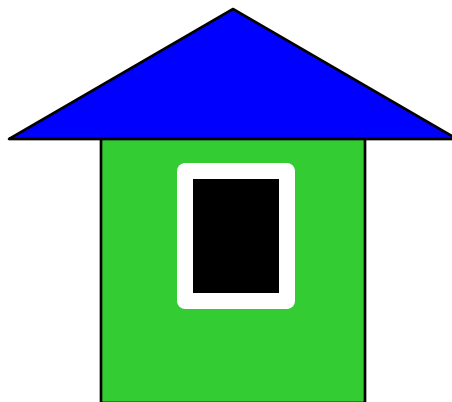


```
from graph import *  
penColor("magenta")  
brushColor("blue")  
rectangle(100, 100, 300, 200)  
brushColor("yellow")  
polygon([(100, 100), (200, 50),  
         (300, 100), (100, 100)])  
penColor("white")  
brushColor("green")  
circle(200, 150, 50)  
run()
```

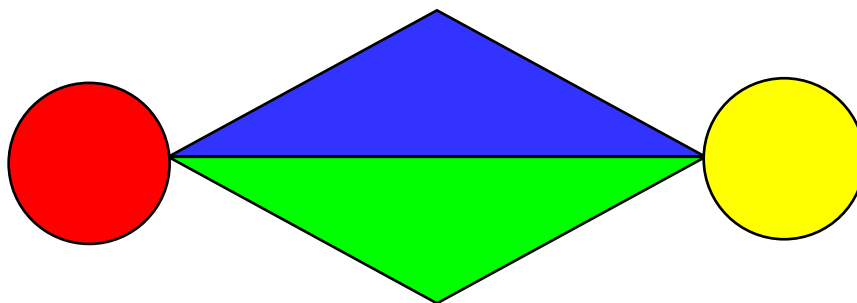
# Задачи

---

«3»: «ДОМИК»



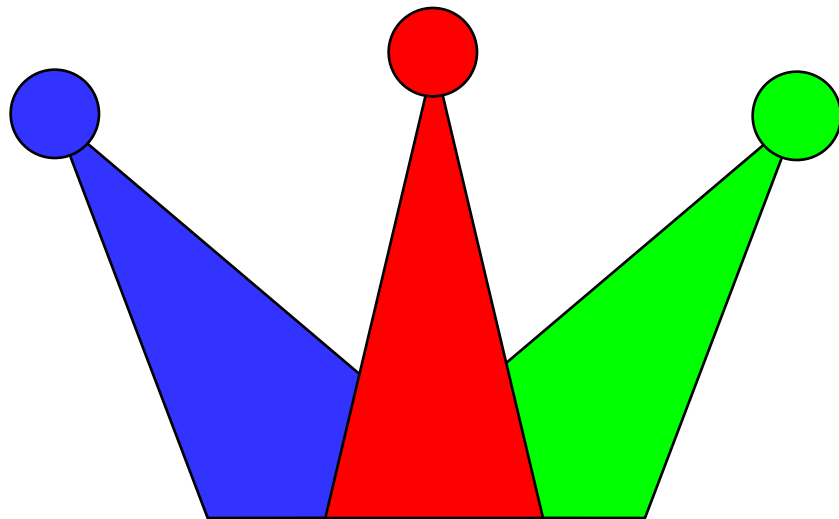
«4»: «Лягушка»



# Задачи

---

## «5»: «Корона»



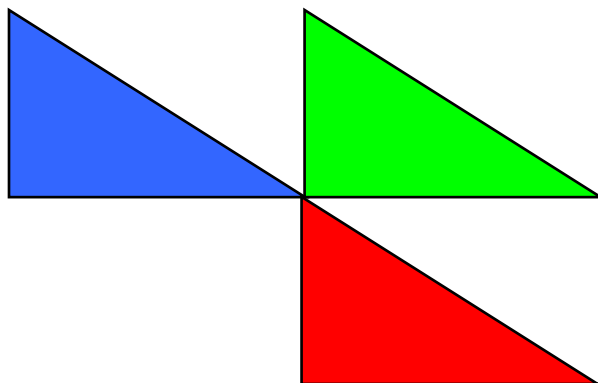
# Программирование на Python: графика

## 2. Процедуры

# Процедуры

---

**Задача:** Построить фигуру:



**?** Можно ли решить известными методами?

**Особенность:** Три похожие фигуры.

общее: размеры, угол поворота

отличия: координаты, цвет

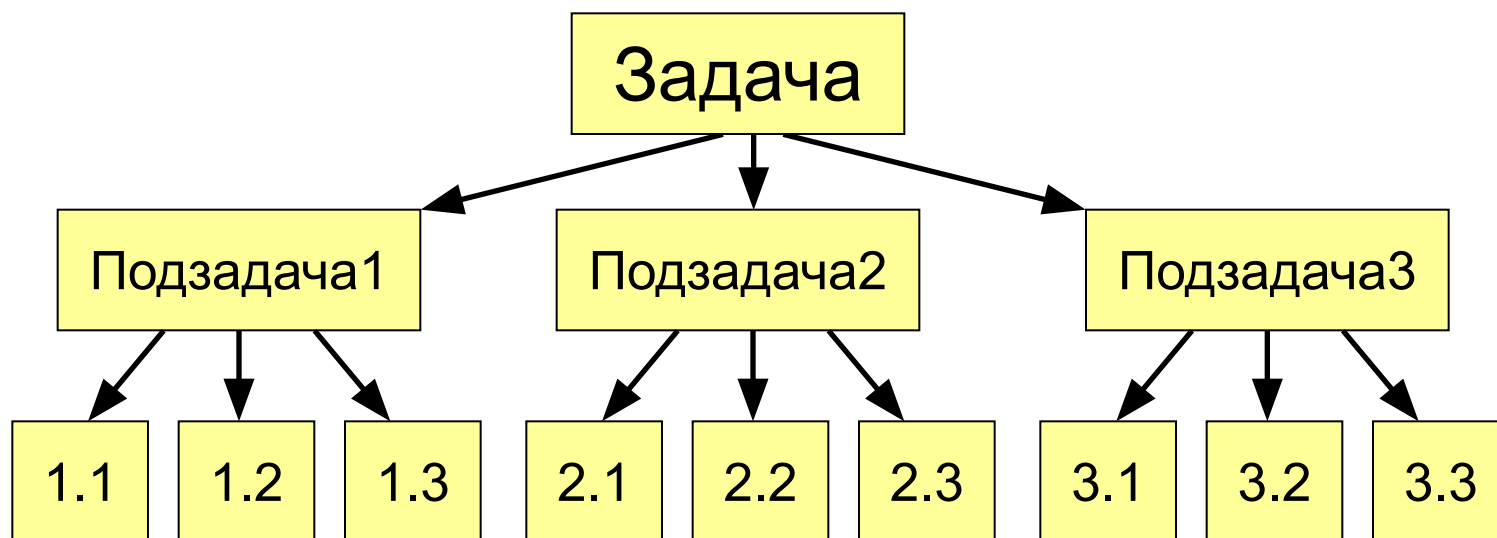
**?** Сколько координат надо задать?

