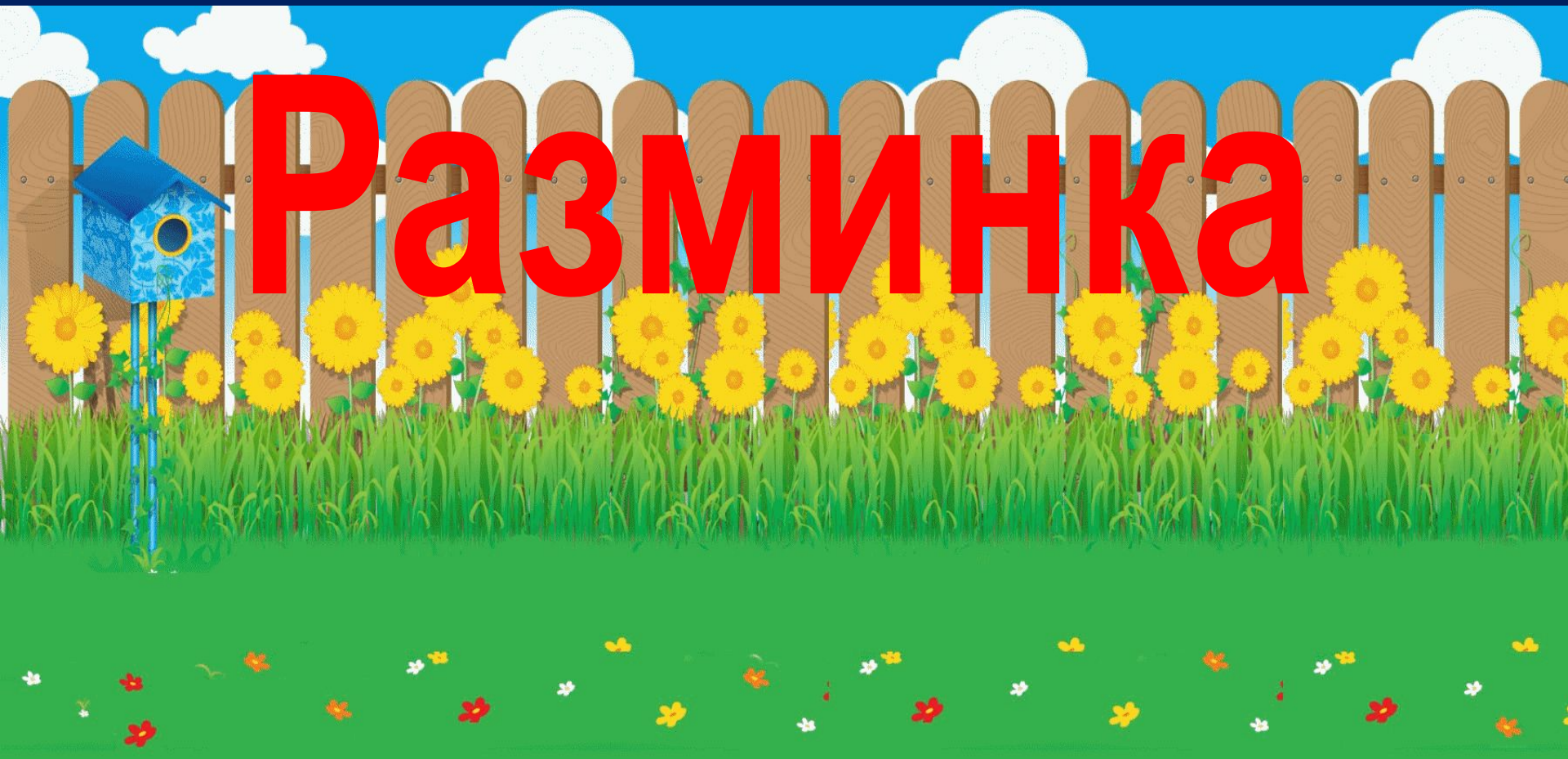
ИНФОРМАТИКА

8

ИНФОРМАТИКА



ИНФОРМАТИКА

0

ИНФОРМАТИКА

ГРАФИКА

ИНФОРМАТИКА

0

ИНФОРМАТИКА

ПИТОН

ИНФОРМАТИКА

0

ИНФОРМАТИКА

ЧЕРЕПАШКА

ИНФОРМАТИКА

ИНФОРМАТИКА 8 КЛАСС

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

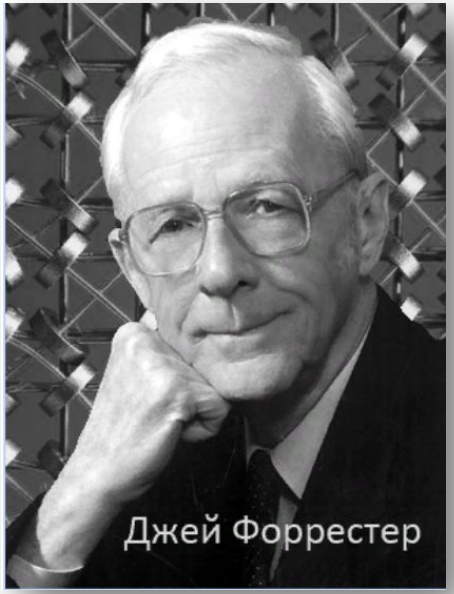
2 Рисование с помощью модуля Turtle

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

4 **Команды управления графическим пером.**

Компьютерная графика

ЭТО ИНТЕРЕСНО



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

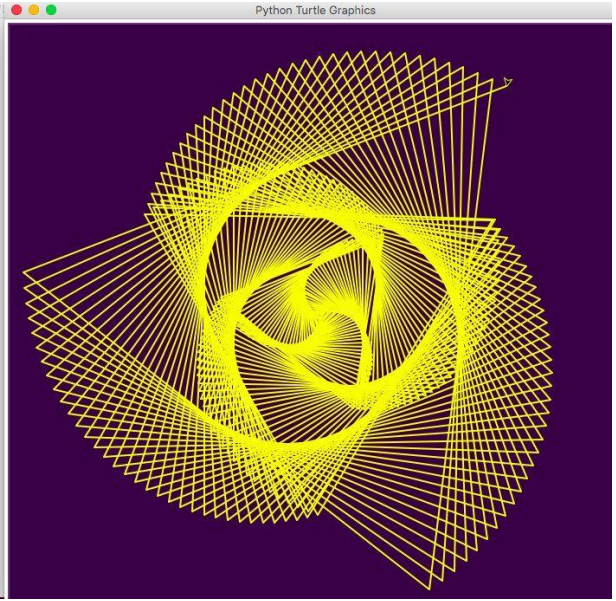
Компьютерная графика *— раздел информатики, предметом изучения которого является создание и обработка на компьютере графических изображений.*

