Конспект 4. Графический модуль turtle в Python

Учитель информатики Батракова Л.В.

В этом уроке мы разберём графику в Python с помощью модуля **Turtle черепашка**. **Turtle** это модуль для Python, позволяющий создавать графические объекты, риснуки в специальном окне. Модуль **Turtle** можно использовать для создания игр на питоне.

Чтобы присоединить (импортировать) модуль **turtle** необходимо выбрать один из следующих способов:

1. **import turtle**
2. **from turtle import***
3. **from turtle import open as t**

Первый способ импортирует модуль, но команды нужно писать с упоминанием модуля (как ссылку на атрибут) в стиле ООП. Например, **turtle.reset()**, что загромождает исходный код и создаёт неудобочитаемость программы.
Второй способ разрешает обращаться к функциям модуля непосредственно без упоминания имени модуля.
Третий способ позволяет подменить название модуля своим именем, т. е. вместо того, чтобы писать полностью имя модуля мы будем использовать только один символ, например:

```
t.reset()
t.fd(100)
```

К особенностям работы с модулем **turtle** относится предварительный вызов графического окружения — так называемого **холста,** т.е. поля по которому перемещается исполнитель. Для этого используется команда без параметров **reset()**. Она же сбрасывает все настройки по умолчанию и возвращает исполнителя в исходное состояние. (Рис. 1).

Внешний вид исполнителя (по умолчанию стиль **classic** черного цвета) — наконечник стрелы всегда направленный в сторону движения.

Существует несколько команд определяющих внешний вид исполнителя:

1. **turtle.shape(«Style»)** - изменяет внешний вид
2. **turtle.shapesize(m,n)** — управляет размерами исполнителя
3. **turtle.tilt(alpha)** — изменяет ориентацию исполнителя
4. **turtle.hideturtle()** - скрывает черепаху
5. **turtle.showturtle()** - показывает черепаху и др.

Параметр «**Style**» в функции **shape()** изменяет внешний вид исполнителя. Поддерживаются следующие стили:

- **arrow**
- **turtle**
- **circle**
- **square**
- **triangle**
- **classic**

Чтобы программа с модулем **turtle** на Python работала корректно, в самом конце программы всегда нужно прописывать две команды:

```
1. t.screen.exitonclick()
2. t.screen.mainloop()
```

С помощью команды **t.screen.exitonclick()** программа на Python реагирует на нажатие кнопки мыши после исполнения программы. Если пользователь нажмёт на левую кнопку мыши, пока курсор находится в окне для графики модуля **turtle**, то окно закроется.
С помощью команды **t.screen.mainloop()** останавливает выполнение программы.

Запустив программу, вы увидите окно для графики с «**черепашкой**» по центру (рис.1):

Конспект 4. Графический модуль turtle в Python

Учитель информатики Батракова Л.В.