# Процедуры (подпрограммы)

**Процедура** – это вспомогательный алгоритм, который предназначен для выполнения некоторых действий.

## Применение:

- выполнение одинаковых действий в разных местах программы
- разбивка программы (или другой процедуры) на подзадачи для лучшего восприятия

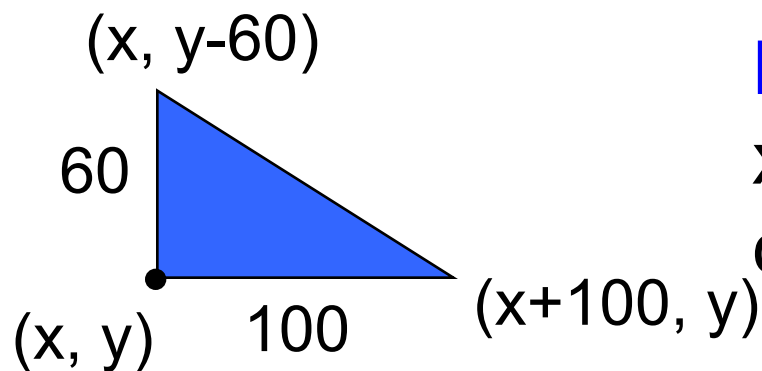


# Как построить процедуру?

- выделить одинаковые или похожие действия (*три фигуры*)
- найти в них общее (*размеры, форма, угол поворота*) и отличия (*координаты, цвет*)
- отличия обозначить как **переменные**, они будут **параметрами** процедуры



**Параметры** – это данные, от которых зависит работа процедуры.

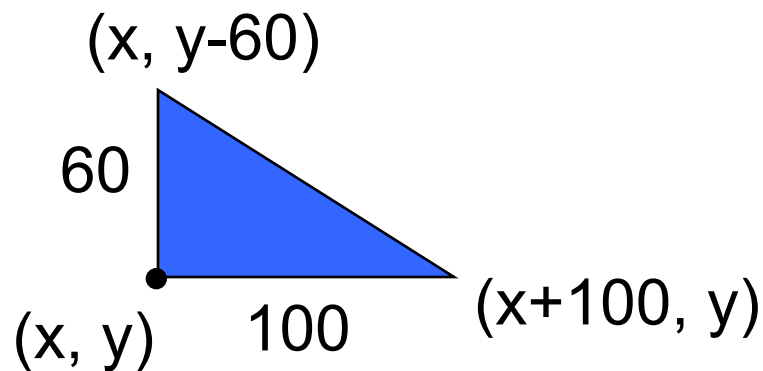


**Параметры:**

$x, y$  – координаты угла

$c$  – цвет заливки

# Процедура



определить  
(*define*)

название

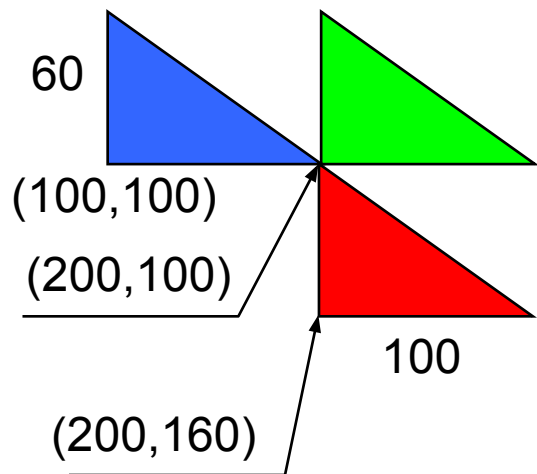
параметры

отступ

```
def treug(x, y, c):  
    brushColor(c)  
    polygon([ (x, y), (x, y-60),  
              (x+100, y), (x, y) ] )
```



# Программа с процедурой



ВЫЗОВЫ  
процедуры

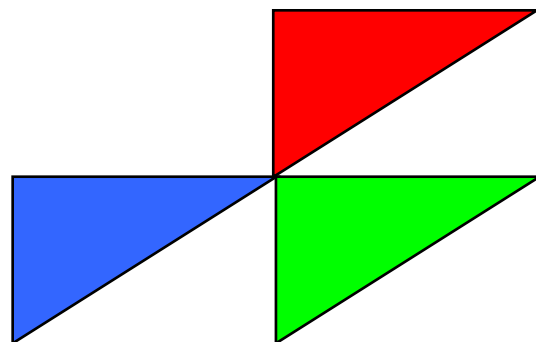
```
from graph import *  
def treug(x, y, c):  
    brushColor(c)  
    polygon([(x,y), (x,y-60),  
            (x+100,y), (x,y)] )  
    penColor("black")  
treug(100, 100, "blue")  
treug(200, 100, "green")  
treug(200, 160, "red")  
run()
```

аргументы (значения  
параметров)

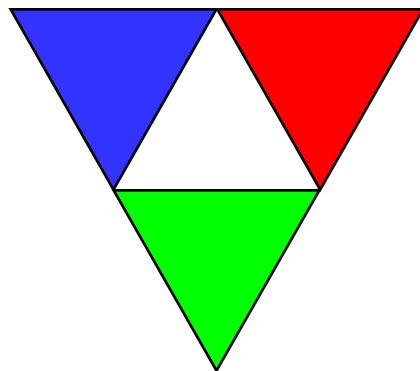
# Задания

---

«3»: Используя одну процедуру, построить фигуру.



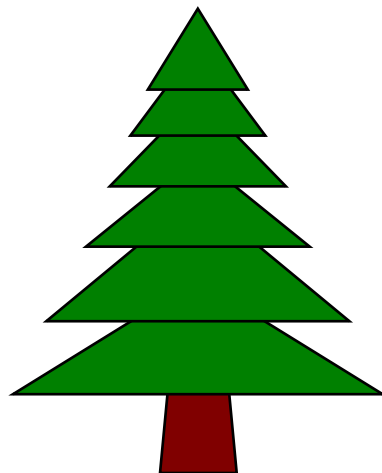
«4»: Используя одну процедуру, построить фигуру.