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()
```



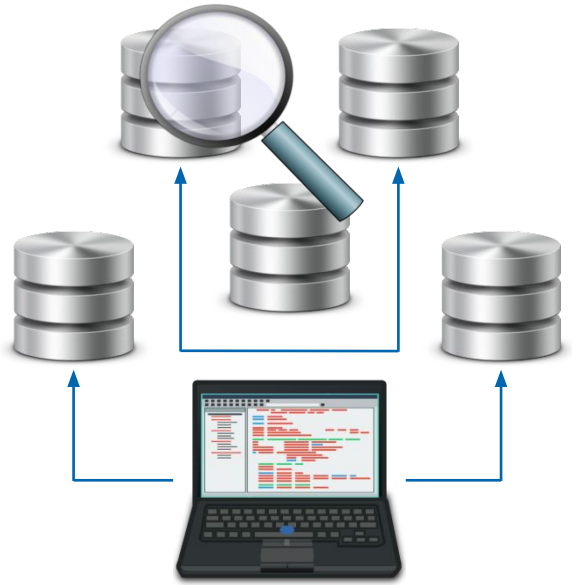
Графический модуль

***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.*

Графический модуль

Графический модуль



Графический модуль

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

*модуль **random** позволяет генерировать псевдослучайные числа*

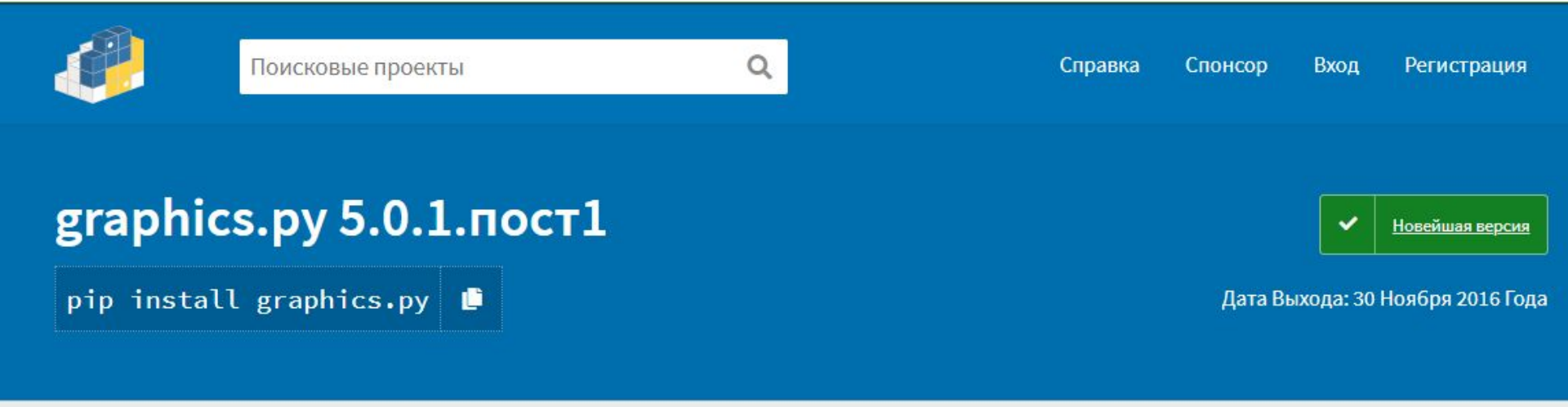
*модуль **sys** предоставляет доступ к системным переменным*

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

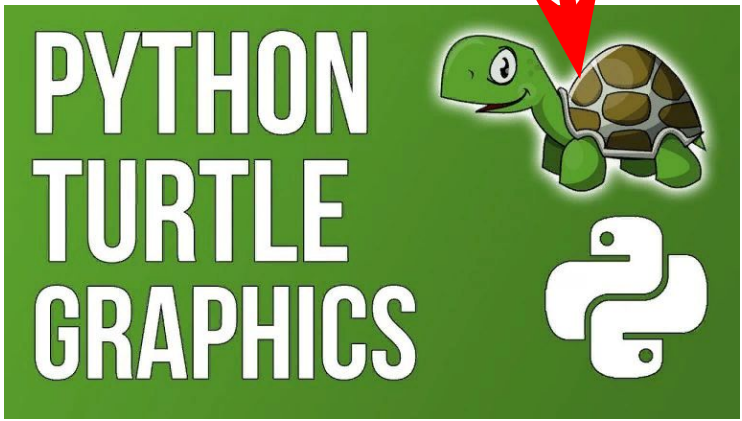
graphics.py



Модуль Turtle (черепашка)

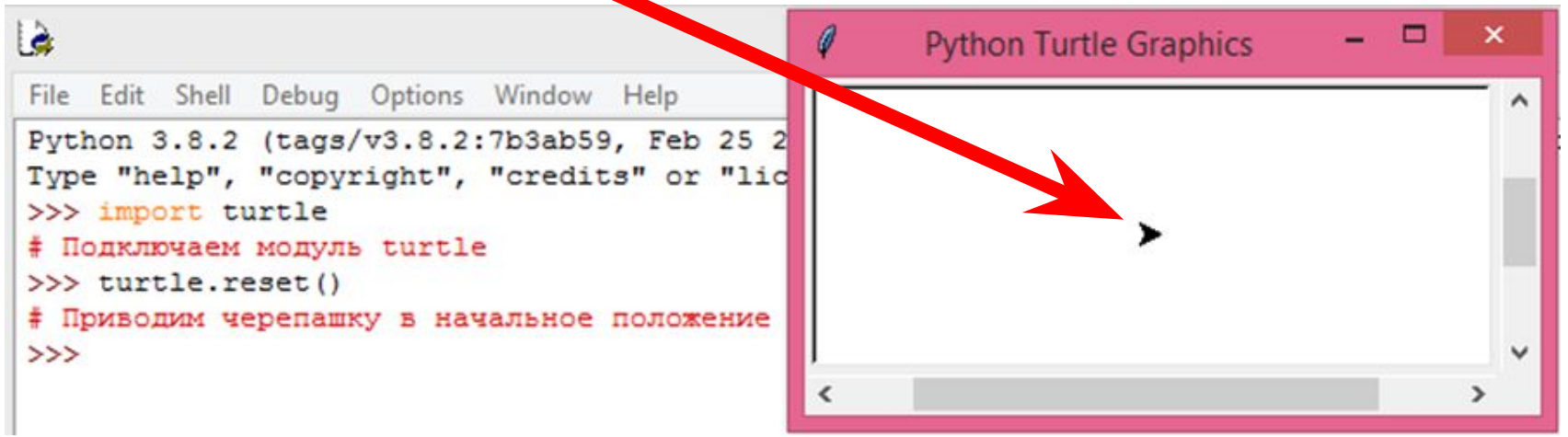
Исполнитель

```
import turtle
# подключаем модуль turtle
```



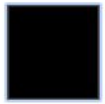
Модуль Turtle (черепашка)

```
turtle.reset()
# приводим черепашку в начальное положение
```

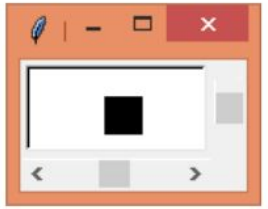


Модуль Turtle (черепашка)

