



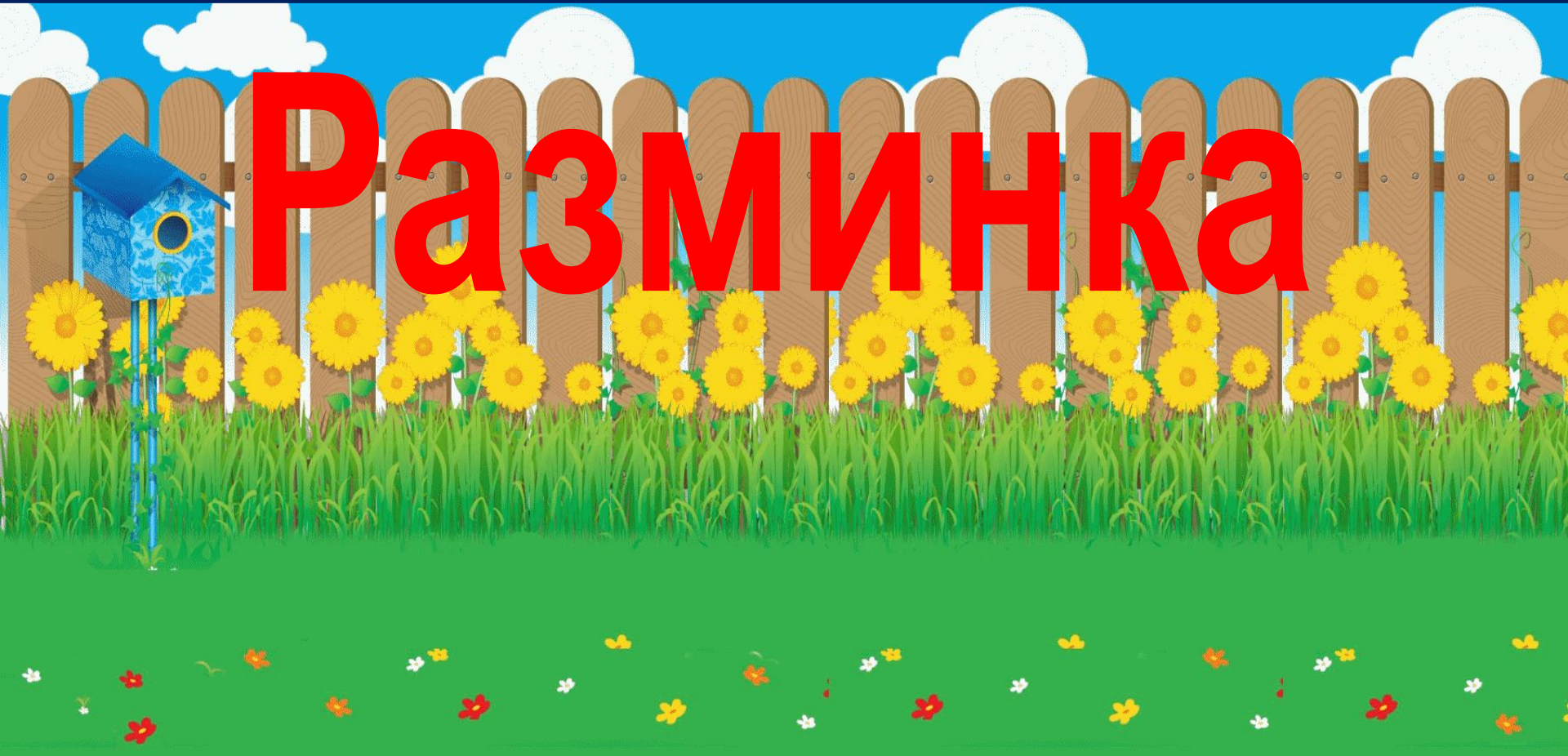
# ИНФОРМАТИКА





# ИНФОРМАТИКА

Разминка

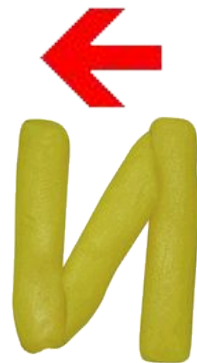




# ИНФОРМАТИКА



”””





# ИНФОРМАТИКА



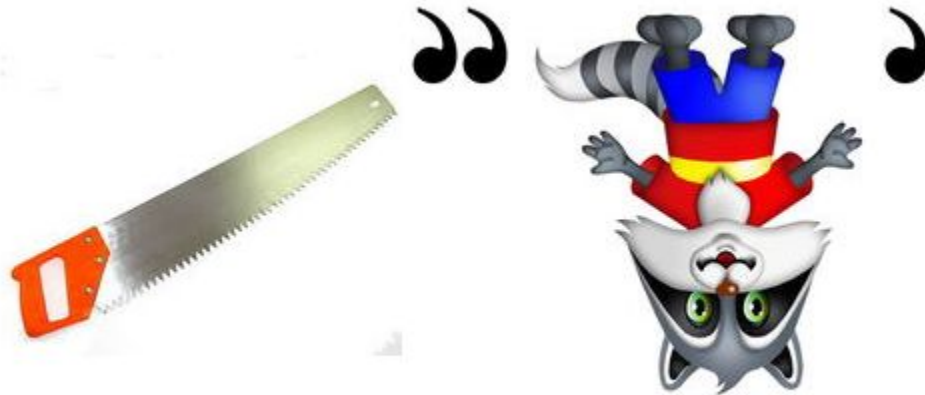
”””



Г Р А Ф И К А

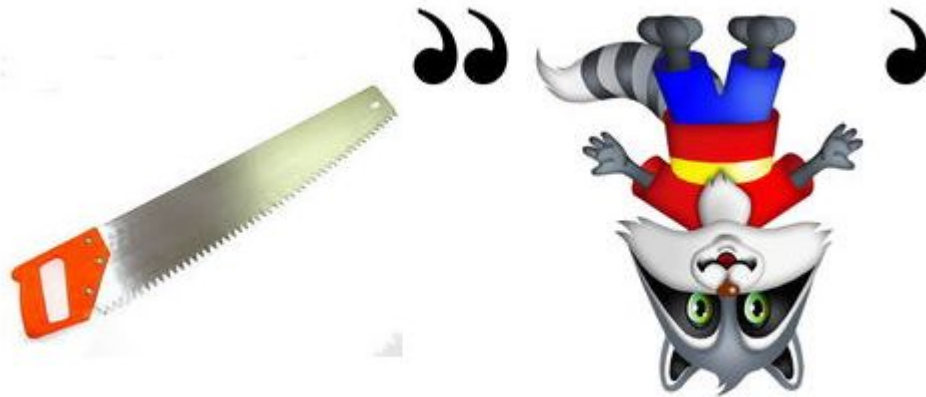


# ИНФОРМАТИКА





# ИНФОРМАТИКА



ПИТОН



# ИНФОРМАТИКА



,



”



☞ 2 = A





# ИНФОРМАТИКА



2 = A

Ч Е Р Е П А Ш К А





# ИНФОРМАТИКА

Г Р А Ф И К А

П И Т О Н

Ч Е Р Е П А Ш К А

# Программирование

Глава

3

Учебник  
«ИНФОРМАТИКА 7-9 КЛАСС»

