Модуль графика Turtle в Python Питон.

Мы разберём **графику в питон Python** с помощью модуля **Turtle черепашка**.

Turtle это модуль для **Питон**, позволяющий создавать графические объекты, рисунки в специальном окне. Модуль **Turtle** можно использовать для создания игр на **Питоне**.

Чтобы начать работу с модулем, нужно ввести объект Turtle()

```
t = Turtle()
```

Далее нужно ввести окно для графических объектов в **Питон**, в котором мы будем проводить все действия. Чтобы задать окно нужного размера в модуле **turtle** используется команда t.screen.setup(x, y), где x и y – ширина и высота окна в пикселях. В этом уроке мы введём окно размером 800×800 пикселей.

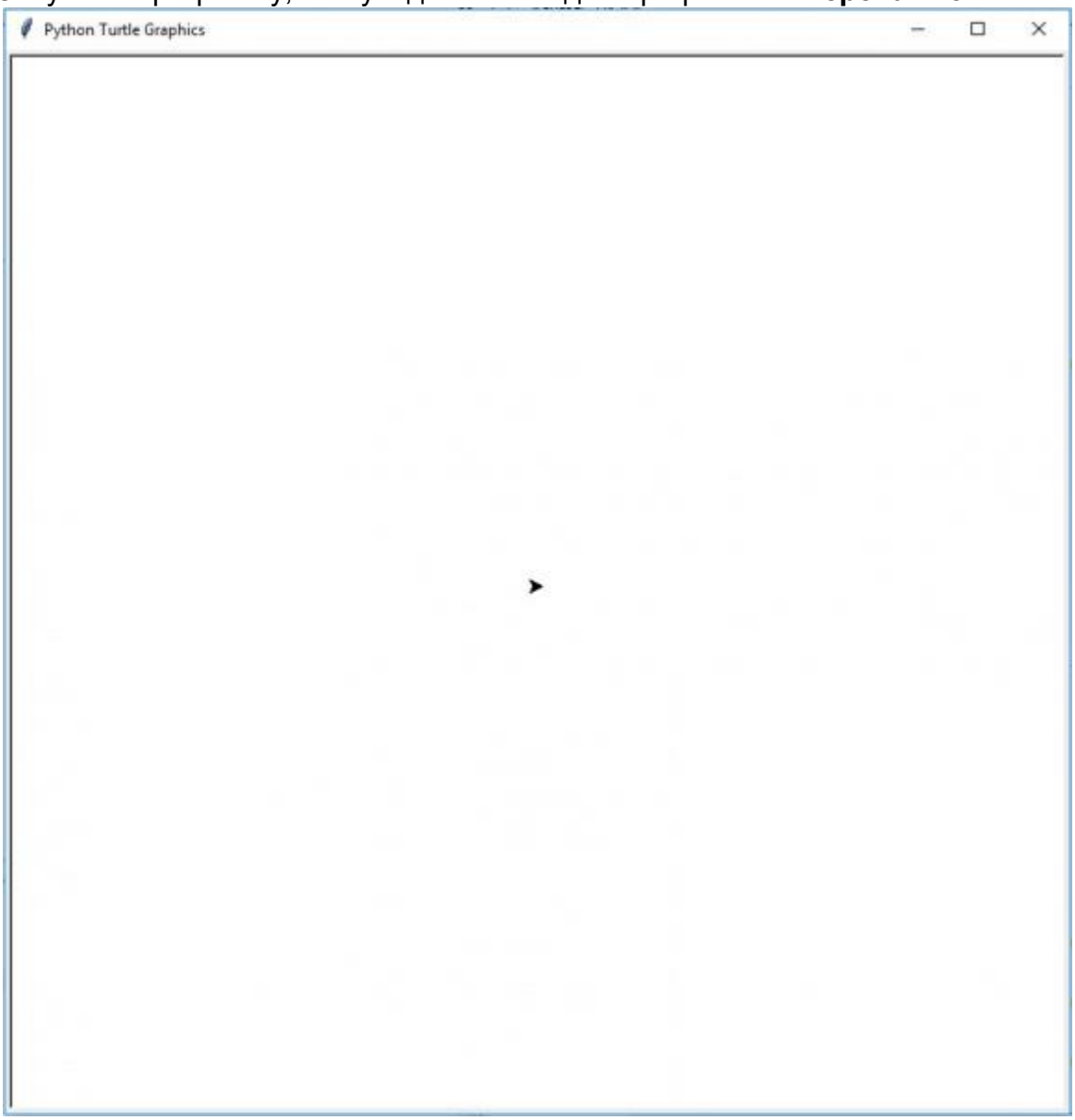
```
t.screen.setup(800, 800)
```

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

```
t.screen.exitonclick()
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. t.screen.mainloop() останавливает выполнение программы.

Запустив программу, вы увидите окно для графики с «**черепашкой**» по центру.

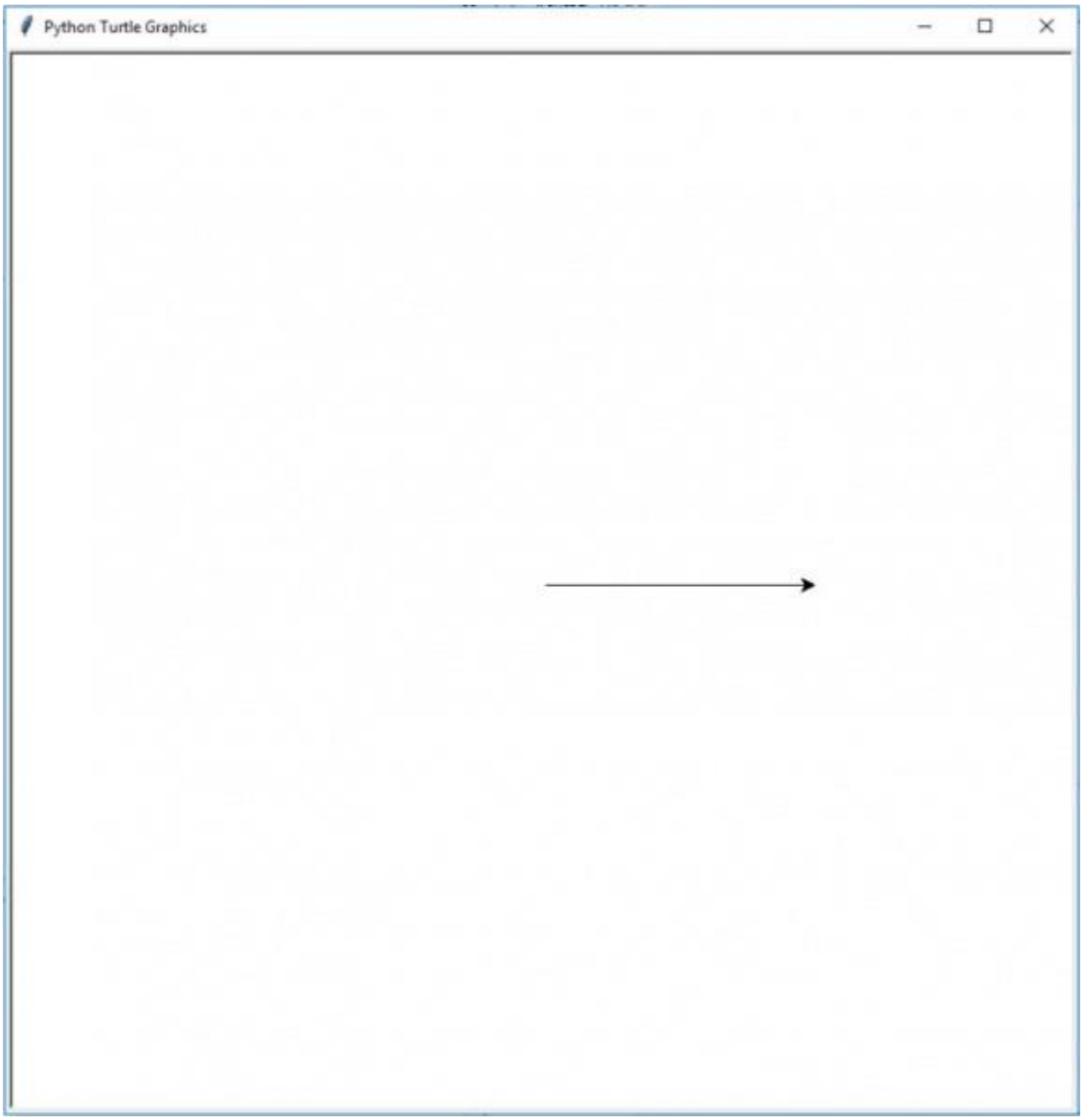


Начало координат в окне для графики модуля **turtle** находится в центре окна. Положительное направление оси X определяется слева направо, положительное направление оси Y определяется снизу вверх, чем больше X, тем правее черепашка, чем больше Y, тем выше черепашка.