```
turtle.shape(“стиль”)   # меняем внешний вид
```

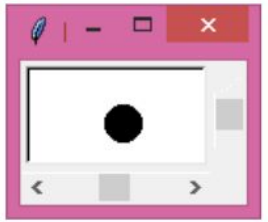


turtle.shape('square')



turtle.shape("turtle")

turtle.shape('circle')

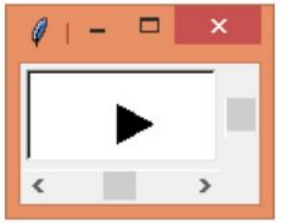


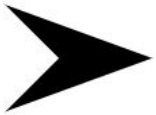
Модуль Turtle (черепашка)

```
turtle.shape(“стиль”)   # меняем внешний вид
```

turtle.shape("arrow")

turtle.shape('triangle')





turtle.shape('classic')

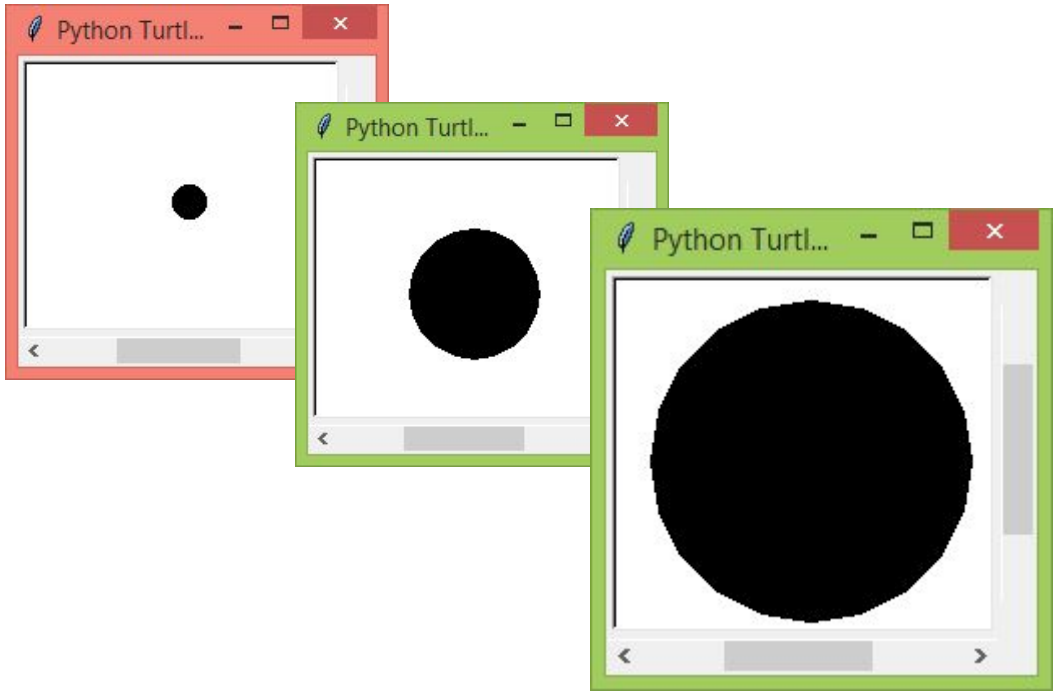
Модуль Turtle (черепашка)

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

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

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

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



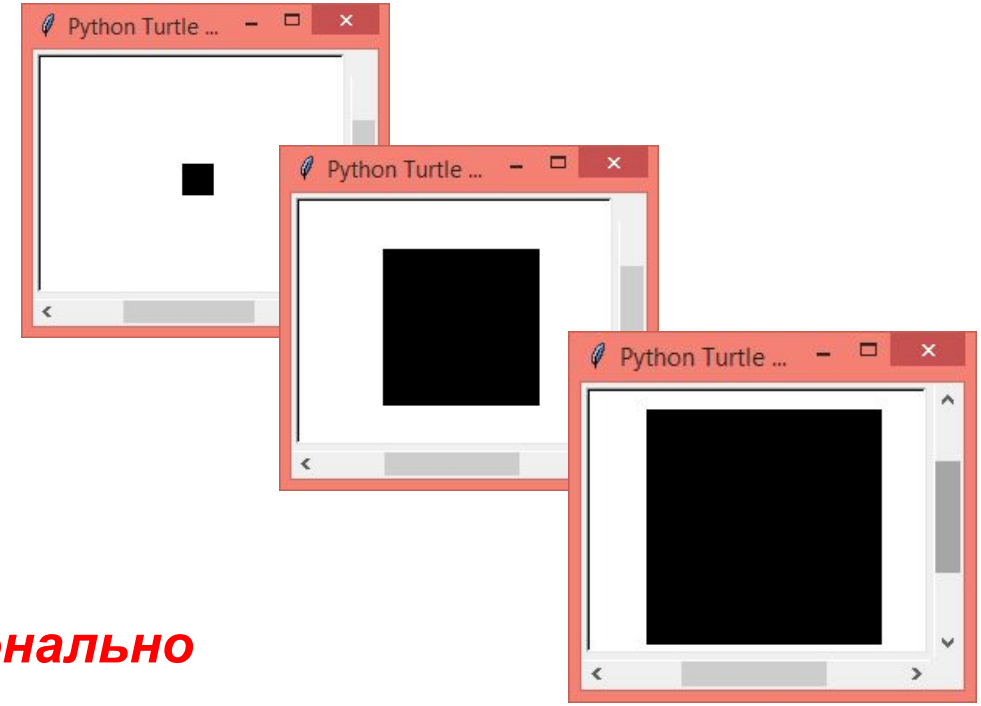
Модуль Turtle (черепашка)

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

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

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

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

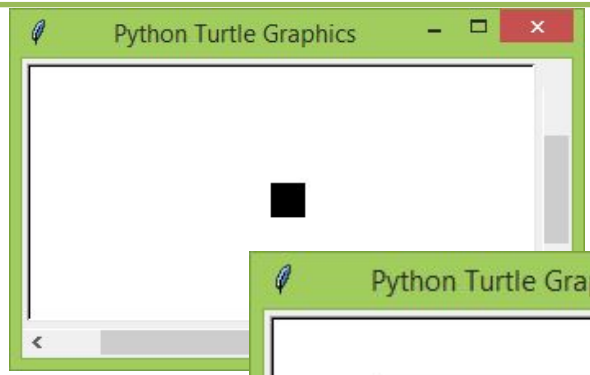


Изменение размера пропорционально

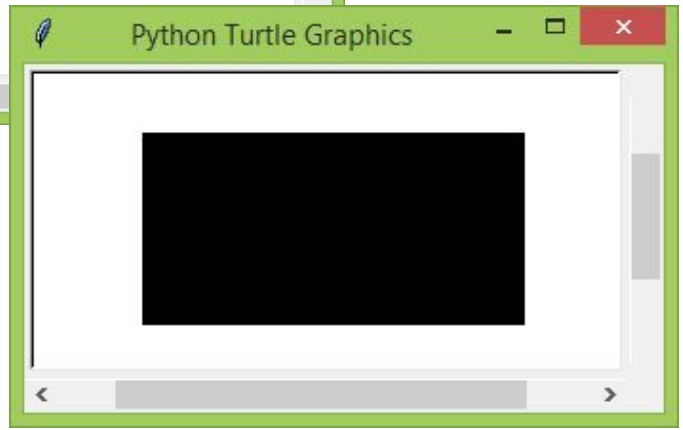
Модуль Turtle (черепашка)

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

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

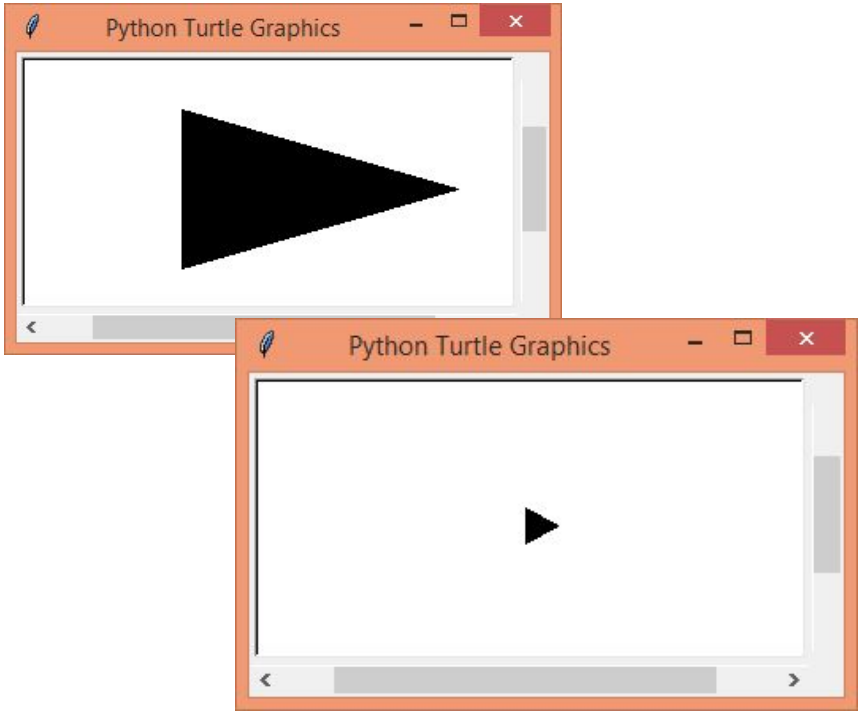