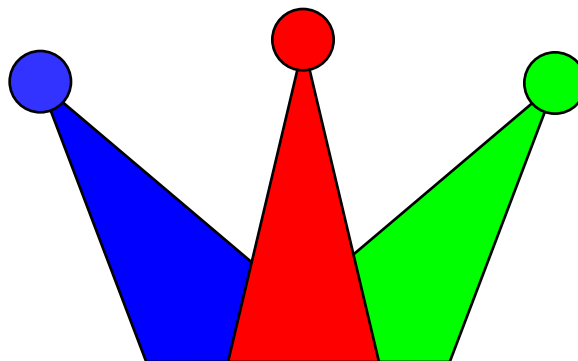
# Задания

---

«5»: Используя одну процедуру, построить фигуру.



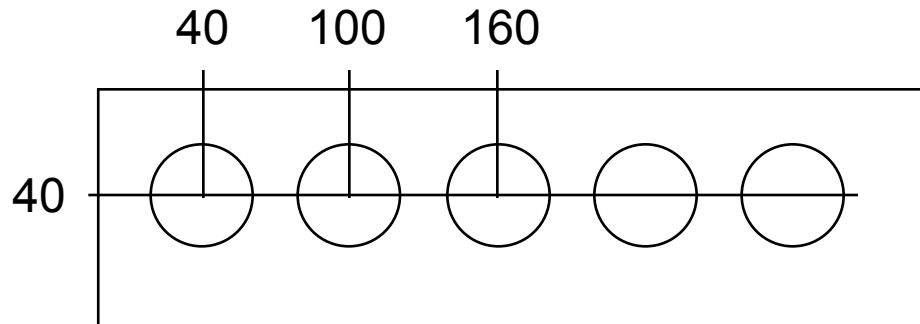
«6»: Используя одну процедуру, построить фигуру.



# Программирование на Python: графика

## 3. Циклы

# Использование циклов



**?** Что меняется?

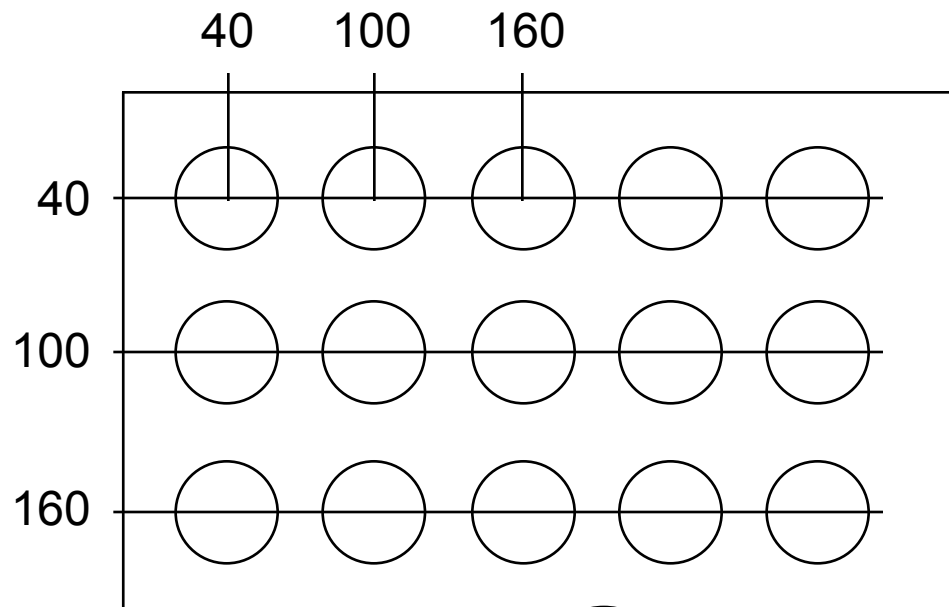
```
circle ( 40, 40, 20 )
circle ( 100, 40, 20 )
circle ( 160, 40, 20 )
...
```

**?** Как меняется x?

```
x = 40
for i in range(5):
    circle(x, 40, 20)
    x += 60
```

"сделай 5 раз"

# Использование циклов



1-й ряд:



Что меняется для 2-го ряда?

```
x = 40
for i in range(5):
    circle(x, 40, 20)
x += 60
```



Можно сделать это процедурой с параметром  $y$ !

# Использование циклов

```
from graph import *
```

```
def row ( y ):
```

```
    x = 40
```

```
    for i in range(5):
```

```
        circle(x, y, 20)
```

```
    x += 60
```

```
y = 40
```

```
for k in range(3):
```

```
    row ( y )
```

```
    y += 60
```

```
run ()
```

процедура

ВЫЗОВ  
процедуры

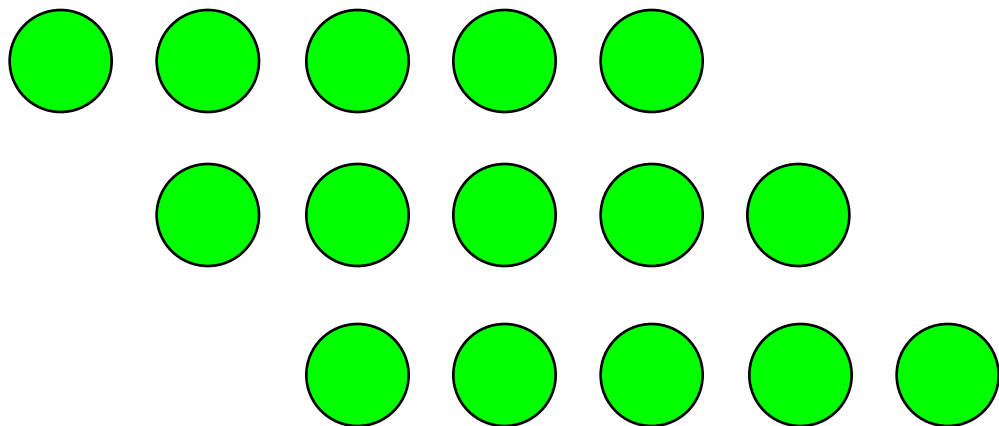
вниз на 60

# Задания

---

«3»: Ввести с клавиатуры число  $N$  и нарисовать  $N$  рядов по 5 кругов.