И. Н. Цыбуля, Л. А. Самыкбаева,  
А. А. Беляев, Н. Н. Осипова, У. Э. Мамбетакунов

## Урок №22

**3.9.Тема:  
«Работа с графикой в  
Python. Знакомство с  
модулем Turtle  
(черепашка)»**



**1**

**Графические режимы в Python**

**2**

**Рисование с помощью модуля Turtle**



**3**

**Управление модулем Turtle (черепашка) для создания графических примитивов и перемещения на плоскости.**

**4**

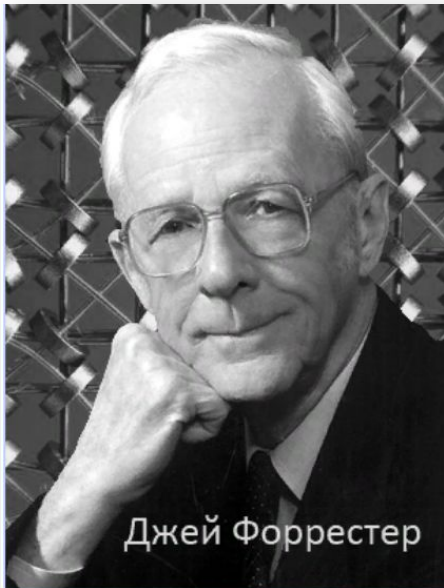
**Команды управления графическим пером.**



## Компьютерная графика



ЭТО  
ИНТЕРЕСНО



Джей Форрестер

*Первые эксперименты по созданию изображений проведены в компьютерной лаборатории Массачусетского технологического института в 1951 году.*

**Компьютерная графика** — раздел информатики, предметом изучения которого является создание и обработка на компьютере графических изображений.



## Графика в Python.

С помощью *графики в Python* можно рисовать фигуры и изображения, создавать анимацию, визуализировать математические вычисления в Python.

В программах *python* можно использовать элементы графики в компьютерных играх.

```
import turtle
import random

window = turtle.Screen()
arthur = turtle.Turtle()
window.colormode(255)
arthur.speed(0)
arthur.width(2)
window.bgcolor(50,0,70)
arthur.pencolor(255,255,0)

def shape(angle,side,limit):
    reverseDirection = 200
    arthur.forward(side)

    if side % (reverseDirection*2) == 0:
        angle = angle + 2
        print side
    elif side % reverseDirection == 0:
        angle = angle - 2
        print side

    arthur.right(angle)
    side = side + 2
    if side < limit:
        shape(angle,side,limit)

angle = 119
side = 0
limit = 600
shape(angle, side, limit)

turtle.done()
```



## Графический модуль



**Модуль** — это ряд связанных между собой операций. **Модуль в Python** — это файл, содержащий код языка программирования python, который вы хотите включить в проект.

**Модули** — это, встроенные в язык программирования функции, которые доступны сразу. Чтобы их вызвать, не надо выполнять никаких дополнительных действий.



# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

### *Графический модуль*



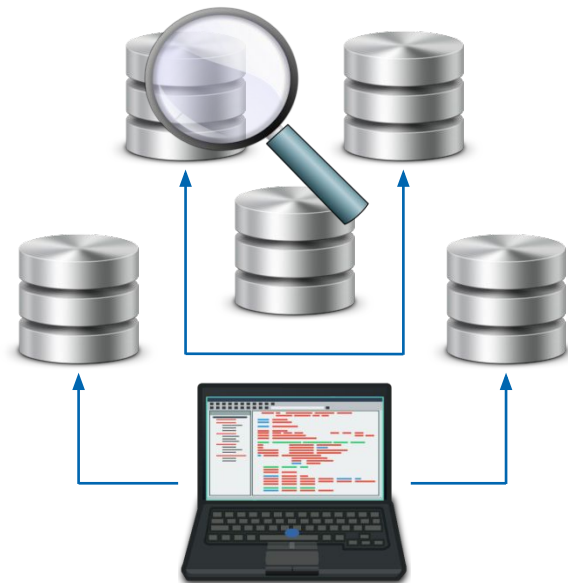




# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

### *Графический модуль*





## Графический модуль

в модуле **math** языка Python содержатся математические функции

модуль **random** позволяет генерировать псевдослучайные числа

модуль **sys** предоставляет доступ к системным переменным

Для доступа к функционалу модуля, его надо импортировать в программу. После импорта интерпретатор "знает" о существовании дополнительных классов и функций и позволяет ими пользоваться.

В Python импорт осуществляется командой **import**.



## Модуль `graphics.py`

*`graphics.py`*



Поисковые проекты



[Справка](#)

[Спонсор](#)

[Вход](#)

[Регистрация](#)

# `graphics.py` 5.0.1.пост1



[Новейшая версия](#)

```
pip install graphics.py
```



Дата Выхода: 30 Ноября 2016 Года



## Модуль *Turtle* (черепашка)

Исполнитель

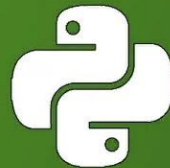


```
import turtle
```



```
# подключаем модуль turtle
```

PYTHON  
TURTLE  
GRAPHICS





## Модуль *Turtle* (черепашка)



`turtle.reset()`



*# приводим черепашку в начальное положение*

```
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2
Type "help", "copyright", "credits" or "lic
>>> import turtle
# Подключаем модуль turtle
>>> turtle.reset()
# Приводим черепашку в начальное положение
>>>
```



## Модуль *Turtle* (черепашка)



```
turtle.shape("стиль") # меняем внешний вид
```



```
turtle.shape('square')
```



```
turtle.shape("turtle")
```



```
turtle.shape('circle')
```





## Модуль *Turtle* (черепашка)



```
turtle.shape("стиль") # меняем внешний вид
```



```
turtle.shape("arrow")
```



```
turtle.shape('triangle')
```



```
turtle.shape('classic')
```





## Модуль *Turtle* (черепашка)

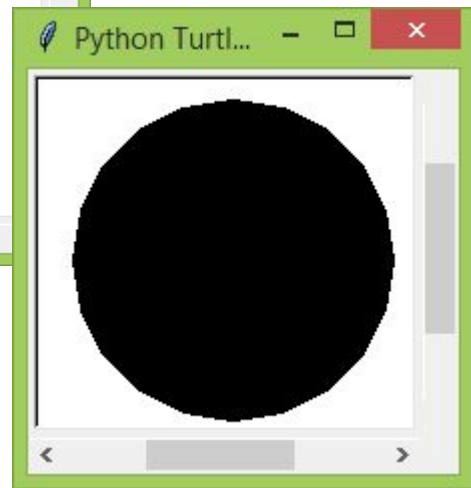
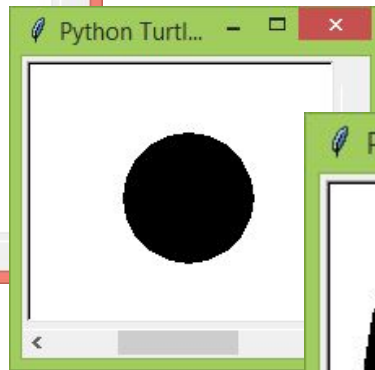
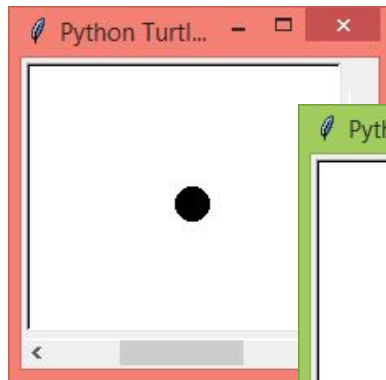


```
turtle.shapesize(размер) # устанавливаем размер
```

```
import turtle  
turtle.shape('circle')
```

```
turtle.shapesize(4)
```

```
turtle.shapesize(8)
```