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 заданую точку использовать координаты t.goto(x, y), где x и y – координаты точки, в которую должна переместиться черепашка. Пример программы на python рисование линии с помощью черепашки

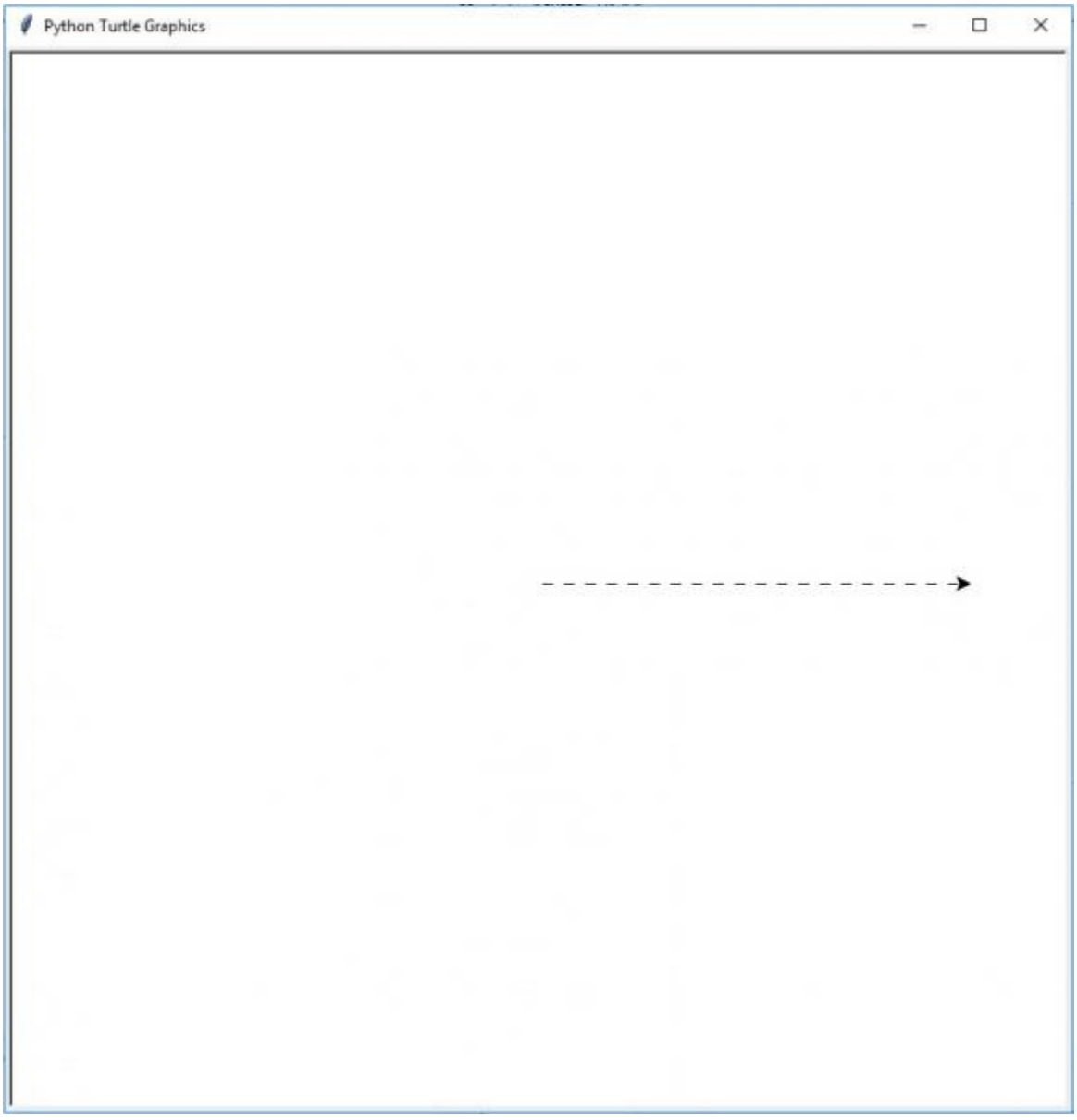
```
from turtle import *
t = Turtle()
t.screen.setup(800, 800)
t.fd(200)
t.screen.exitonclick()
t.screen.mainloop()
```



При движении **черепашка** из **модуля turtle в Питоне** рисует линию на своей траектории. Чтобы **черепашка** двигалась без отображения линии, нужно использовать команду t.up(). Чтобы **черепашка** опять рисовала линию по своей траектории, используйте команду t.down().

Пример. Черепашка рисует пунктирную линию.