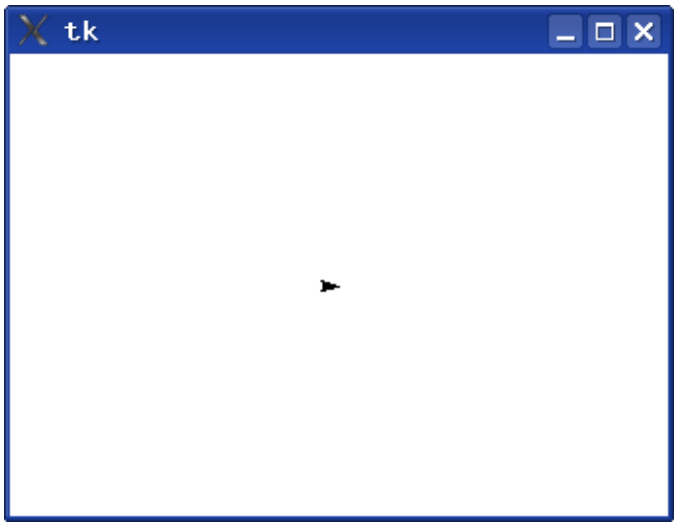


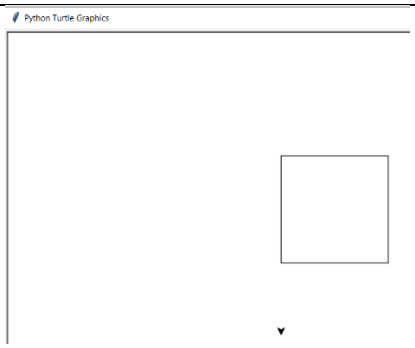
Рис.1.

Начало координат в окне для графики модуля **`turtle`** находится в центре окна. Положительное направление оси X определяется слева направо, положительное направление оси Y определяется снизу вверх, чем больше X, тем правее черепашка, чем больше Y, тем выше черепашка.
Рисунки на экране появляются с помощью перемещения «черепашки» в окне для графики модуля **`turtle`**, черепашка рисует линию.
Чтобы черепашка в окне для графики модуля **`turtle`** в Питоне двигалась вперёд, используется команда **`t.fd(x)`**, где x – количество пикселей, на которое сдвигается черепашка. Для движения назад используется команда **`t.bk(x)`.** Для передвижения черепашки в заданную точку использовать координаты **`t.goto(x, y)`**, где x и y – координаты точки, в которую должна переместиться черепашка.

Примеры программ:

Текст программы	Рисунок

пунктир.py - G:/Группа Игры на Python/Черепашка
File Edit Format Run Options Window Help
`from turtle import *`
`t = Turtle()`
`t.screen.setup(800, 800)`
`for i in range(20):`
`    t.fd(8)`
`    t.up()`
`    t.fd(8)`
`    t.down()`
`t.screen.exitonclick()`
`t.screen.mainloop()`	

gr12.py - F:/Питон/Питон_уроки Полякова/gr12.py (3.7.0)
File Edit Format Run Options Window Help
`import turtle  # Подключаем модуль turtle`
`turtle.reset()  # Приводим черепашку в начальное положение`
`turtle.down()  # Опускаем перо перо (начало рисования)`
`turtle.forward(150)  # Проползти 20 пикселей вперед`
`turtle.left(90)  # Поворот влево на 90 градусов`
`turtle.forward(150)  # Рисуем вторую сторону квадрата`
`turtle.left(90)`
`turtle.forward(150)  # Рисуем третью сторону квадрата`
`turtle.left(90)`
`turtle.forward(150)  # Рисуем четвертую сторону квадрата`
`turtle.up()  # Поднять перо (закончить рисовать)`
`turtle.forward(100)  # Отвести черепашку от рисунка в сторону`
`turtle._root.mainloop()  # Задержать окно на экране`
	

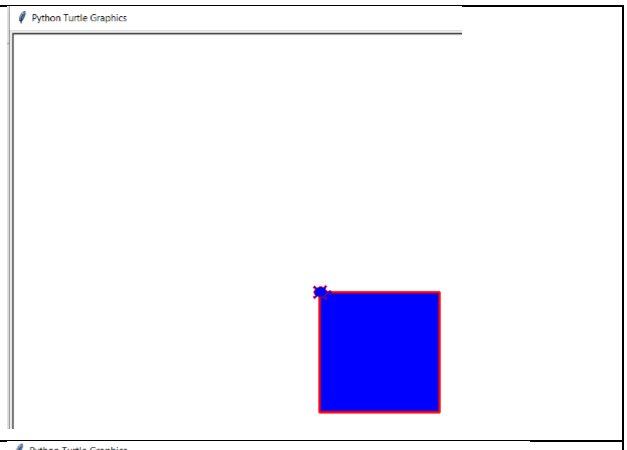
Конспект 4. Графический модуль turtle в Python

Учитель информатики Батракова Л.В.

gr2.py - F:\Питон\Питон_уроки Полякова\gr2.py (3.7.0)