## Модуль *Turtle* (черепашка)



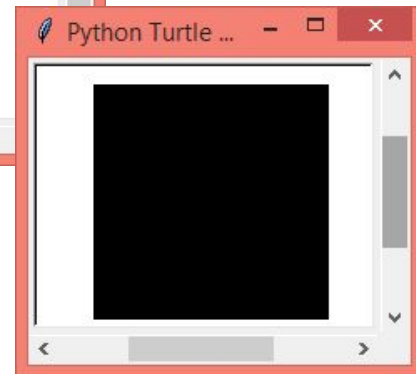
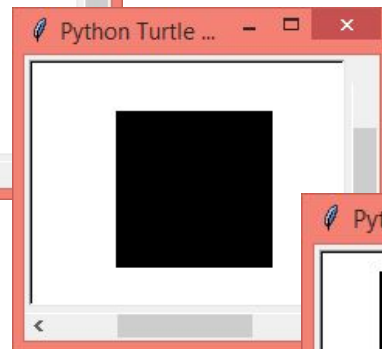
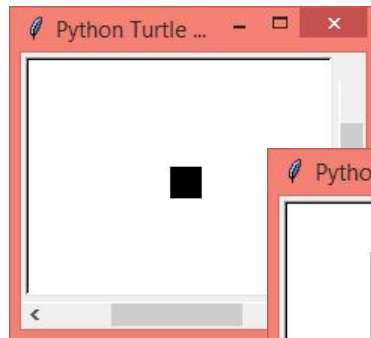
```
turtle.shapesize(размер) # устанавливаем размер
```

```
import turtle  
turtle.shape('square')
```

```
turtle.shapesize(5)
```

```
turtle.shapesize(7)
```

*Изменение размера пропорционально*



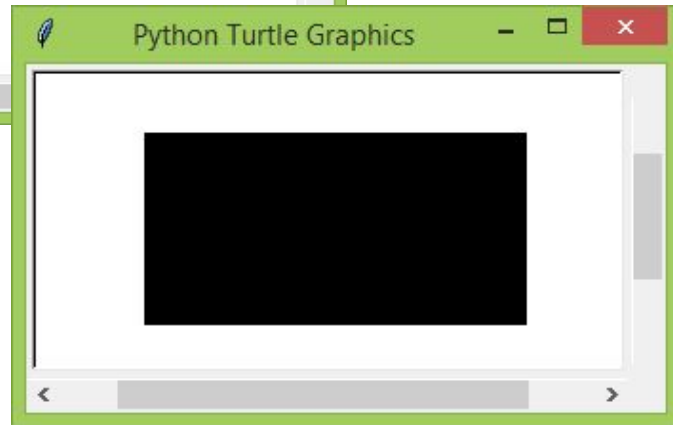
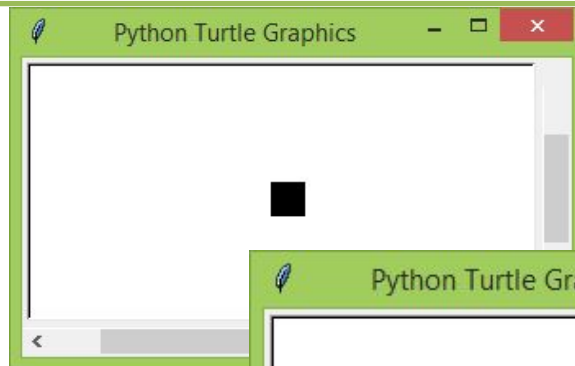


## Модуль *Turtle* (черепашка)

*turtle.shapesize*(**высота, ширина, контур**)  
# устанавливаем размер

```
import turtle  
turtle.shape('square')
```

```
turtle.shapesize(5,10)
```

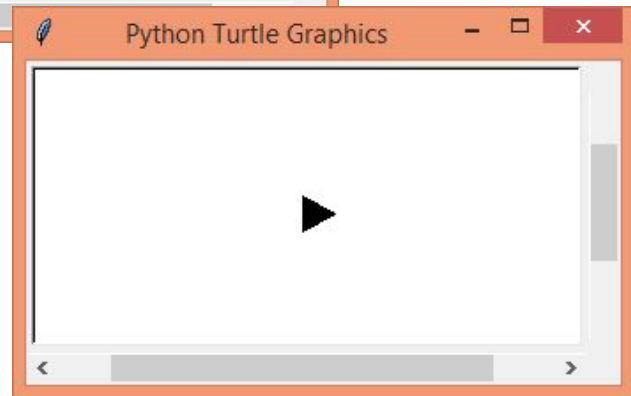
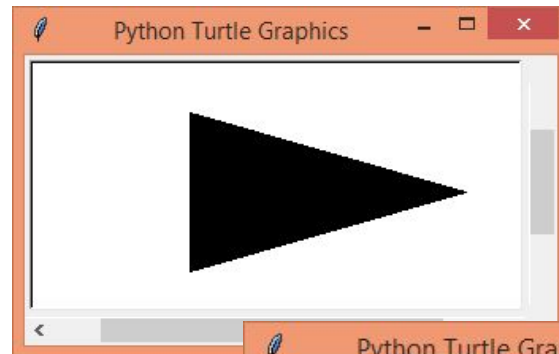




## Меняем внешний вид:

```
import turtle  
turtle.shape("arrow")
```

```
turtle.reset()
```



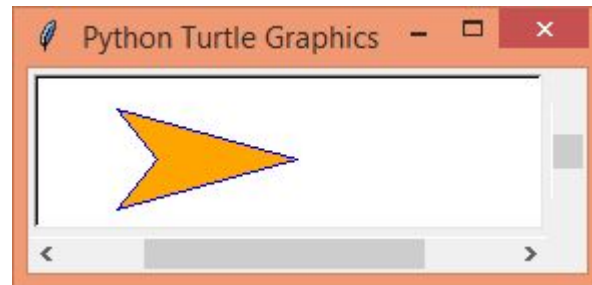


## Меняем внешний вид:

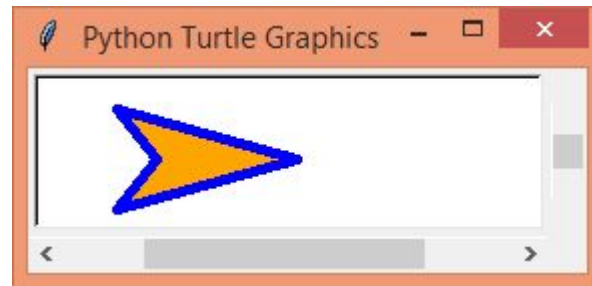


```
turtle.color('цвет контура','цвет заливки')  
# установка цветовой гаммы исполнителя
```

```
turtle.color('blue','orange')  
turtle.shapesize(5,10)
```



```
turtle.shapesize(5,10,5)
```