```
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()
```

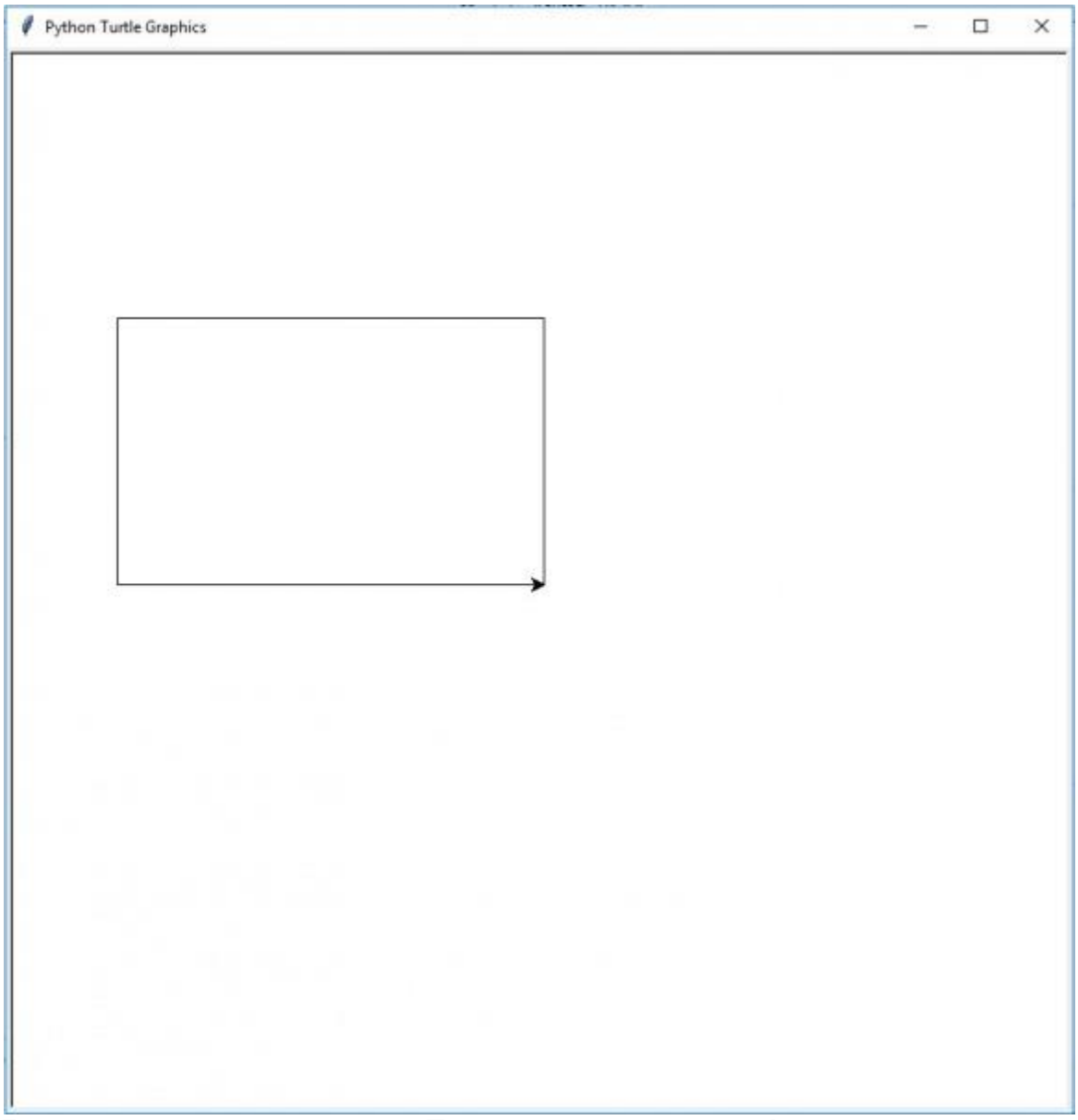


В модуле turtle в Python черепашка может рисовать не только прямые линии. **Черепашку** можно поворачивать, используя команды t.left(x) (поворачивает черепашку влево) и t.right(x) (поворот вправо), где x – угол поворота в градусах.

Пример. Черепашка рисует прямоугольник.