File Edit Format Run Options Window Help

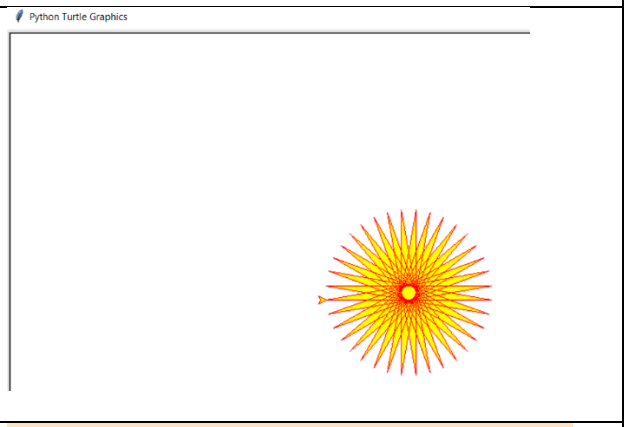
```
import turtle
from turtle import *
reset()
shape("turtle")
color("red", "blue")
pensize(3)
begin_fill()
for i in range(4):
    fd(150)
    rt(90)
end_fill()
```



gr3.py - F:\Питон\Питон_уроки Полякова\gr3.py (3.7.0)

File Edit Format Run Options Window Help

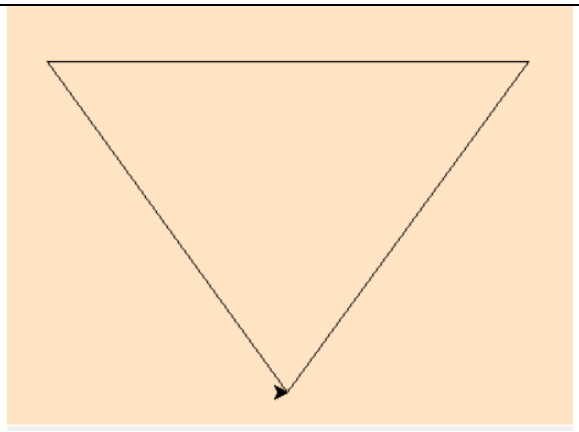
```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```



з10_треугольник.py - F:\Группа Игры на Python\Черепашка\з10_треугольник.py (3

File Edit Format Run Options Window Help

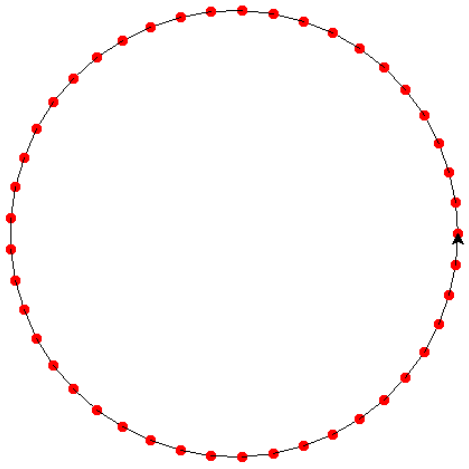
```
from turtle import *
iter=4 # Количество итераций
screensize(1200,1200,'Bisque')
reset() # сброс настроек черепашки и очистка окна
setworldcoordinates(-50,-50,1150,1150)
speed(1) # устанавливаем самую медленную скорость пера
# Процедура строит тр-к по координатам вершин
def tr(x1,y1,x2,y2,x3,y3):
    penup()
    goto(x1,y1)
    pendown()
    goto(x2,y2)
    goto(x3,y3)
    goto(x1,y1)

down()
tr(320,10,600,470,40,470)
```



окружность с точками.py - G:/Группа Игры на Python/Чер

File Edit Format Run Options Window Help