Пример ( $N = 3$ ):



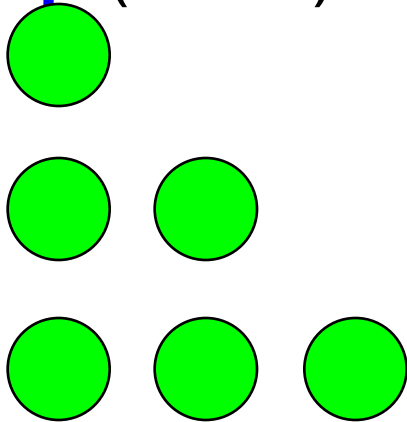


# Задания

---

«4»: Ввести с клавиатуры число  $N$  и нарисовать из кругов прямоугольный треугольник размером  $N$  на  $N$ .

**Пример** ( $N = 3$ ):

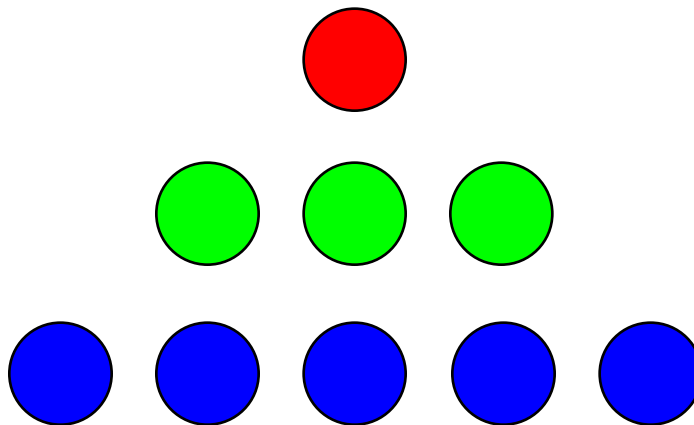


## Задания

---

«5»: Ввести с клавиатуры число  $N$  и нарисовать из кругов равнобедренный треугольник с высотой  $N$ . Каждый ряд должен быть покрашен в свой цвет.

Пример ( $N = 3$ ):

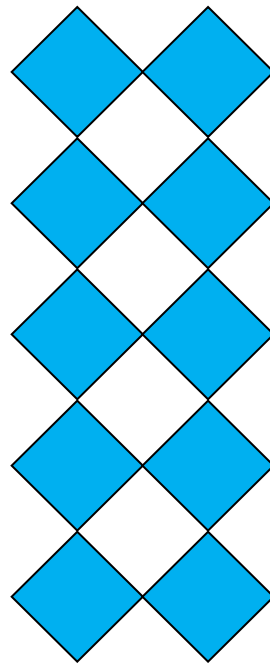


## Задания-2

---

«3»: Ввести с клавиатуры число  $N$  и нарисовать  $N$  вертикальных рядов по 5 ромбиков.

Пример ( $N = 2$ ):

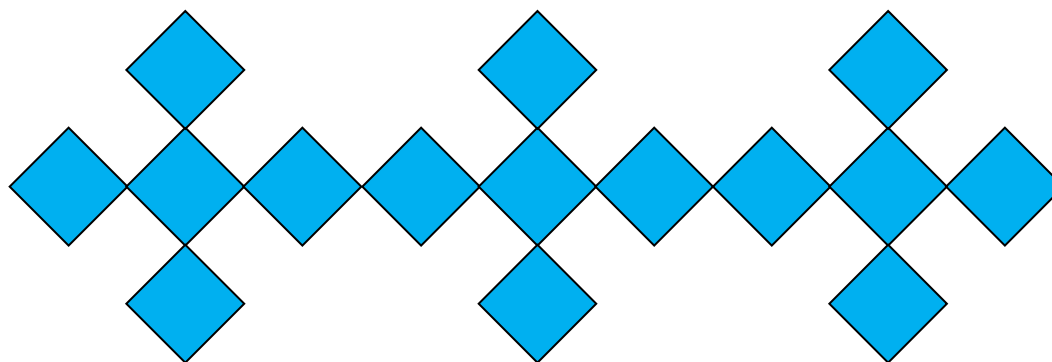


## Задания-2

---

«4»: Используя циклы и процедуры, нарисуйте узор. Число повторений рисунка  $N$  введите с клавиатуры.

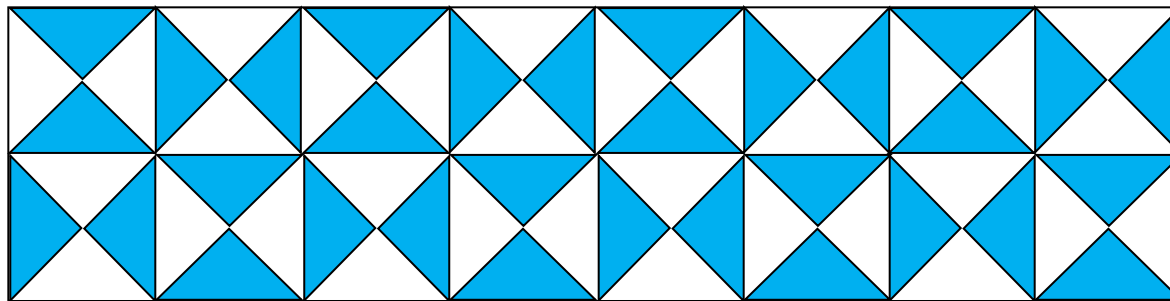
**Пример** ( $N = 3$ ):



## Задания-2

---

«5»: Используя циклы и процедуры, нарисуйте узор.

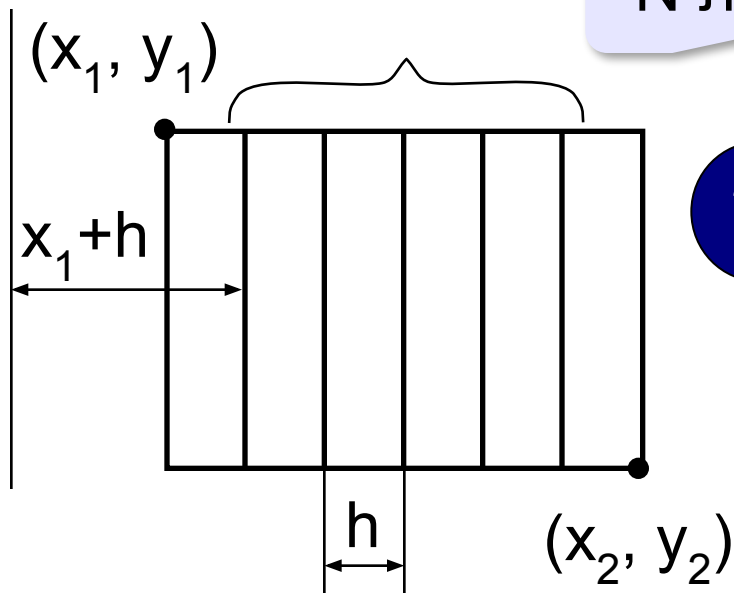


# Программирование на Python: графика

## 4. Штриховка

# Штриховка

N линий (N=5)