```
from turtle import *
t = Turtle()
t.screen.setup(800, 800)
def rectangle(w, h):
    for i in range(2):
        t.left(90)
        t.fd(h)
```

```
        t.left(90)
        t.fd(w)
    rectangle(320, 200)
    t.screen.exitonclick()
t.screen.mainloop()
```

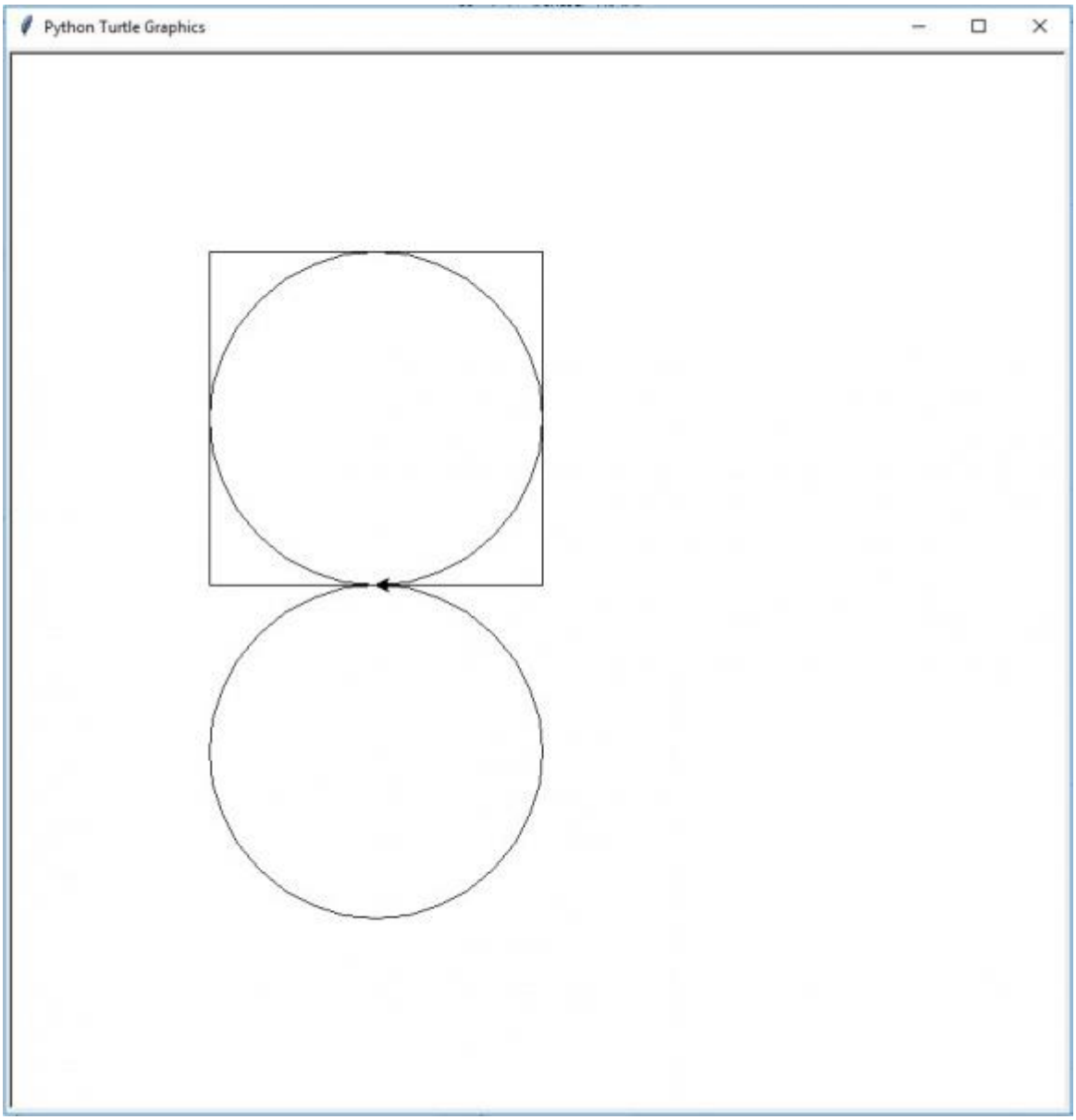


Для поворота **черепашки в Питоне в модуле turtle** в определённую сторону используется команда **модуля turtle** t.setheading(x), где x – угол поворота в градусах относительно начального положения **черепашки** при запуске программы. Если вы введёте в эту команду значение параметра 90, то **черепашка** повернётся вверх, 180 – повернётся налево, 270 – повернётся вниз, 360 или 0 – повернётся направо.

С помощью **черепашки в модуле turtle** можно рисовать окружности. Для этого используется команда t.circle(r, ϕ), где r – радиус круга, ϕ – часть окружности, которую мы рисуем, в градусах. При значении ϕ в 180 градусов **черепашка в Питоне** нарисует полуокружность, при 360 градусах нарисует полную окружность. Пример программы на Python в которой с помощью черепашки рисуется квадрат и вписанная в него окружность

```
from turtle import *
t = Turtle()
t.screen.setup(800, 800)
def sq_cr(side):
    for i in range(4):
        t.left(90)
```