```
from turtle import *
t = Turtle()
t.screen.setup(800, 800)
def circ(d, r, rBig):
    for i in range(d):
        t.circle(rBig, 360 / d)
        t.dot(r, "red")
t.up()
t.goto(350, 0)
t.setheading(90)
t.down()
circ(45, 10, 200)
t.screen.exitonclick()
t.screen.mainloop()
```



Конспект 4. Графический модуль turtle в Python

Учитель информатики Батракова Л.В.

Рассмотрите, пожалуйста, рисунки и постарайтесь понять, что означает каждая команда.

Замечание: Оператор **`for i in range (4)`** означает, что цикл выполнится 4 раза.

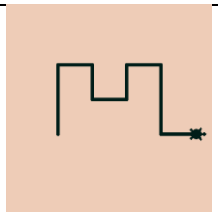
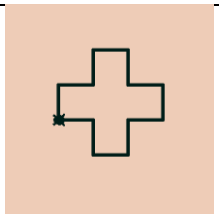
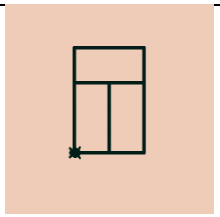
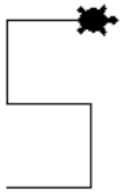
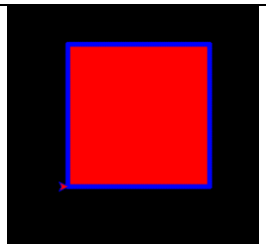
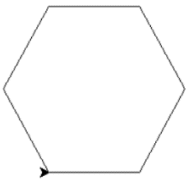
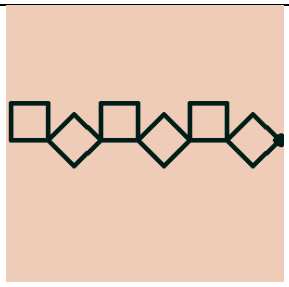
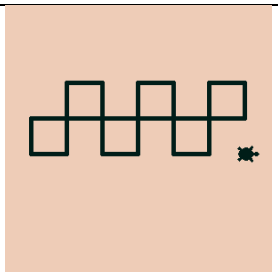
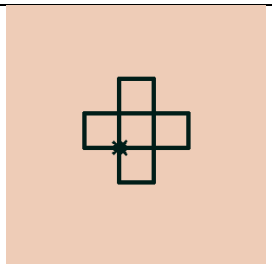
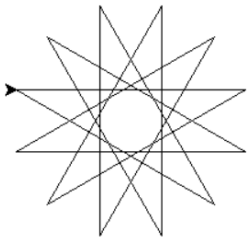
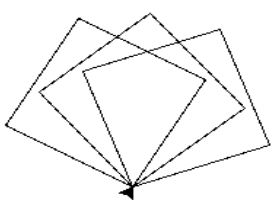
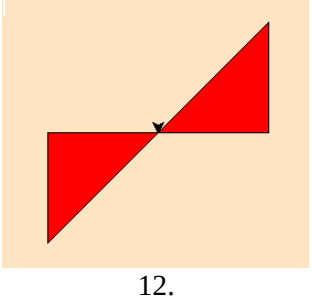
Ниже приведен список основных команд модуля **`turtle.`**

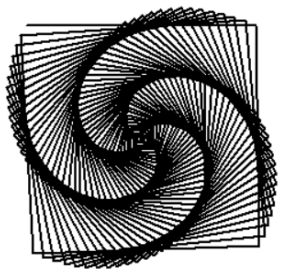
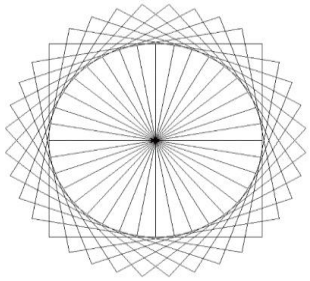
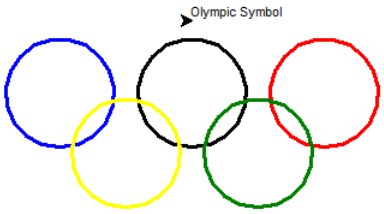
Список основных команд черепашки

Команды перемещения черепашки	
`forward(n)/fd(n)`	Проползти вперед n шагов (пикселей)
`backward(n)`	Проползти назад n шагов (пикселей)
`left(angle)`	Повернуться налево на angle градусов
`right(angle)`	Повернуться направо на angle градусов
`circle(r)`	Нарисовать окружность радиуса |r|, центр которой находится слева от черепашки, если r>0 и справа, если r<0
`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
`goto(x,y)`	Переместить черепашку в точку с координатами (x,y)
Команды рисования	
`down()`	Опустить перо. После этой команды черепашка начнет оставлять след при любом своем передвижении
`up()`	Поднять перо
`width(n)`	Установить ширину следа черепашки в n пикселей
`color(s)`	Установить цвет следа черепашки в s. s должно быть текстовой строкой, заключенной в кавычки, с названием цвета (по-английски), например: "red", "yellow", "green" и т.д.
`Bgcolor(s)`	Устанавливает цвет фона в s. s должно быть текстовой строкой, заключенной в кавычки, с названием цвета (по-английски), например: "red", "yellow", "green" и т.д.
`fill(f)`	Используется для рисования закрашенных областей. Начиная рисовать закрашенную область, дайте команду turtle.fill(1), а закончив рисование области — turtle.fill(0)
`stamp()`	После выполнения этой команды в окне для графики в месте, на котором была черепашка, останется рисунок этой черепашки**.**
`setheading(x)`	где x – угол поворота в градусах относительно начального положения

	черепашки при запуске программы
Прочие команды	
`reset()`	Возврат черепашки в исходное состояние: очищается экран, сбрасываются все параметры, черепашка устанавливается в начало координат, глядя вправо
`write(s)`	Вывести текстовую строку s в точке нахождения черепашки
`dot(r, color)`	Команда ставит точку, где r – радиус точки и color – цвет точки
`radians()`	Установить меру измерения углов (во всех командах черепашки) в радианы
`degrees()`	Установить меру измерения углов (во всех командах черепашки) в градусы. Этот режим включен по умолчанию
`tracer(f)`	Включить режим отладки (трассировки) программы черепашки, если значение f равно 1. Если значение f равно 0, отключить режим отладки. В режиме отладки черепашка перемещается медленнее и на экране нарисована сама черепашка. По умолчанию режим отладки включен

Задания:
Нарисуйте следующие картинки, используя команды черепашки:

<table>
<tr><td> 
1.</td><td> 
2.</td><td> 
3.</td></tr>
<tr><td> 
4.</td><td> 
5.</td><td> 
6.</td></tr>
<tr><td> 
7.</td><td> 
8.</td><td> 
9.</td></tr>
<tr><td> 
10.</td><td> 
11.</td><td> 
12.</td></tr>
</table>

		
13.	14.	15.

Вот еще несколько примеров:

1. **Программа рисует оси координат (смотри рис.2).**