```
turtle.shapesize(5,10)
```



Меняем внешний вид:

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

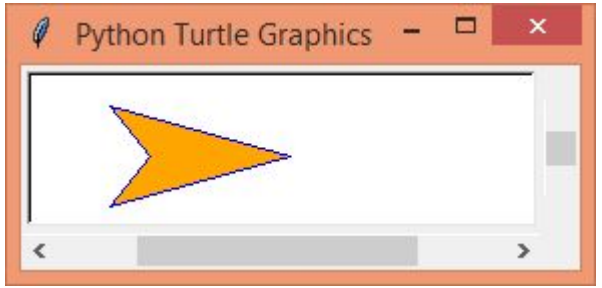



```
turtle.reset()
```

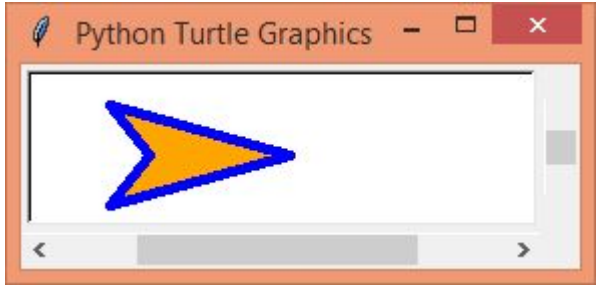
Меняем внешний вид:

```
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

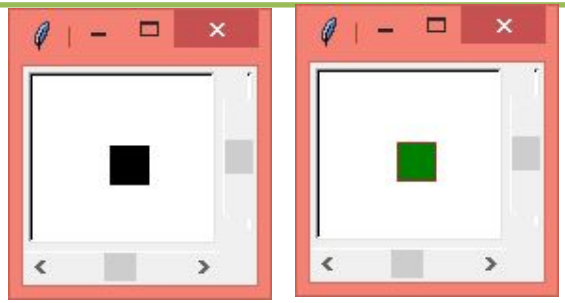
Gray — серый

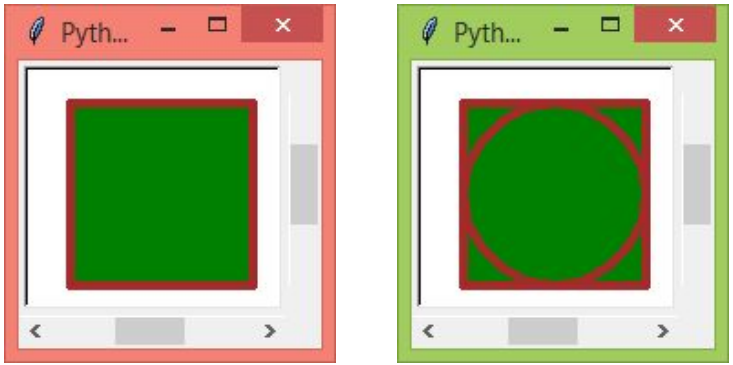
Orange — оранжевый

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')
```

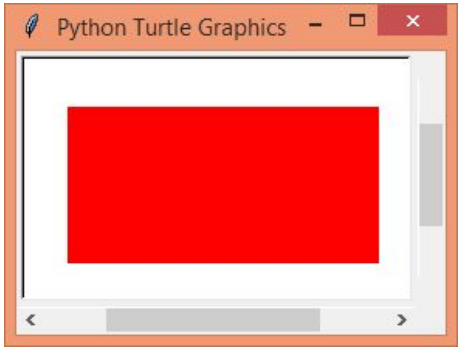


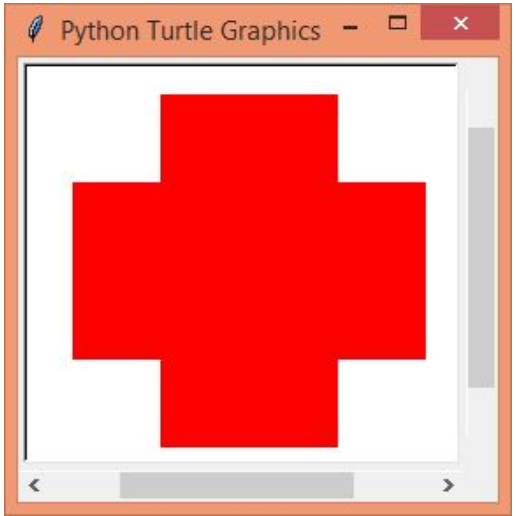


Модуль Turtle (черепашка)