```
        t.fd(side)
        t.bk(side / 2)
        t.circle(side / 2, 360)
        t.left(180)
        t.circle(side / 2, 360)
    sq_cr(250)
    t.screen.exitonclick()
    t.screen.mainloop()
```

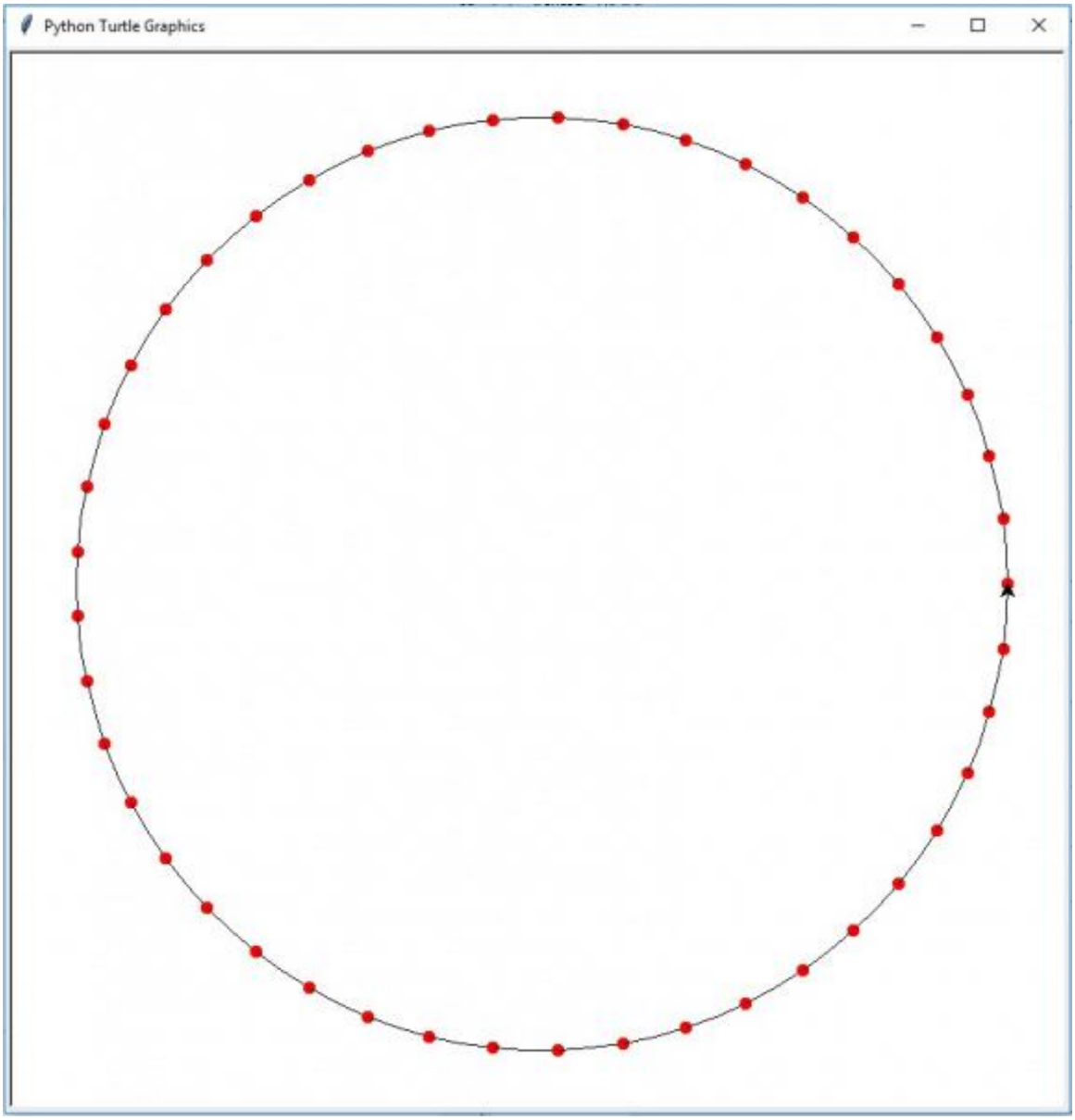


Модуль turtle в Питоне позволяет рисовать точки. Для этого используется команда t.dot(r, color), где r – радиус точки в пикселях, color – цвет, которым будет рисоваться точка.

Пример. Программа рисует круг с заданным количеством точек на нём. У точек и у круга можно изменять радиус.