```
# -*- coding: utf-8 -*-
  import turtle
#
turtle.reset()
turtle.tracer(0)
turtle.color('#0000ff')
#
turtle.write('0,0')
#
turtle.up()
x=-170
y=-120
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
x=130
y=100
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
x=0
y=-100
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.down()
x=0
y=100
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.up()
x=-150
y=0
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.down()
x=150
y=0
coords=str(x)+","+str(y)
```

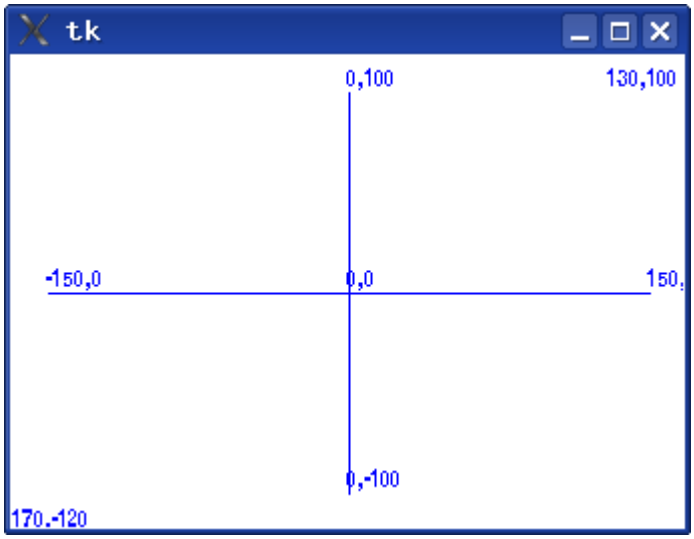


Рис.2