## *Цветовая палитра*

**Yellow** — жёлтый

**Green** — зелёный

**Blue** — голубой, синий

**Brown** — коричневый

**Red** — красный

**Pink** — розовый

**Black** — чёрный

**White** — белый

**Gray** — серый

**Orange** — оранжевый

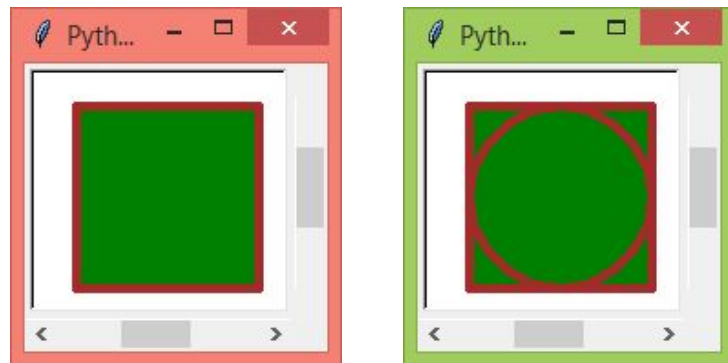


## Модуль *Turtle* (черепашка)



`turtle.stamp()` # отпечаток исполнителя на холсте

```
import turtle  
turtle.shape('square')  
  
turtle.color('red','green')  
  
turtle.shapesize(15,15,5)  
  
turtle.stamp()  
  
turtle.shape('circle')
```





## Модуль *Turtle* (черепашка)

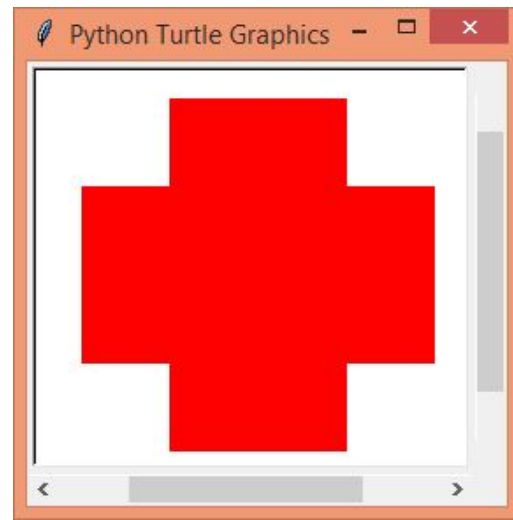
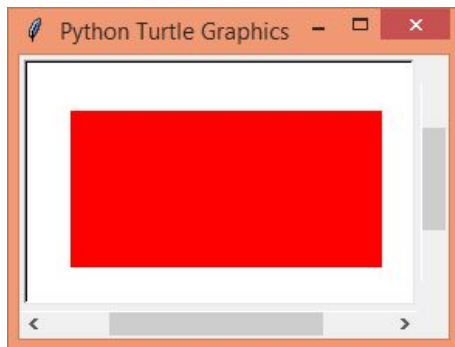


```
turtle.left(угол поворота)
```

```
# поворот влево на N градусов
```

```
import turtle  
turtle.shape('square')  
turtle.color('Red')  
turtle.shapesize(5,10,1)
```

```
turtle.stamp()  
turtle.left(90)  
turtle.shapesize(5,10,1)
```





## Модуль *Turtle* (черепашка)

```
import turtle
```

```
# Подключение модуля turtle
```

**Исполнитель**

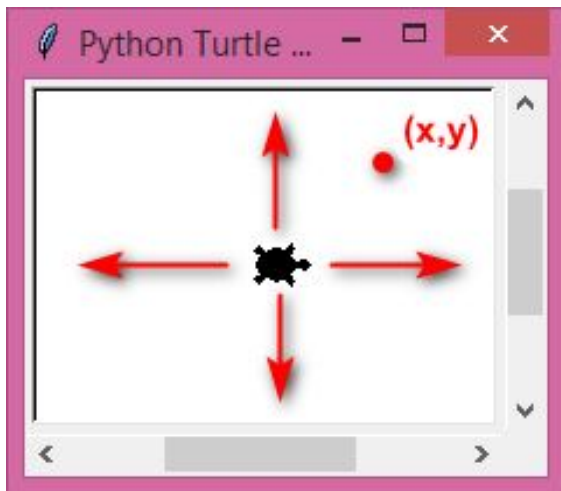






## Модуль *Turtle* (черепашка)

```
import turtle  
t = turtle.Turtle()  
t.shape('turtle')
```

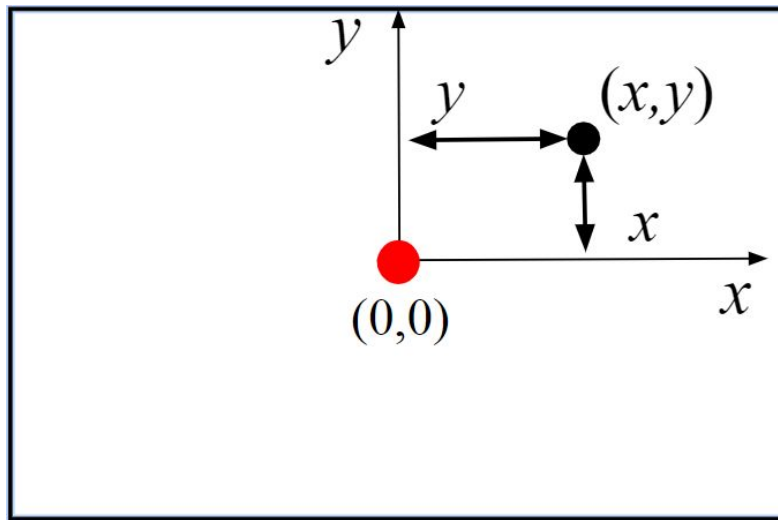


Управляется командами **относительных** («вперёд назад» и «направо налево») и **абсолютных** («перейти в точку с координатами...») **перемещений**.

Исполнитель представляет собой «перо», оставляющее след на плоскости рисования.



## Графическое окружение — холст



При работе в графическом режиме изображение на экране строится из точек, которые называются **пикселями**.

Каждый **пиксель** (точка) имеет две координаты: **x** и **y**.