```
    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, 350)
t.screen.exitonclick()
t.screen.mainloop()
```

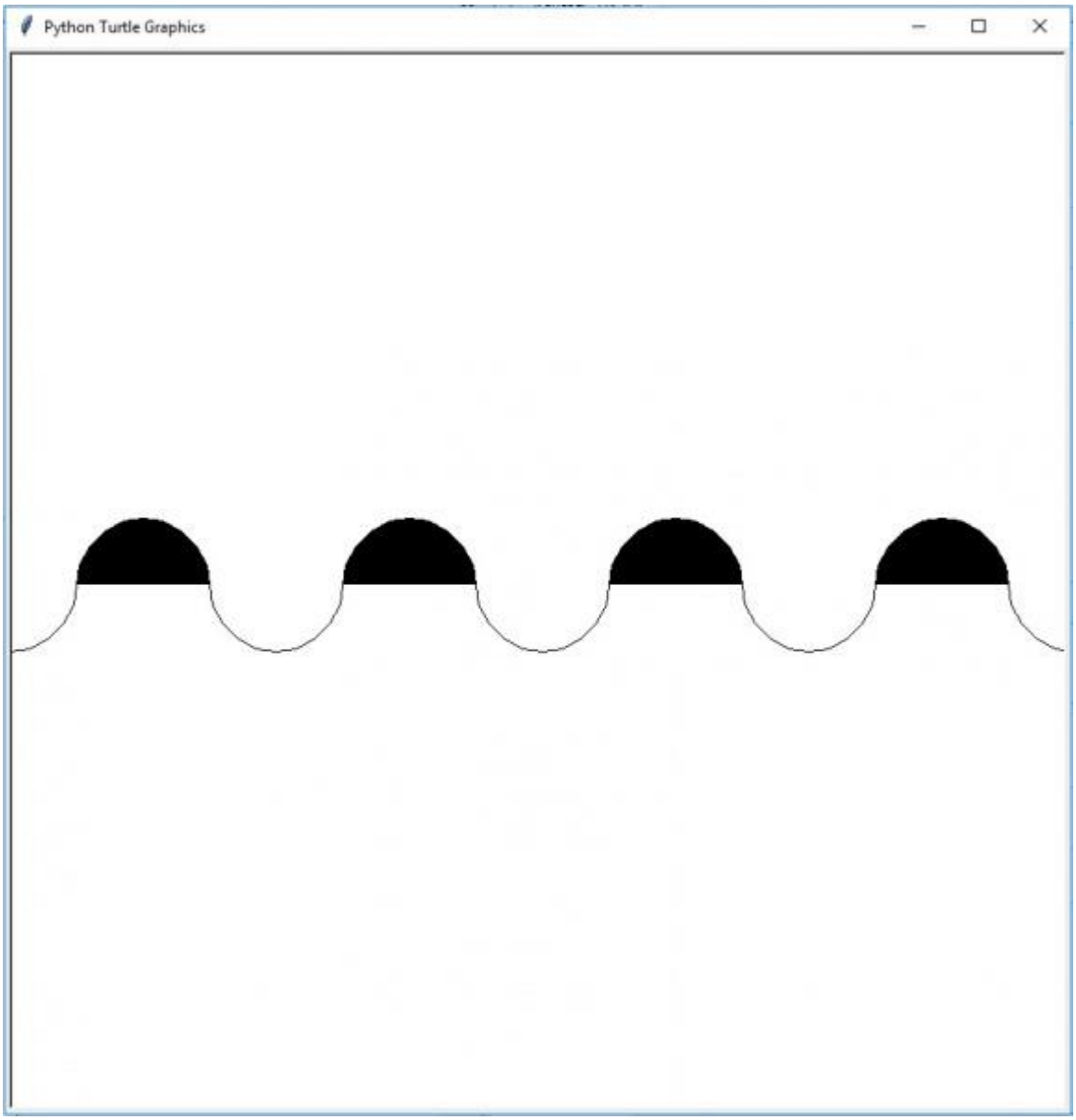


В **модуле turtle в Python** есть возможность рисовать закрашенные фигуры. Для закрашивания фигуры в модуле turtle используется команда t.begin_fill(). Все нарисованные фигуры будут заливаться цветом черепашки. Если вы хотите поставить другой цвет заливки, но оставить тот же цвет черепашки, пропишите команду t.fillcolor("цвет"), в кавычках пишется цвет, которым нужно заливать фигуры. Чтобы черепашка перестала заливать фигуры, нужно написать t.end_fill().

Пример. Программа рисует волны, закрашивается только верхняя часть этих волн.

```
from turtle import *
    t = Turtle()
    t.screen.setup(800, 800)
    t.up()
    t.goto(-450, 0)
    t.down()
    t.setheading(270)
    for i in range(5):
        t.circle(50, 180)
        t.begin_fill()
        t.circle(-50, 180)
        t.end_fill()
```

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

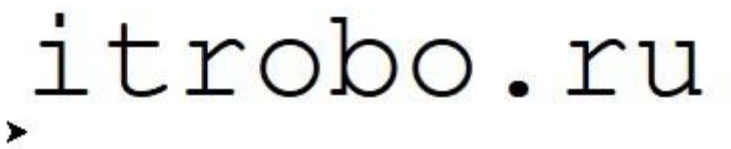


В окне для графики **модуля turtle Питон** возможно рисовать текст. Для отображения текста в модуле turtle используется команда t.write()
t.write(text, move, align, font = (fontname, fontsize, fontstyle))
В параметр text команды t.write() пишется текст, который будет отображаться в окне для графики **turtle Python**. Текст пишется в кавычках.

Параметр move принимает только логические значения (True, False), этот параметр отвечает за то, появится ли анимация **черепашки**, после отображения текста. В анимации **черепашка** подчёркивает написанный текст.

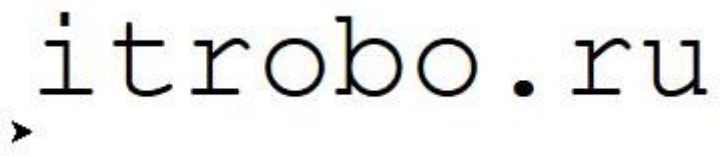
```
move = True
```