Как найти  $h$ ?

$$h = \frac{x_2 - x_1}{N + 1}$$

В цикле менять  $x$ :

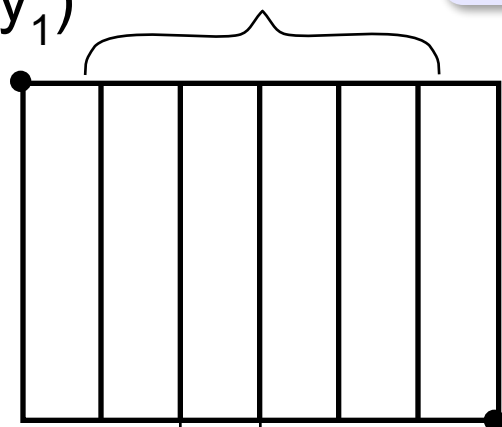
```
line ( x, y1, x, y2)
```

```
rectangle (x1, y1, x2, y2)
line ( x1+h, y1, x1+h, y2)
line ( x1+2*h, y1, x1+2*h, y2)
line ( x1+3*h, y1, x1+3*h, y2)
...
      x           x
```

# Штриховка

N линий (N=5)

$(x_1, y_1)$



меняется!

`line(x, y1, x, y2)`

?

Как меняется?

$x = ?$

$h$

$(x_2, y_2)$

для 1-й линии

```
x = x1 + h
```

```
for i in range(N):
```

```
    line(x, y1, x, y2)
```

```
    x += h
```

"сделай N раз"

для следующей  
линии

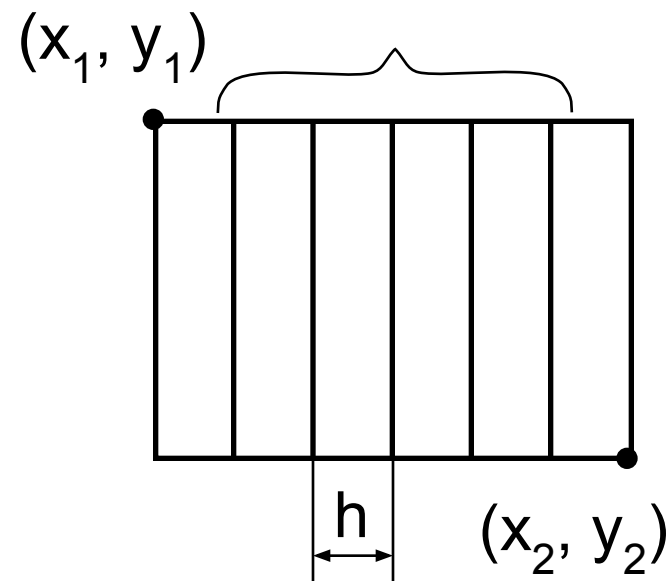
?

Что плохо?



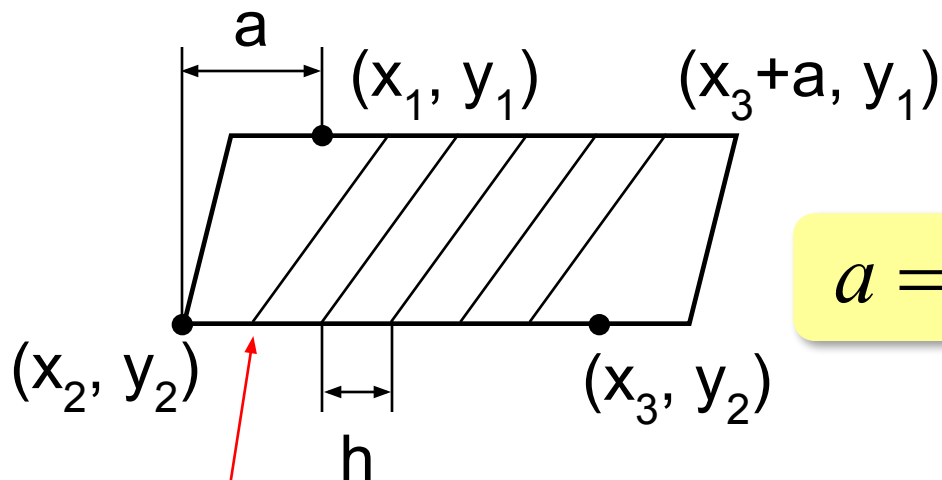
# Штриховка

N линий



```
from graph import *  
x1 = 100; y1 = 100  
x2 = 300; y2 = 200  
N = 10  
rectangle(x1, y1, x2, y2)  
h = (x2-x1) / (N+1)  
x = x1 + h  
for i in range(N):  
    line(x, y1, x, y2)  
    x += h  
run()
```

# Сложная штриховка



Как найти  $a$  и  $h$ ?

$$a = x_1 - x_2$$

$$h = \frac{x_3 - x_2}{N + 1}$$

```
line ( x1+h,      y1,  x1+h-a,      y2 );
line ( x1+2*h,   y1,  x1+2*h-a,    y2 );
line ( x1+3*h,   y1,  x1+3*h-a,    y2 );
...
```

$x$

$x - a$



Как меняется  $x$ ?

Сначала:

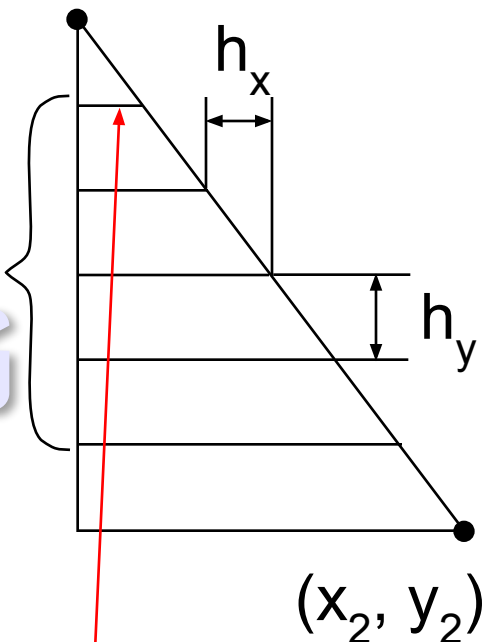
$$x = x1 + h$$

В цикле:

$$x += h$$

# Очень сложная штриховка

$(x_1, y_1)$



Как найти  $h_x$  и  $h_y$ ?