```
turtle.goto(x, y)
turtle.write(coords)
turtle.up()
#
turtle.mainloop()
```

2. **Программа рисует смайлик (рис.3)**

```
# -*- coding: utf-8 -*-
import turtle
from turtle import *
#
reset()
tracer(0)
width(2)
#
up()
x=0
y=-100
goto(x, y)
begin_fill()
color('#ffaa00')
down()
circle(100)
end_fill()
color('black')
circle(100)
up()
#
x=-45
y=50
goto(x, y)
down()
color('#0000aa')
begin_fill()
circle(7)
up()
end_fill()
#
x=45
y=50
goto(x, y)
down()
color('#0000aa')
begin_fill()
circle(7)
up()
end_fill()
#
x=-55
y=-50
goto(x, y)
right(45)
width(3)
down()
color('#aa0000')
```

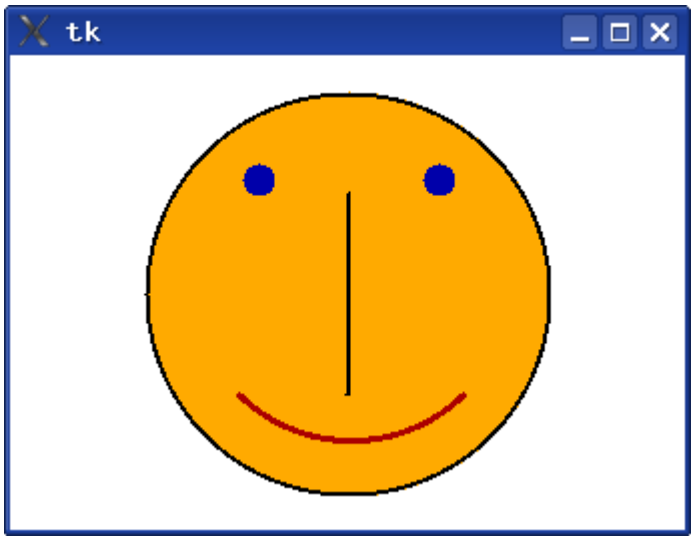


Рис. 3.

```
circle(80, 90)
up()
#
x=0
y=50
goto(x, y)
right(135)
width(3)
down()
color('#000000')
forward(100)
up()
#
turtle.mainloop()
```

Скопируйте и запустите следующие 3 программы. Может они вас вдохновят на создание своей интересной программы.

Пример 1. Полет черепахи:

```
from turtle import *
color("orange")
dot(10)
center = pos()
color("blue")
shape("turtle")
speed(0)
penup()
goto(200, 0)
pendown()
G = 800
v = Vec2D(0, 1)
t = 0
dt = 1
while t < 1100:
    goto(pos() + v * dt)
    setheading(towards(center))
    r = distance(center)
    acc = (-G / r ** 3) * pos()
    v += acc * dt
    t += dt
mainloop()
```

Пример 2 Радиоактивный знак:

```
from turtle import *

def square(length):
    for i in range(4):
        forward(length)
        left(90)

def sector(radius, angle):
    forward(radius)
    left(90)
    circle(radius, angle)
```