```
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)
```





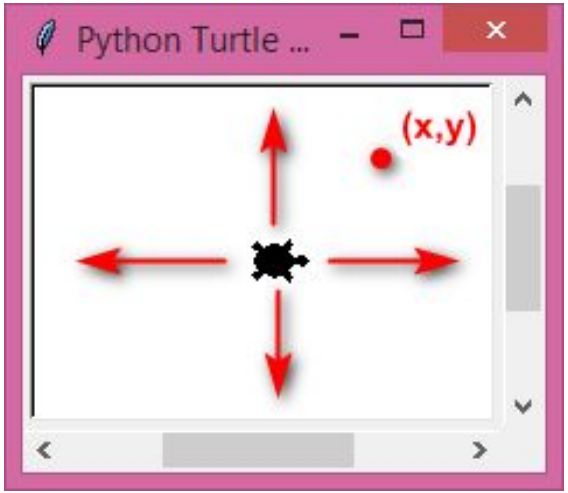
Модуль Turtle (черепашка)

```
import turtle
# Подключение модуля turtle
```

Исполнитель

Модуль Turtle (черепашка)

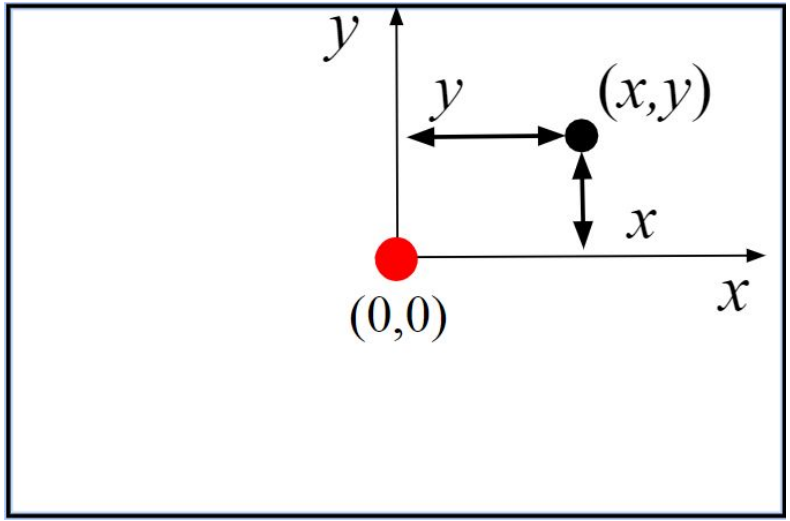
```
import turtle
t = turtle.Turtle()
t.shape('turtle')
```



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*

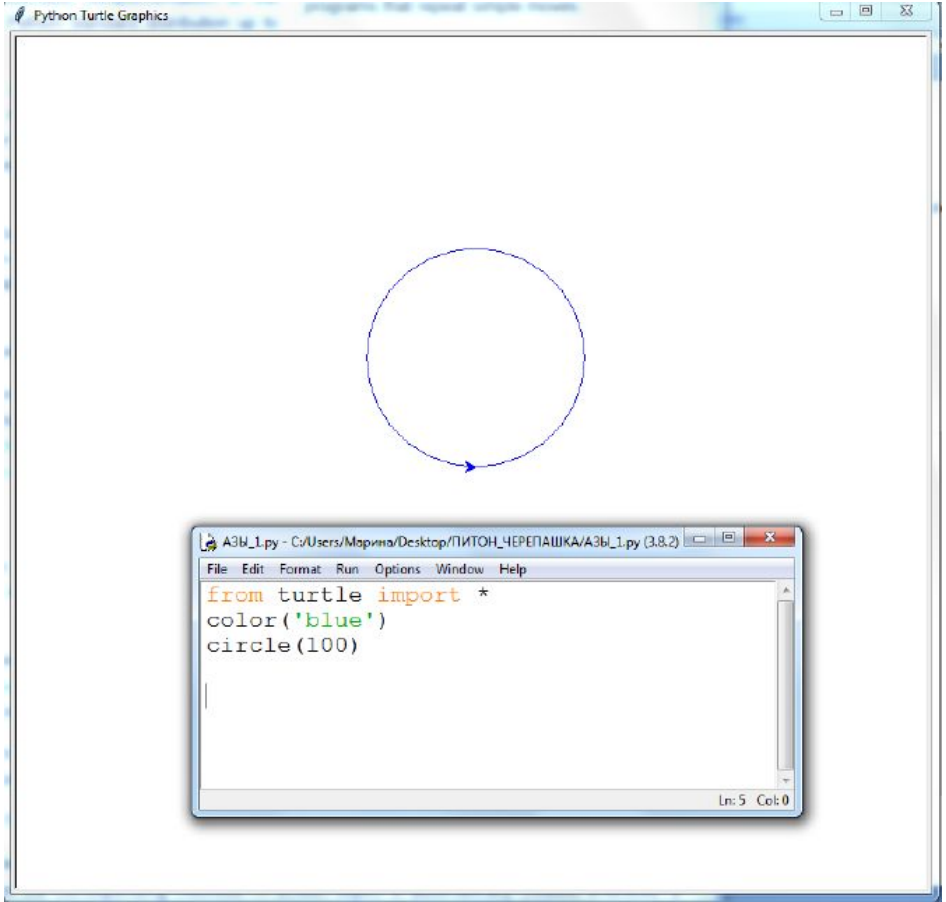
tracer(f) *#режим отладки*

Основные команды:

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

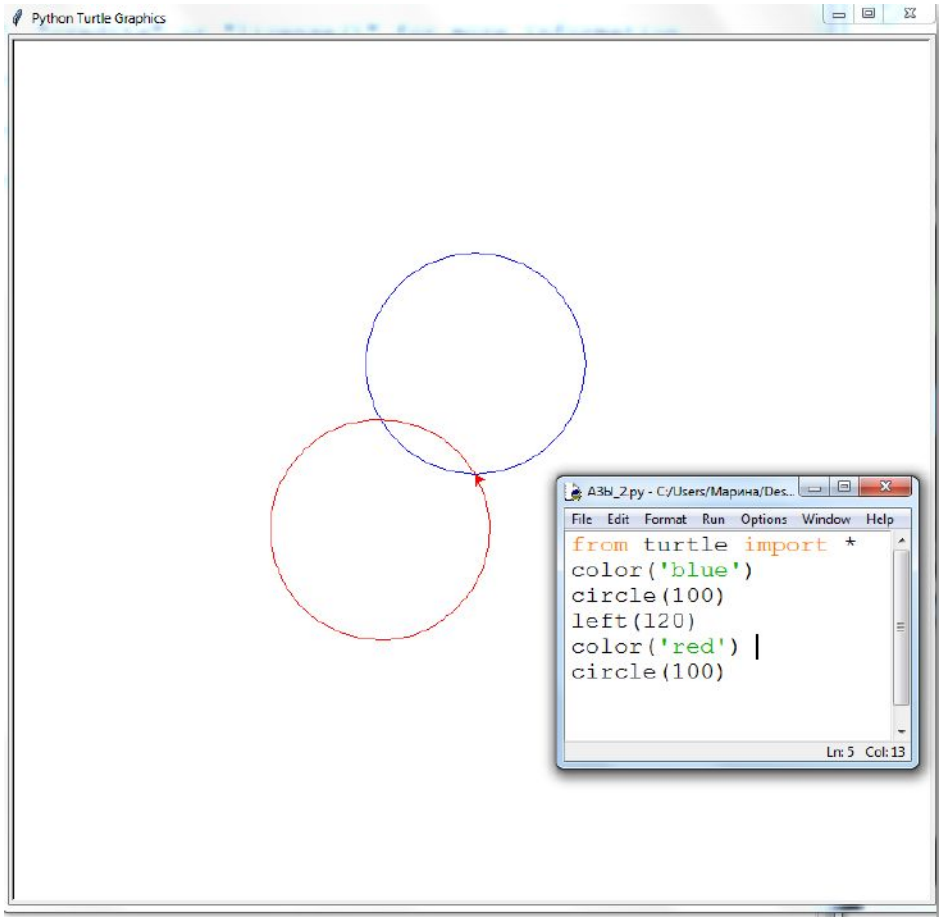
Основные команды:

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



Пример 1

*Нарисуем командой **circle** синюю окружность.*



Пример 2

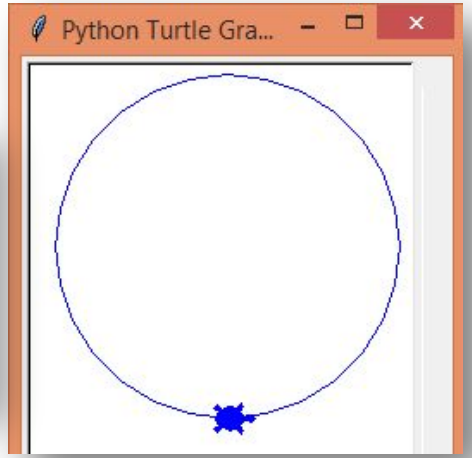
Повернём черепашку на 120 градусов влево и добавим красную окружность:

Пример 2

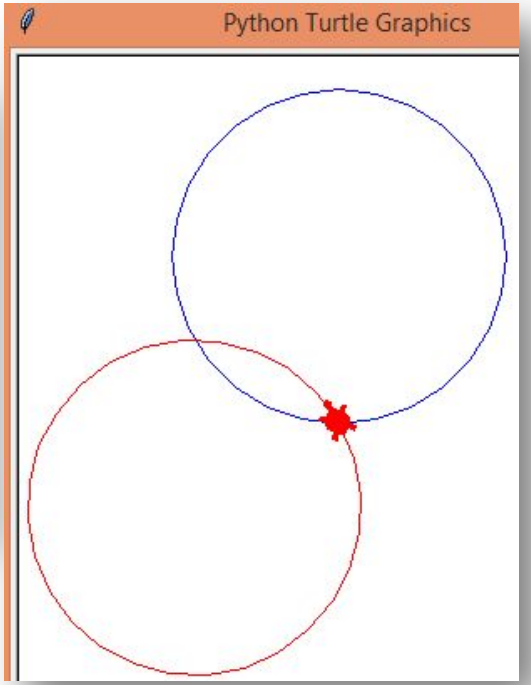
Рисуем окружности

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

1

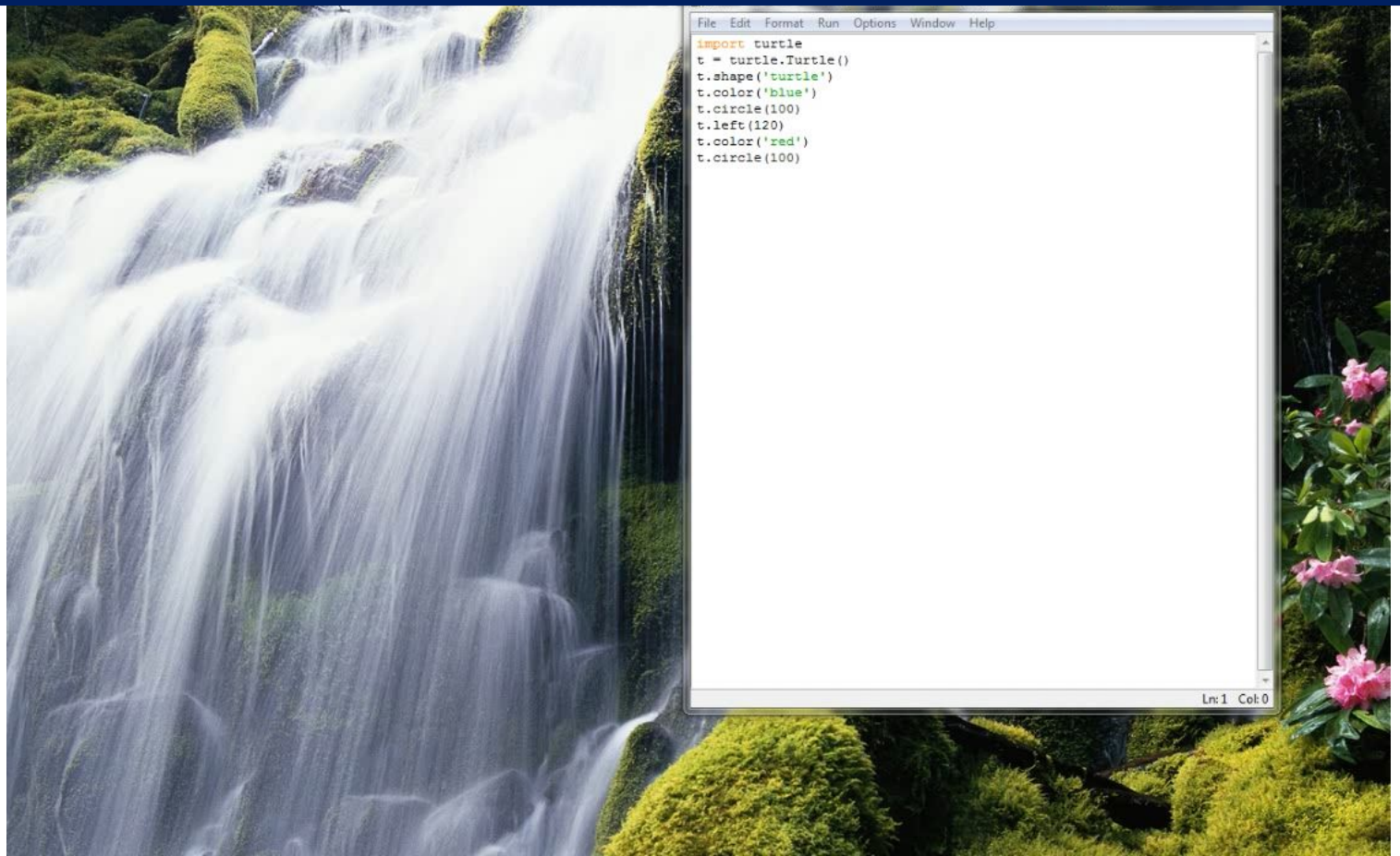


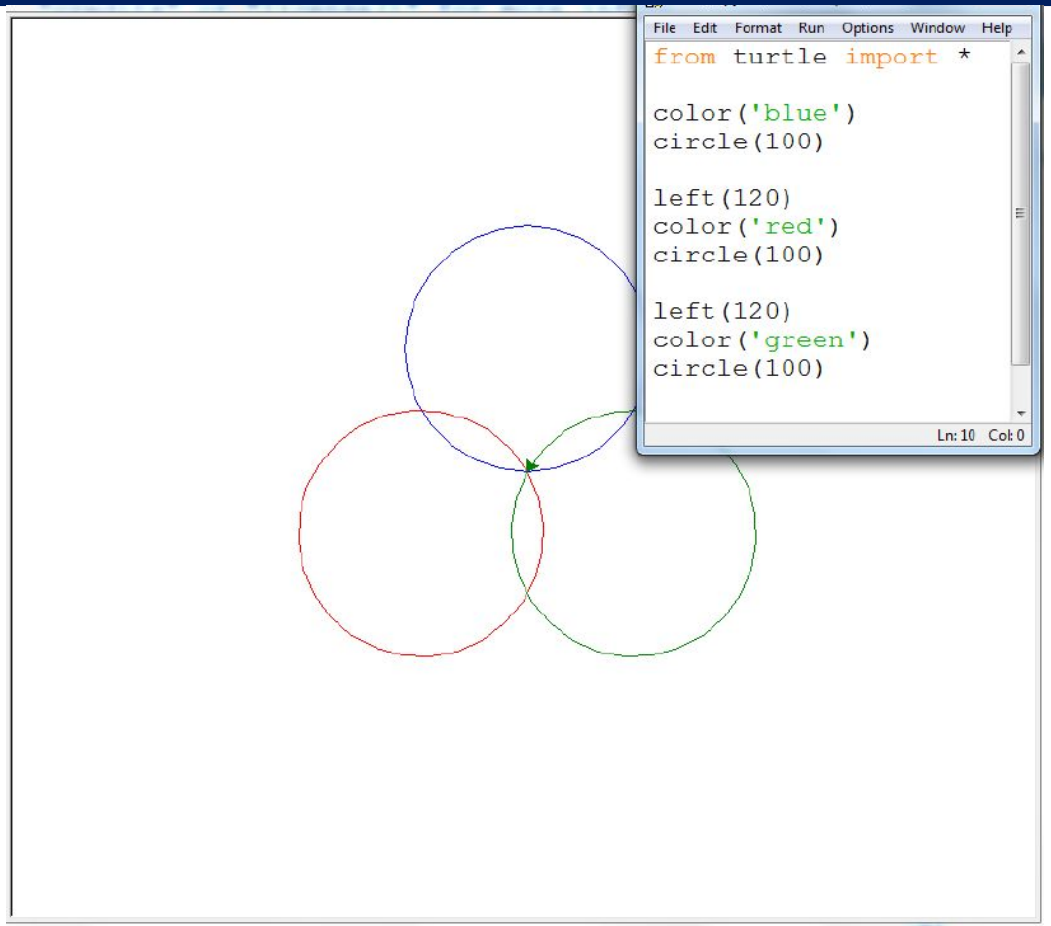
2



3

ПРОГРАММИРОВАНИЕ.
3.9. «Работа с графикой в Python»



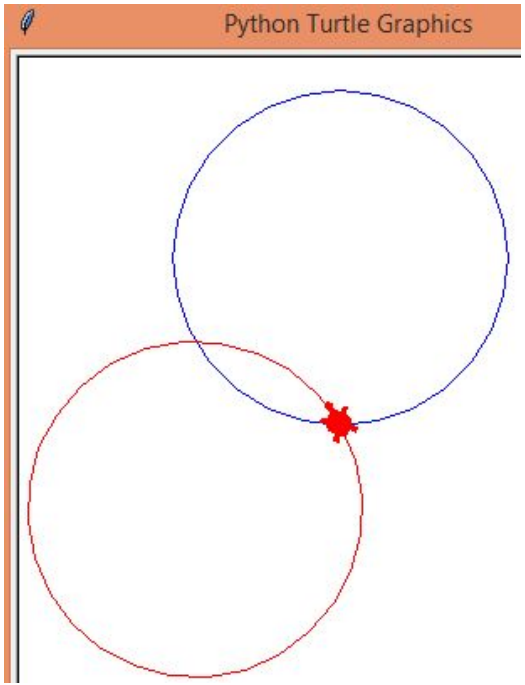


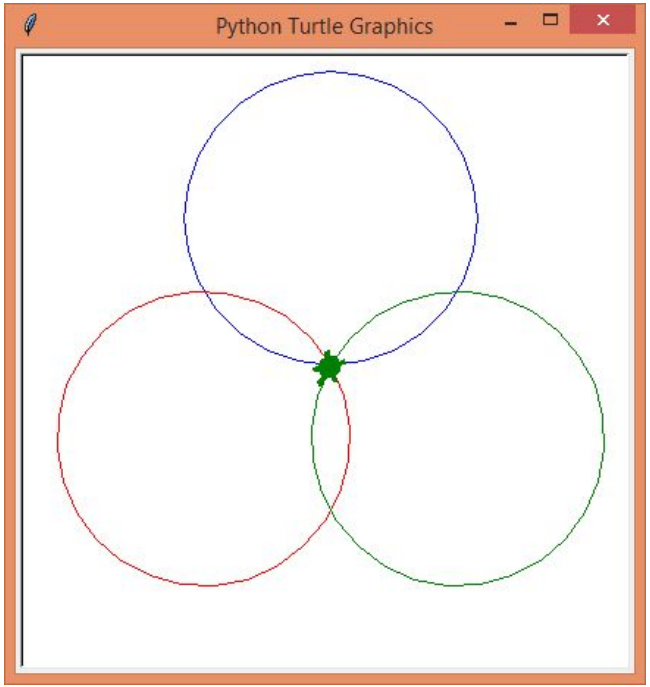
Пример 3

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

Пример 3

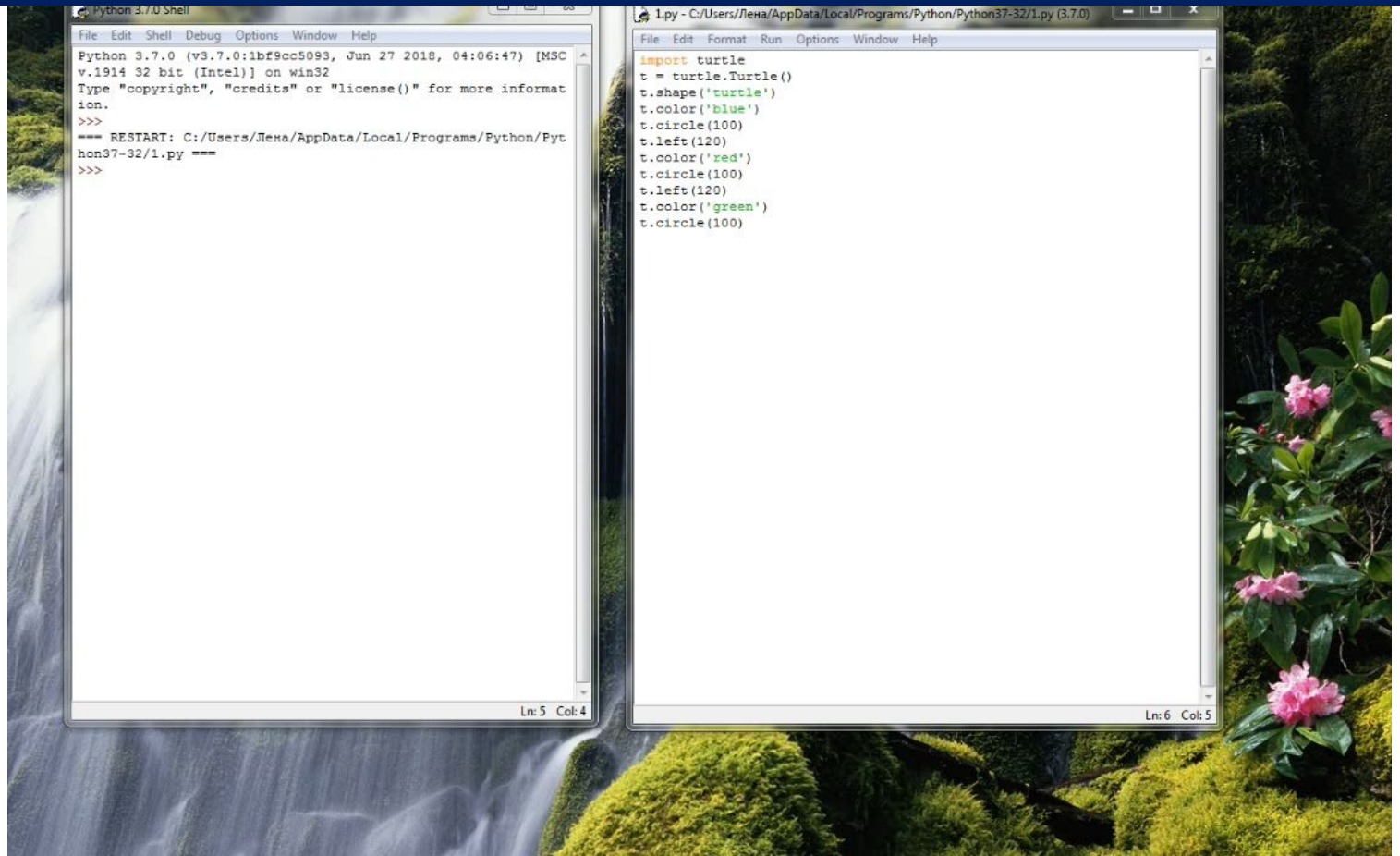
```
import turtle
t = turtle.Turtle()
t.shape('turtle')
t.color('blue')
t.circle(100)
t.left(120)
t.color('red')
t.circle(100)
t.left(120)
t.color('green')
t.circle(100)
```

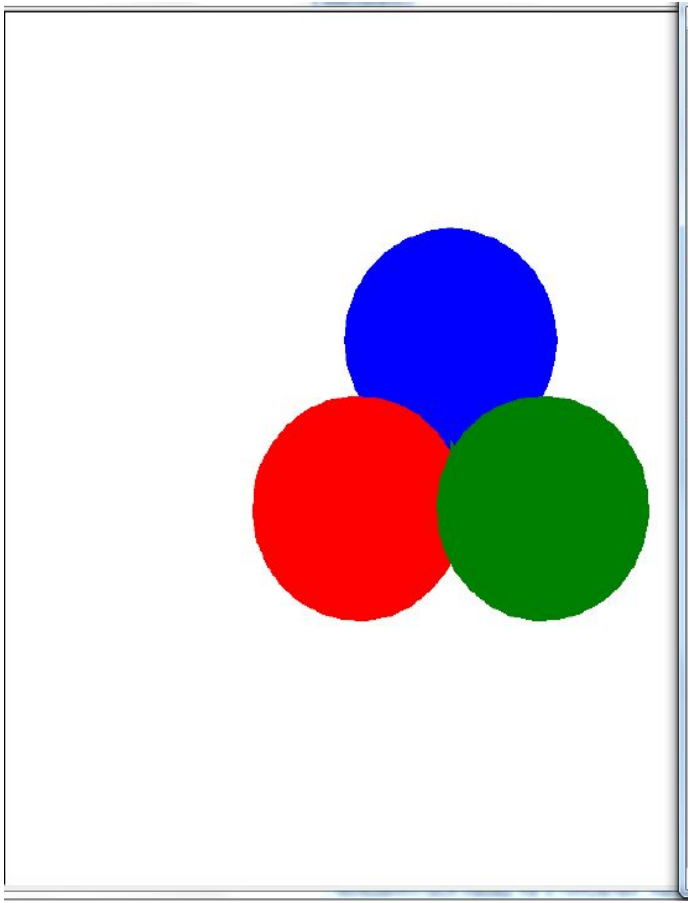




Рисуем окружности

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: 45

Пример 4