$$h_x = \frac{x_2 - x_1}{N + 1}$$

$$h_y = \frac{y_2 - y_1}{N + 1}$$

**Сначала:**

`x = x1+hx`

`y = y1+hy`

**В цикле:**

`x += hx`

`y += hy`

```
line( x1, y1+hy, x1+hx, y1+hy) ;
line( x1, y1+2*hy, x1+2*hx, y1+2*hy) ;
line( x1, y1+3*hy, x1+3*hx, y1+3*hy) ;
...
```

y

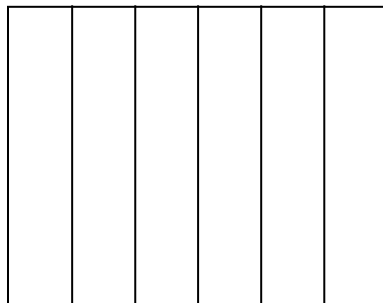
x

y

## Задания

---

«3»: Ввести с клавиатуры количество линий,  
построить фигуру и выполнить штриховку:



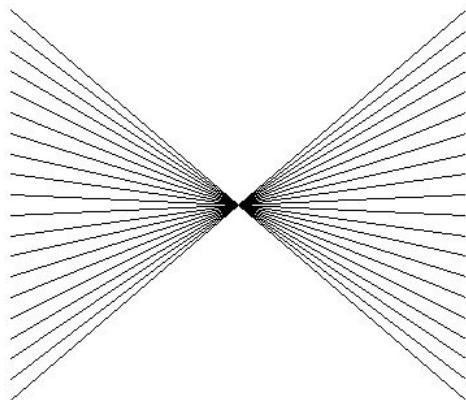
«4»: Ввести с клавиатуры количество линий,  
построить фигуру и выполнить штриховку:



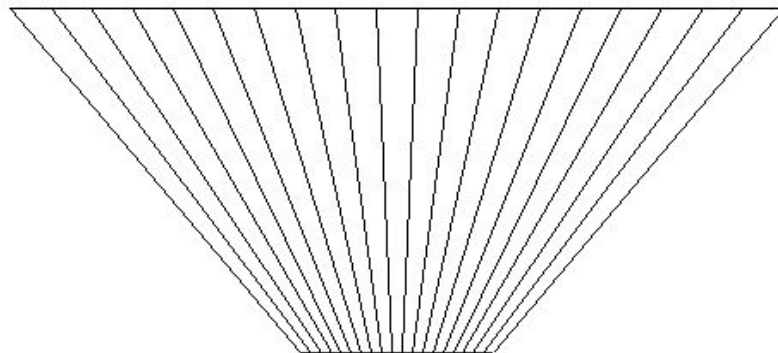
## Задания

---

«5»: Ввести с клавиатуры количество линий и построить фигуру:



«6»: Ввести с клавиатуры количество линий и построить фигуру:

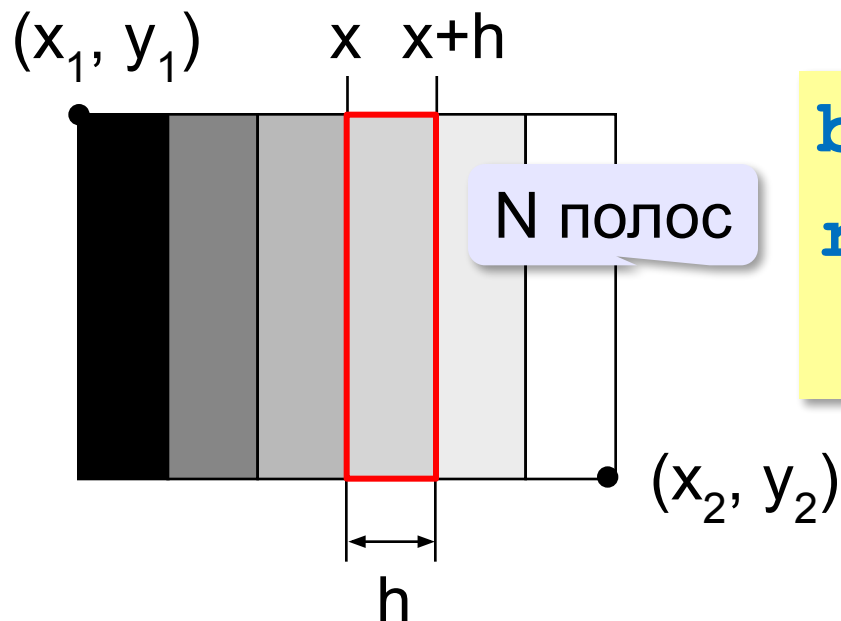


# Программирование на Python: графика

## 5. Закрашивание областей

# Заливка разными цветами

серый: R=G=B



```
brushColor(c, c, c)
rectangle(x, y1,
          x+h, y2)
```

Шаг изменения цвета:

$$hc = 255 // N$$

```
x = x1; c = 0
for i in range(N):
    brushColor(c, c, c)
    rectangle(x, y1, x+h, y2)
    x += h; c += hc
```

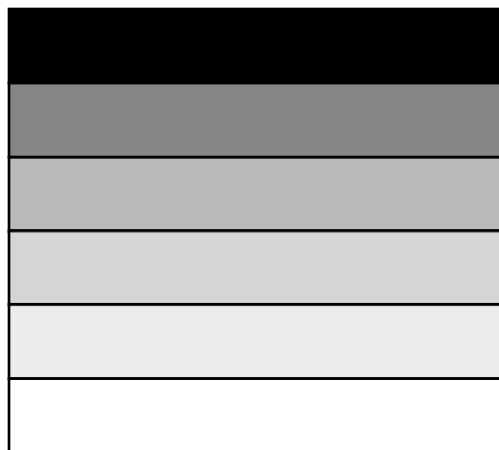
# Задания

---

«3»: Ввести с клавиатуры число полос и построить фигуру, залив все области разным цветом.



«4»: Ввести с клавиатуры число полос и построить фигуру, залив все области разным цветом.