```
    left(90)
    forward(radius)
    left(180-angle)

def move(x, y):
    up()
    forward(x)
    left(90)
    forward(y)
    right(90)
    down()

def radioactive(radius1, radius2, side, angle=60, outlinecol="black",
fillcol="yellow"):
    color(outlinecol)
    move(-(side/2), -(side/2))

    begin_fill()
    square(side)
    color(fillcol)
    end_fill()
    move((side/2), (side/2))
    color(outlinecol)
    right(90 + angle/2)

    for i in range(3):
        begin_fill()
        sector(radius1,angle)
        left(120)
        color(outlinecol)
        end_fill()

    up()
    forward(radius2)
    left(90)
    down()

    color(fillcol)
    begin_fill()
    circle(radius2)
    color(outlinecol)
    end_fill()

    up()
    left(90)
    forward(radius2)
    width(1)

reset()
width(5)
speed(1)
radioactive(160, 36, 400)
mainloop()
```

Пример 3 Пример разработчиков:

```
from turtle import *
def switchpen():
        if isdown():
            pu()
        else:
            pd()

def demo2():
        """Demo of some new features."""
        speed(1)
        st()
        pensize(3)
        setheading(towards(0, 0))
        radius = distance(0, 0)/2.0
        rt(90)
        for _ in range(18):
            switchpen()
            circle(radius, 10)
        write("wait a moment...")
        while undobufferentries():
            undo()
        reset()
        lt(90)
        colormode(255)
        laenge = 10
        pencolor("green")
        pensize(3)
        lt(180)
        for i in range(-2, 16):
            if i > 0:
                begin_fill()
                fillcolor(255-15*i, 0, 15*i)
            for _ in range(3):
                fd(laenge)
                lt(120)
            end_fill()
            laenge += 10
            lt(15)
            speed((speed()+1)%12)
        #end_fill()

        lt(120)
        pu()
        fd(70)
        rt(30)
        pd()
        color("red","yellow")
        speed(0)
        begin_fill()
        for _ in range(4):
            circle(50, 90)
            rt(90)
```

```
            fd(30)
            rt(90)
        end_fill()
        lt(90)
        pu()
        fd(30)
        pd()
        shape("turtle")

        tri = getturtle()
        tri.resizemode("auto")
        turtle = Turtle()
        turtle.resizemode("auto")
        turtle.shape("turtle")
        turtle.reset()
        turtle.left(90)
        turtle.speed(0)
        turtle.up()
        turtle.goto(280, 40)
        turtle.lt(30)
        turtle.down()
        turtle.speed(6)
        turtle.color("blue","orange")
        turtle.pensize(2)
        tri.speed(6)
        setheading(towards(turtle))
        count = 1
        while tri.distance(turtle) > 4:
            turtle.fd(3.5)
            turtle.lt(0.6)
            tri.setheading(tri.towards(turtle))
            tri.fd(4)
            if count % 20 == 0:
                turtle.stamp()
                tri.stamp()
                switchpen()
            count += 1
        tri.write("CAUGHT! ", font=("Arial", 16, "bold"), align="right")
        tri.pencolor("black")
        tri.pencolor("red")

        def baba(xdummy, ydummy):
            clearscreen()
            bye()

        while undobufferentries():
            tri.undo()
            turtle.undo()
        tri.fd(50)
        tri.write("  Click me!", font = ("Courier", 12, "bold") )
        tri.onclick(baba, 1)

demo2()
```

Конспект 4. Графический модуль turtle в Python

Учитель информатики Батракова Л.В.

Источники информации:

1. http://server.179.ru/tasks/python/2017b1/pgm12.5_Turtle.html
2. http://itrobo.ru/programmirovanie/python/grafika-turtle-cherepashka-v-piton.html
3. http://31415.ru/blog/python/
4. https://ru.wikiversity.org/wiki/%D0%9A%D1%83%D1%80%D1%81_%D0%BF%D0%BE_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B5_Tkinter_