## Задаем движение черепахи



**forward(n)** # вперед на  $n$  пикселей



**backward(n)** #назад на  $n$  пикселей

**left(n)** #влево на  $n$  градусов

**right(n)** #вправо на  $n$  градусов

**circle(r)** #начертить окружность радиуса  $r$ , с центром слева от курсора, если  $r > 0$ , справа, если  $r < 0$

**circle(r,n)** #начертить дугу радиуса  $r$ , градусной мерой  $n$  против часовой стрелки, если  $r > 0$ , по часовой стрелке, если  $r < 0$

**goto(x,y)** #переместить курсор в точку с координатами  $(x,y)$



## Команды рисования (управление пером)



***down()***    *#опустить курсор для рисования*



***up()***        *#поднять курсор*

***width(n)***    *#ширина следа курсора в n пикселей*

***color(s)***    *#где s цвет рисования курсора*

***begin\_fill(),end\_fill()***    *#рисует закрашенные области (начало и конец рисунка)*



## Сервисные команды:



**reset()** *#очищается экран, возвращает курсор к центру*



**clear()** *#очистить экран*

**write(s)** *#вывести строку s в точке нахождения курсора*

**radians()** *#мера измерения углов в радианы*

**degrees()** *#мера измерения углов в градусах*

**mainloop()** *#задержка окна*

**tracer(f)** *#режим отладки*



# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

### Основные команды:

Команда	Сокращение	Назначение команды
<code>forward (n)</code>	<code>fd()</code>	Проползти <b>вперёд</b> на $n$ пикселей;
<code>backward(n)</code>	<code>bk()</code>	Проползти <b>назад</b> на $n$ пикселей;
<code>right(angle)</code>	<code>rt()</code>	Повернуться <b>налево</b> на $angle$ градусов;
<code>left(angle)</code>	<code>lt()</code>	Повернуться <b>направо</b> на $angle$ градусов;
<code>goto(x, y)</code>		Переместить черепашку в точку с координатами $(x,y)$ ;
<code>circle(radius)</code>		Нарисовать <b>окружность</b> радиуса $ r $ , центр которой находится слева от черепашки, если $r>0$ и справа, если $r<1$ ;
<code>circle(r,angle)</code>		Нарисовать <b>дугу</b> радиуса $ r $ и градусной мерой $angle$ . Дуга рисуется против часовой стрелки, если $r>0$ и по часовой стрелке, если $r<0$ .
<code>circle(r, angle, n)</code>		Нарисовать <b>дугу</b> радиусом $r$ , с углом $angle$ и числом шагов $n$ . Чем больше число шагов, тем <b>плавнее</b> дуга. Например, нарисуем дугу радиусом 50 пикселей, с углом 180 градусов и числом шагов 100
<code>circle(r, 360, n)</code>		Нарисовать <b>многоугольник</b> с радиусом описанной окружности $r$ и числом сторон $n$ . Например, нарисуем шестиугольник с радиусом описанной окружности 100 пикселей
<code>color(s, z)</code>		Установить <b>цвет пера</b> $s$ и <b>цвет заливки</b> $z$
<code>pencolor (s)</code>		Установить <b>цвет пера</b>
<code>fillcolor(z)</code>		Установить <b>цвет заливки</b>
<code>dot(size, color)</code>		Нарисовать <b>точку</b> диаметра $size$ цвета $color$ .





# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

### Основные команды:

Команда	Сокращение	Назначение команды
<code>clear()</code>		Очистка экрана.
<code>write(s)</code>		Вывести текстовую строку <code>s</code> в точке нахождения черепашки.
<code>shape()</code>		Изменить значок черепахи (" <code>arrow</code> ", " <code>turtle</code> ", " <code>circle</code> ", " <code>square</code> ", " <code>triangle</code> ", " <code>classic</code> ")
<code>stamp()</code>		Нарисовать копию черепахи в текущем месте
<code>begin_fill()</code>		Начать заливку. Необходимо вызвать перед рисованием фигуры, которую надо закрасить
<code>end_fill()</code>		Остановить заливку. Вызвать после окончания рисования фигуры (конец заливки)
<code>home()</code>		Вернуть черепашку <b>домой</b> — в точку, с координатами (0,0);
<code>speed(n)</code>		Установить <b>скорость</b> черепашки. <code>speed</code> должно быть от 1 (медленно) до 10 (быстро), или 0 (мгновенно);
<code>width(n)</code>		Установить <b>ширину</b> следа черепашки в <code>n</code> пикселей.
<code>reset()</code>		<b>Возврат</b> черепашки в исходное состояние: <b>очищается</b> экран
<code>down()</code>	<code>pd()</code>	Опустить перо.
<code>up()</code>	<code>pu()</code>	Поднять перо

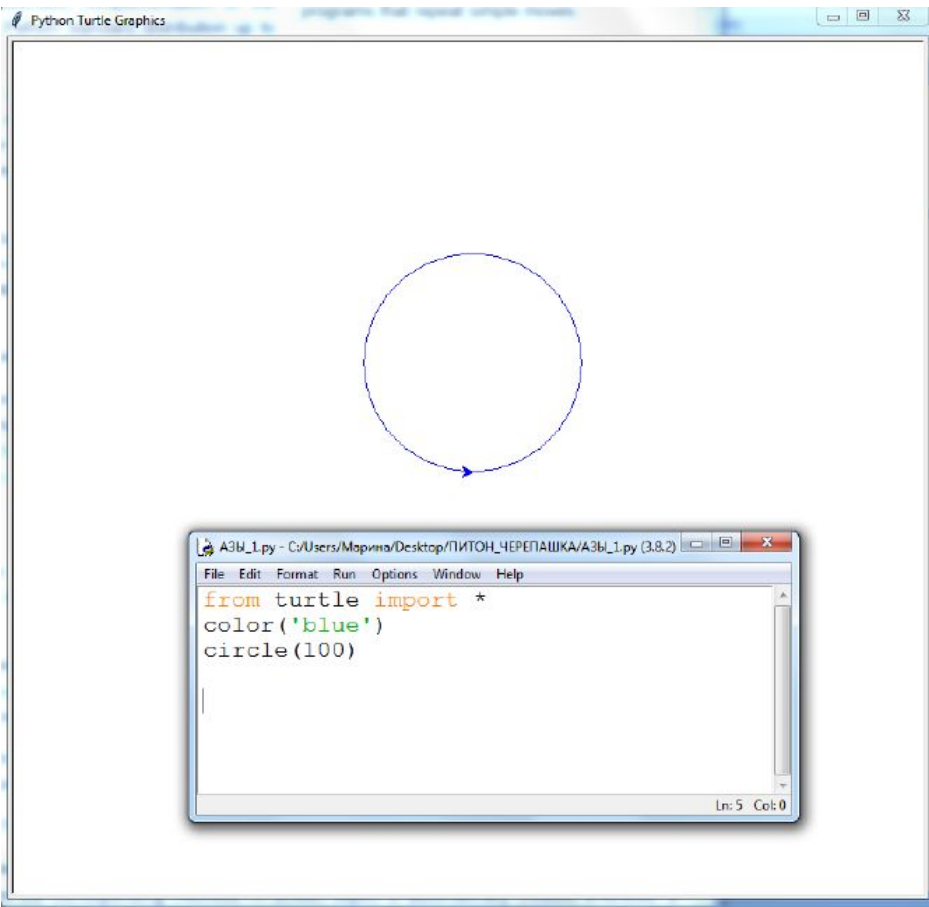




## ПРОГРАММИРОВАНИЕ. 3.9. «Работа с графикой в Python»

### Пример 1

Нарисуем командой **circle** синюю окружность.