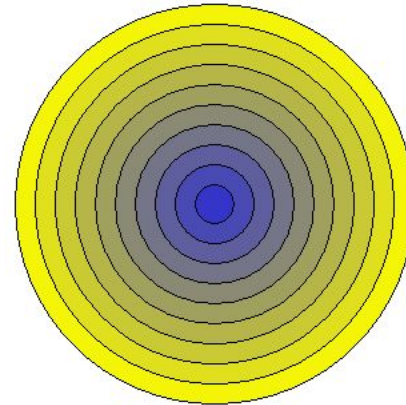
# Задания

---

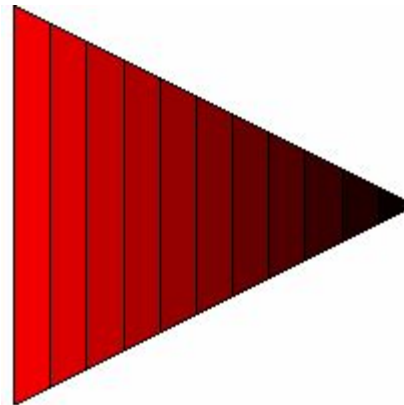
«5»: Ввести с клавиатуры число полос и построить фигуру, залив все области разным цветом.



или



«6»: Ввести с клавиатуры число полос и построить фигуру, залив все области разным цветом.



# Программирование на Python: графика

## 6. Построение графиков функций

# Графики функций

---

**Задача:** построить график функции  $y = x^2$  на отрезке от -2 до 2.

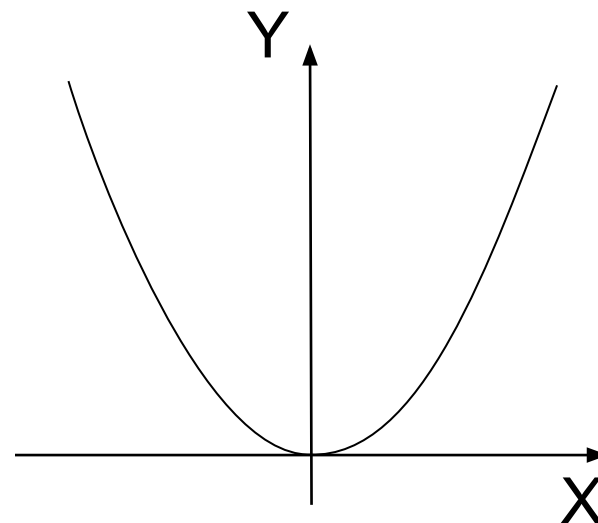
**Анализ:**

максимальное значение

$$y_{\max} = 4 \quad \text{при} \quad x = \pm 2$$

минимальное значение

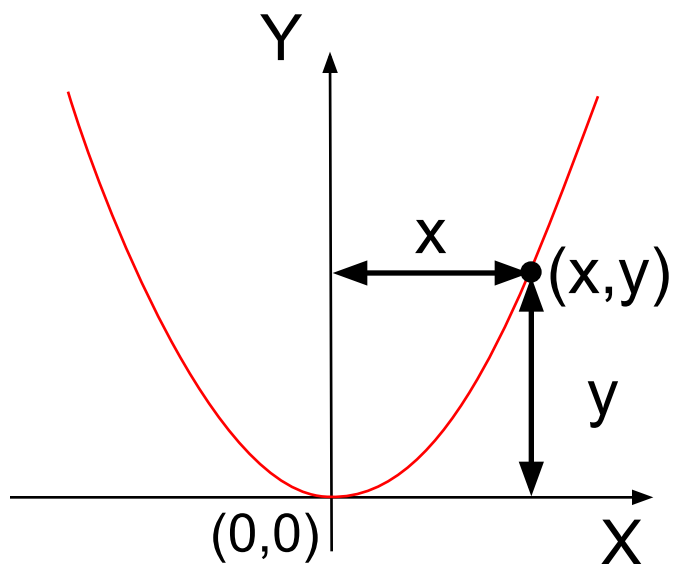
$$y_{\min} = 0 \quad \text{при} \quad x = 0$$



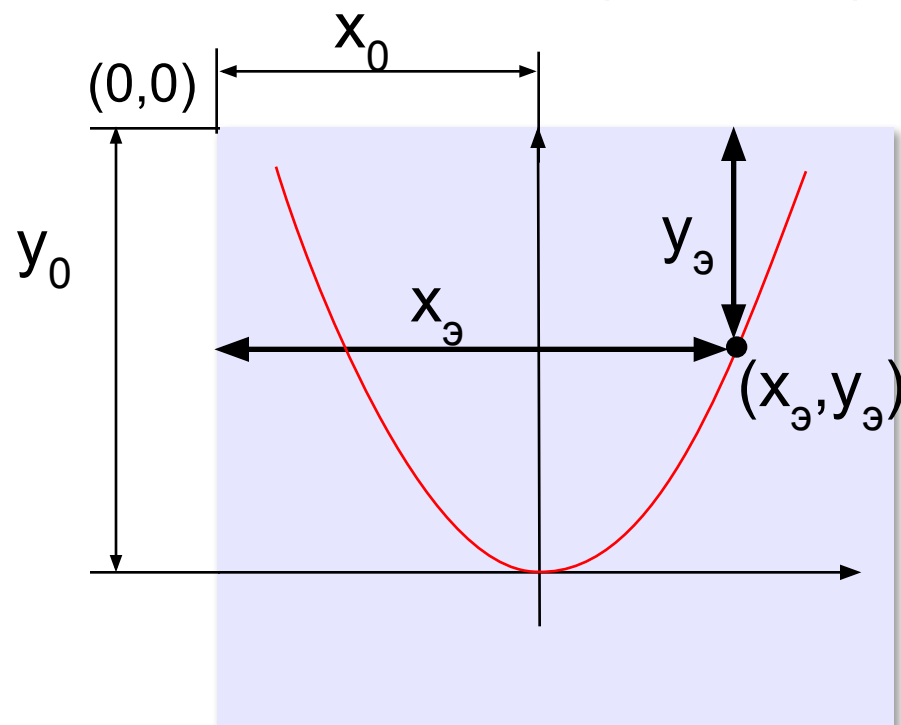
**Проблема:** функция задана в математической системе координат, строить надо на экране, указывая координаты в пикселях.