Добавляем в color цвет заливки и команды: «начать заливку» и «остановить заливку». Сохраняем программу в файле ***grafika1.py***

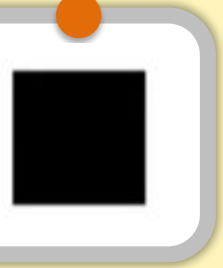
```
turtle.stamp()
```

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

```
turtle.color('blue', "orang
```

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

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



Команда устанавливает цвет формы

Задание

«Совместите команду и результат выполнения»

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

Оставляет отпечаток формы выбранного цвета и размера

Команда устанавливает размер формы

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

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

```
turtle.stamp()
```

Команда устанавливает цвет формы

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

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

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

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

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

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

0

Оставляет отпечаток формы выбранного цвета и размера

Команда устанавливает размер формы

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

```
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 !

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

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

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

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

1 Графические режимы в Python

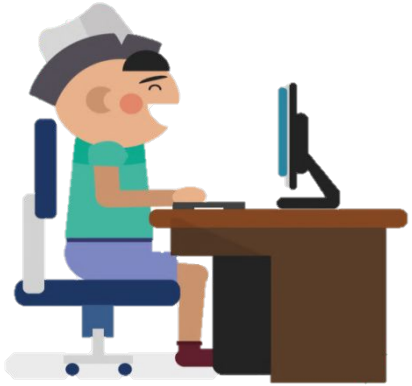
2 Рисование с помощью модуля Turtle

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