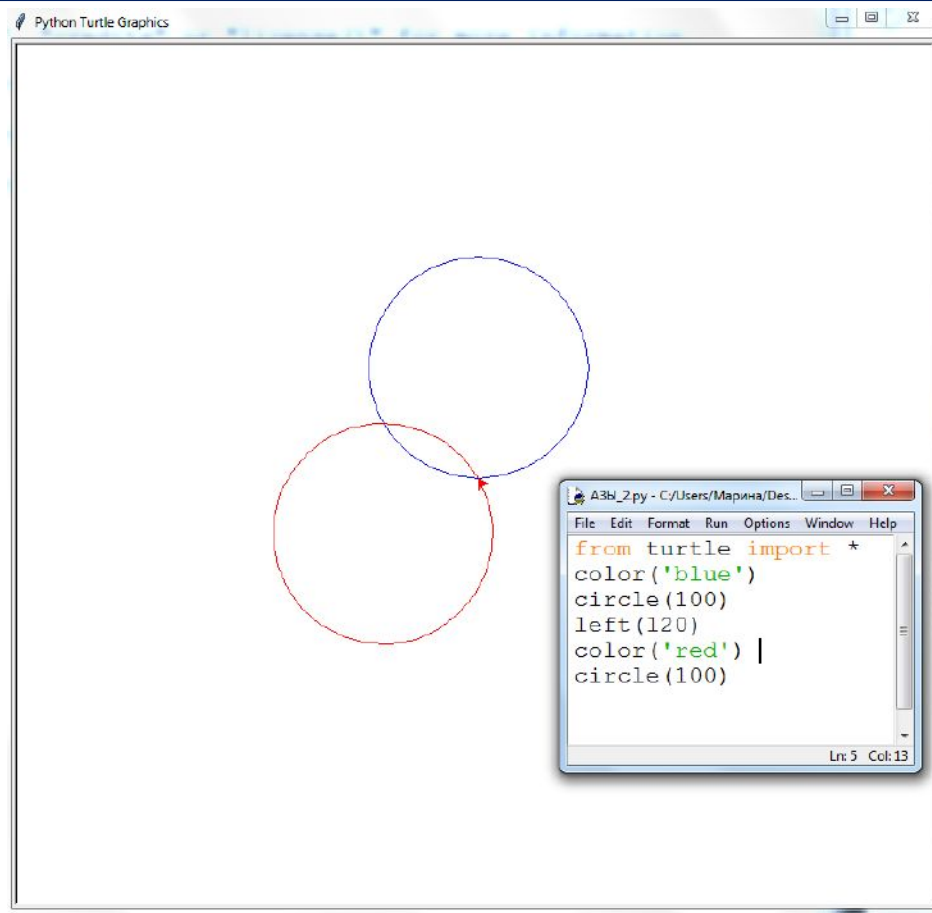




## ПРОГРАММИРОВАНИЕ. 3.9. «Работа с графикой в Python»

### Пример 2

*Повернём черепашку на 120 градусов влево и добавим красную окружность:*

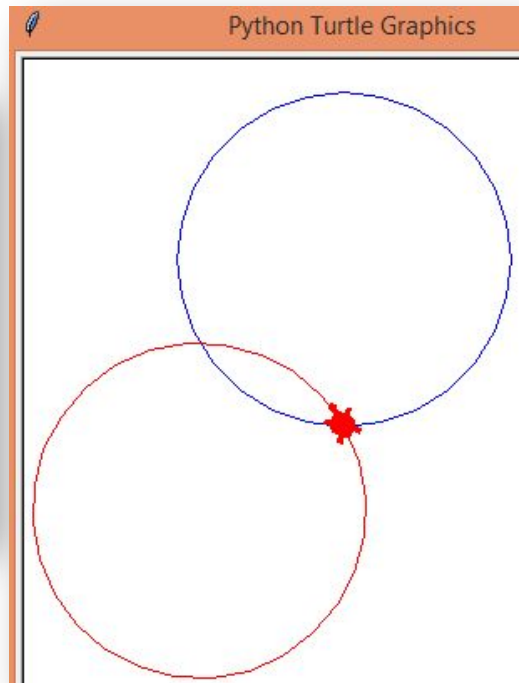
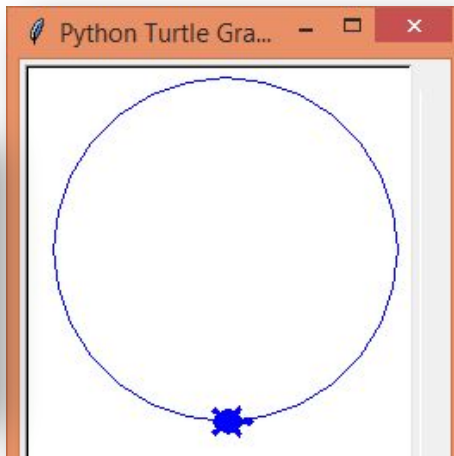




## Пример 2

### Рисуем окружности

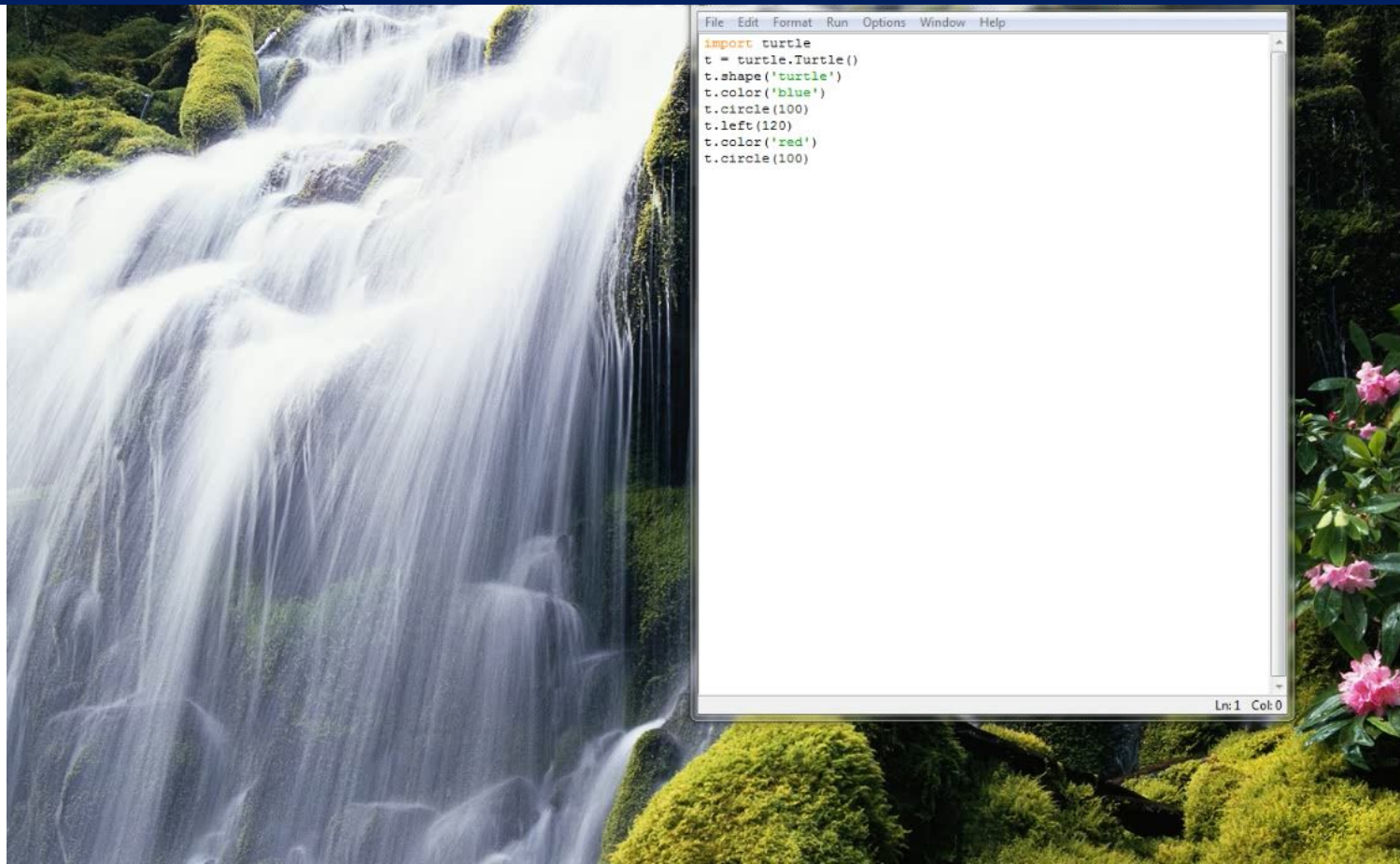
```
import turtle  
t = turtle.Turtle()  
t.shape('turtle')  
t.color('blue')  
t.circle(100)  
t.left(120)  
t.color('red')  
t.circle(100)
```





# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»





## ПРОГРАММИРОВАНИЕ. 3.9. «Работа с графикой в Python»

### Пример 3

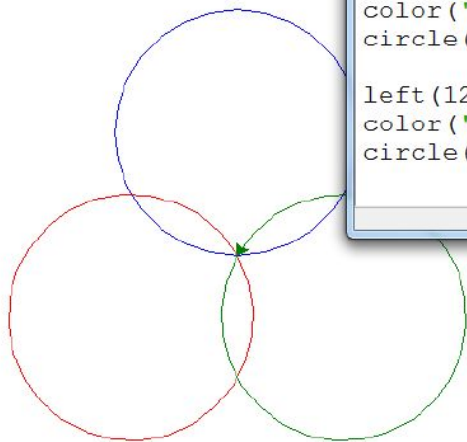
*Ещё раз повернём черепашку на 120 градусов влево и добавим зелёную окружность:*

```
File Edit Format Run Options Window Help
from turtle import *
color('blue')
circle(100)

left(120)
color('red')
circle(100)

left(120)
color('green')
circle(100)

Ln: 10 Col: 0
```





```
import turtle
```

```
t = turtle.Turtle()
```

```
t.shape('turtle')
```

```
t.color('blue')
```

```
t.circle(100)
```

```
t.left(120)
```

```
t.color('red')
```

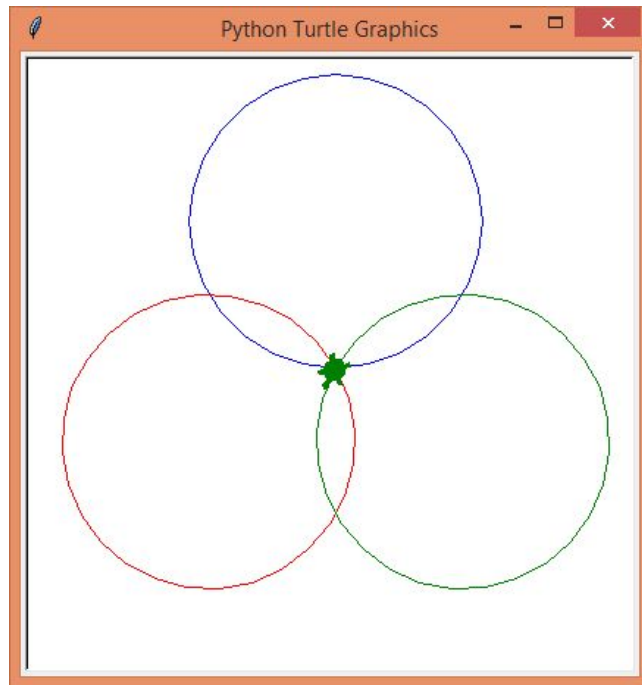
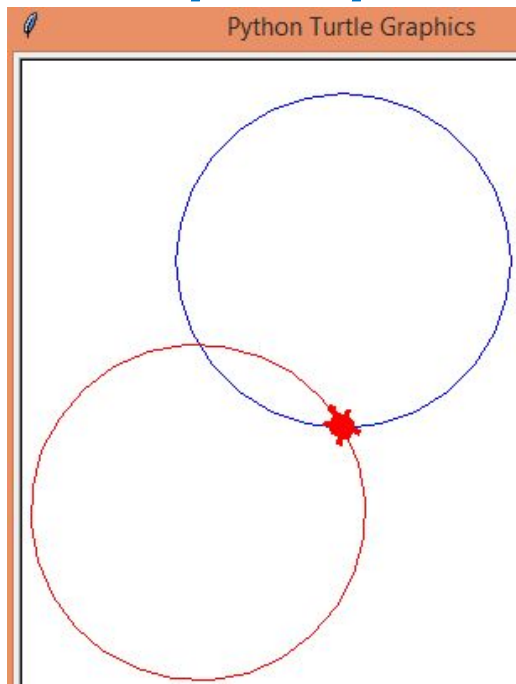
```
t.circle(100)
```

```
t.left(120)
```

```
t.color('green')
```

```
t.circle(100)
```