# Преобразование координат

Математическая  
система координат



Экранная система  
координат (пиксели)

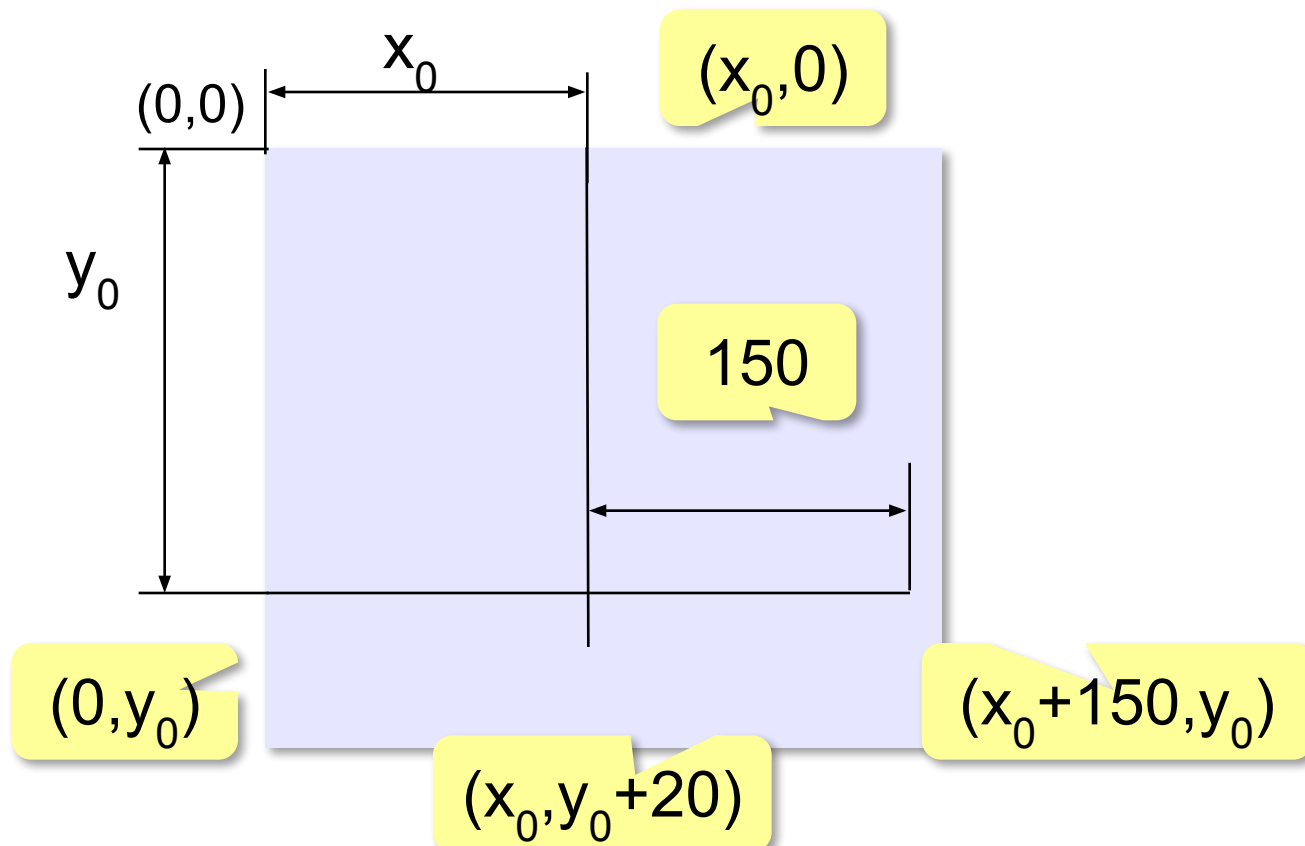


$k$  – масштаб (длина  
изображения единичного  
отрезка на экране)

$$x_{\text{э}} =$$

$$y_{\text{э}} =$$

# Оси координат



```
line (0, y0, x0+150, y0)
```

```
line (x0, 0, x0, y0+20)
```

# Рисуем оси координат

```
from graph import *
x0 = 150 # начало координат
y0 = 250
k = 50    # масштаб
xmin = -2; xmax = 2 # пределы по x
line(0, y0, x0+150, y0)
line(x0, 0, x0, y0+20)
...
```

# Строим по точкам

```
...  
x = xmin    # начальное значение x  
h = 0.02    # шаг изменения x  
penColor("red")  
while x <= xmax:  
    y = x*x  # функция  
    xe = x0 + k*x  
    ye = y0 - k*y  
    point(xe, ye) # точка на экране  
    x += h      # к следующей точке  
run()
```

экранные координаты  
(в пикселях)

# Соединяем точки линиями

**Идея:** сначала создаём в памяти массив точек, затем соединяем точки линиями (**polygon**)

```
points = [] # пустой массив
```

```
while x <= xmax:
```

```
    y = x*x
```

```
    xe = x0 + k*x
```

```
    ye = y0 - k*y
```

```
    points.append( (xe, ye) )
```

```
    x += h
```

добавляем точку  
в массив

```
penColor("red")
```

```
polyline(points) # рисуем линию!
```



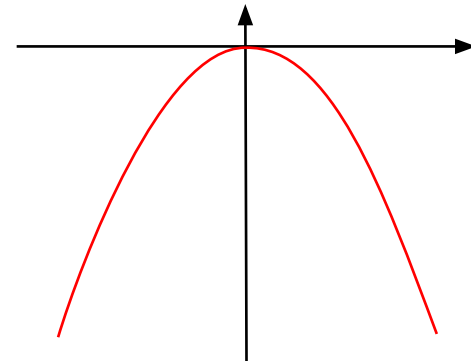
# Задания

---

«3»: Построить график функции

$$y = -x^2$$

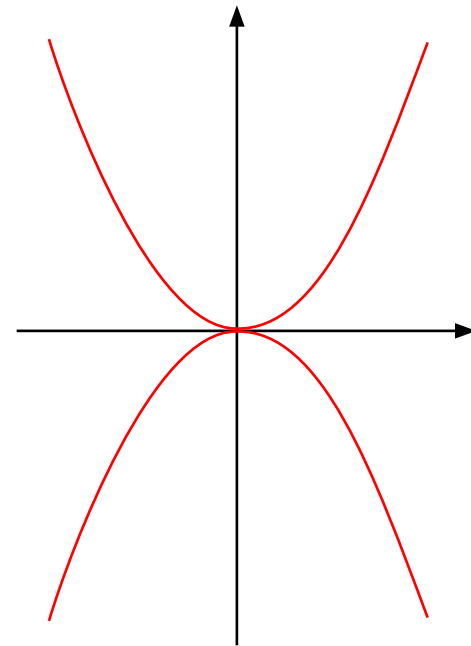
на отрезке  $[-2,2]$ .



«4»: Построить графики функций

$$y = x^2 \text{ и } y = -x^2$$

на отрезке  $[-2,2]$ .



# Задания

---

«5»: Построить графики функций

$$x = y^2 \text{ и } x = -y^2$$

на отрезке  $[-2, 2]$ .