## Пример 3



*Рисуем окружности*

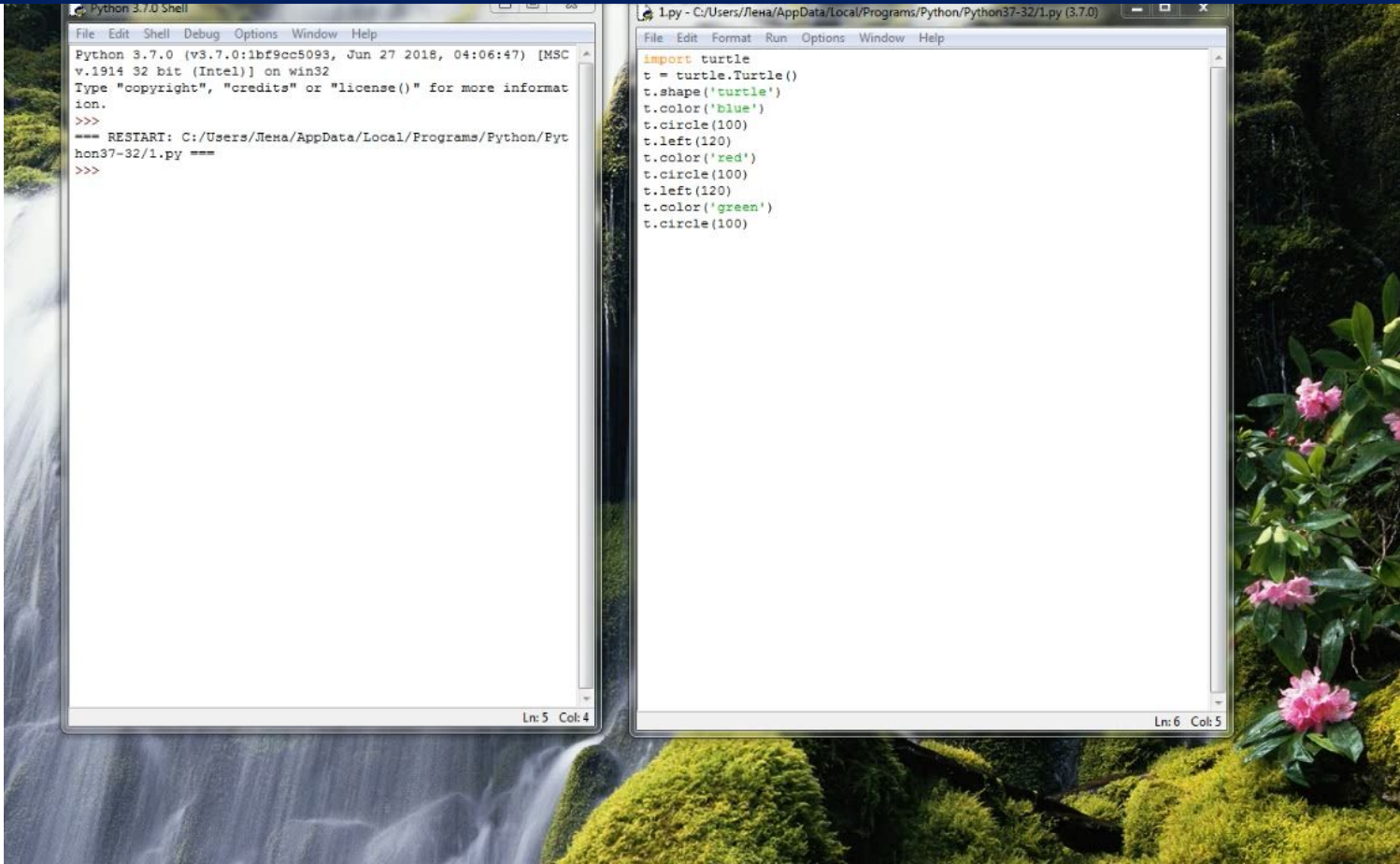


# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

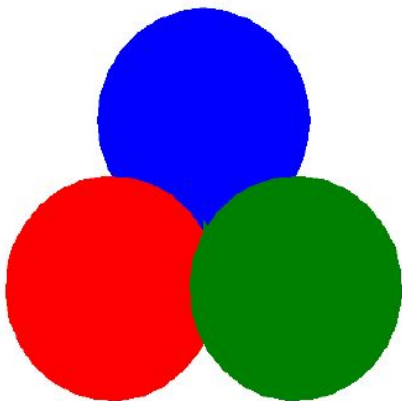
```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC
v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more informat
ion.
>>>
=== RESTART: C:/Users/Лена/AppData/Local/Programs/Python/Pyt
hon37-32/1.py ===
>>>
```

```
lpy - C:/Users/Лена/AppData/Local/Programs/Python/Python37-32/1.py (3.7.0)
File Edit Format Run Options Window Help
import turtle
t = turtle.Turtle()
t.shape('turtle')
t.color('blue')
t.circle(100)
t.left(120)
t.color('red')
t.circle(100)
t.left(120)
t.color('green')
t.circle(100)
```





## ПРОГРАММИРОВАНИЕ. 3.9. «Работа с графикой в Python»



```
File Edit Format Run Options Window Help
from turtle import *

color('blue','blue') # цвет пера, цвет заливки
begin_fill() # начать заливку
circle(100)
end_fill() # остановить заливку

left(120)
color('red','red')
begin_fill()
circle(100)
end_fill()

left(120)
color('green','green')
begin_fill()
circle(100)
end_fill()
```

Ln: 3 Col: 46

### Пример 4

Добавляем в `color` цвет заливки и команды:

«начать заливку» и «остановить заливку».

Сохраняем программу в файле **`grafika1.py`**



# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

```
turtle.stamp()
```

```
turtle.shape('triangle')
```

```
turtle.color('blue', 'orange')
```

```
turtle.shape('turtle')
```

```
turtle.shape('circle')
```

Команда  
устанавливает  
размер формы

```
turtle.shape('arrow')
```

### Задание

«Совместите команду и  
результат выполнения»



Команда  
устанавливает  
цвет формы

```
turtle.shapesize(5, 8, 10)
```

Оставляет  
отпечаток формы  
выбранного  
цвета и размера



```
turtle.shape('square')
```





## ПРОГРАММИРОВАНИЕ. 3.9. «Работа с графикой в Python»

```
turtle.stamp()
```

```
turtle.shape('triangle')
```

```
turtle.color('blue', "orange" )
```

```
turtle.shape('turtle')
```

```
turtle.shape('circle')
```

Команда  
устанавливает  
размер формы



Команда  
устанавливает  
цвет формы

```
turtle.shape('arrow')
```

```
turtle.shapesize(5, 8, 10)
```



Оставляет  
отпечаток формы  
выбранного  
цвета и размера



```
turtle.shape('square')
```



# ПРОГРАММИРОВАНИЕ.

## 3.9. «Работа с графикой в Python»

```
turtle.stamp()
```

Оставляет отпечаток формы выбранного цвета и размера

```
turtle.shapesize(5, 8, 10)
```

Команда устанавливает размер формы

```
turtle.color('blue', "orange" )
```

Команда устанавливает цвет формы

МОЛОДЕЦ !



```
arrow')
```

```
turtle.shape('circle')
```



```
turtle.shape('triangle')
```



```
turtle.shape('square')
```



```
turtle.shape('turtle')
```





**1**

**Графические режимы в Python**

**2**

**Рисование с помощью модуля Turtle**



**3**

**Управление модулем Turtle (черепашка) для создания графических примитивов и перемещения на плоскости.**

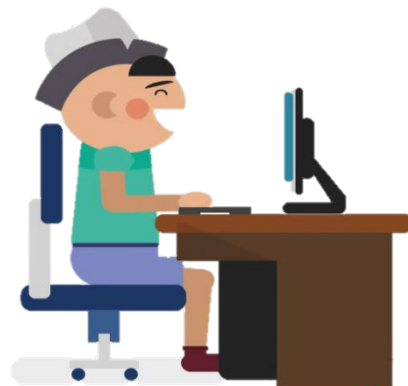
**4**

**Команды управления графическим пером.**

Глава

3

**ДО НОВЫХ  
ВСТРЕЧ!**



***В подготовке данного урока  
использовались материалы  
образовательно-методического  
Интернет-ресурса для учителей  
<https://videouroki.net>***



***Урок разработала  
Клепачёва Е.А.,  
учитель информатики УК АФМШЛ №61***