```
move = False
```

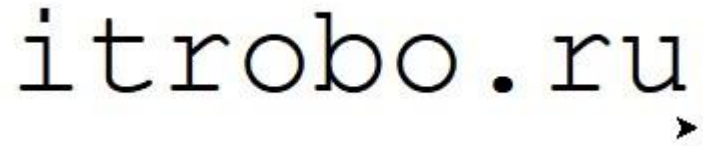


align принимает значения "left", "right", "center", этот параметр отвечает за положение

текста относительно **черепашки**. Все значения пишутся в кавычках. Все варианты параметра align показаны на картинках ниже с параметром move = False. Для сравнения посмотрите на положение черепашки относительно текста в каждом варианте.
align = “left”



align = “right”



align = ”center”

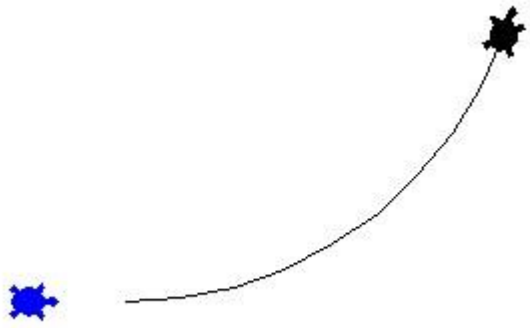
Параметр font **модуля turtle в Питоне** принимает значения fontname, fontsize, fontstyle. В параметре fontname задается название шрифта в кавычках, fontsize отвечает за размер шрифта, fontstyle отвечает за стиль текста. Стиль текста пишется в кавычках. Параметр fontstyle имеет значения "normal" для обычного текста, "bold" полужирного текста, "italic" курсивного текста, "bold italic" полужирного курсивного текста.

В **модуле turtle в Питоне** можно изменять саму **черепашку**, её размер и цвет. Чтобы изменить форму **черепашки**, используйте команду t.shape(""), в кавычках указывается форма черепашки в кавычках. В модуле **turtle** существуют такие формы черепашки, как "arrow", "circle", "square", "triangle", "turtle", "classic". Например, если вы хотите **черепашку** в форме квадрата, напишите команду t.shape(“square”). Размер **черепашки** можно изменить с помощью команды t.shapesize(n), где n – размер черепашки.

Если вы хотите, чтобы **черепашка** не показывалась на экране, используйте команду t.hideturtle(). Чтобы **черепашка** опять показывалась, используйте команду t.showturtle().

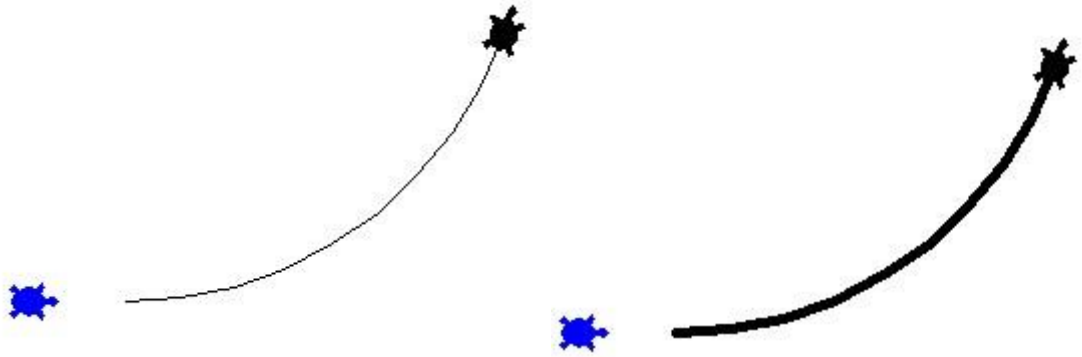
Черепашка в модуле turtle в Питоне может оставлять след. Для этого используйте t.stamp(). После выполнения этой команды в окне для графики в месте, на котором была **черепашка**, останется рисунок этой **черепашки**.
Пример программы на python в которой черепашка оставляет след синего цвета, затем рисует дугу.

```
from turtle import *
t = Turtle()
t.screen.setup(800, 800)
t.shape("turtle")
t.color("blue")
t.stamp()
t.color("black")
t.up()
t.fd(50)
t.down()
t.circle(200, 70)
t.screen.exitonclick()
t.screen.mainloop()
```



Ширину линии, рисуемой **черепашкой** из **модуля turtle в Python**, можно изменить. Для этого используется команда t.pensize(n), n – ширина линии. По умолчанию ширина линии равна 1.

Ниже показано сравнение линий с шириной 1 и шириной 5.

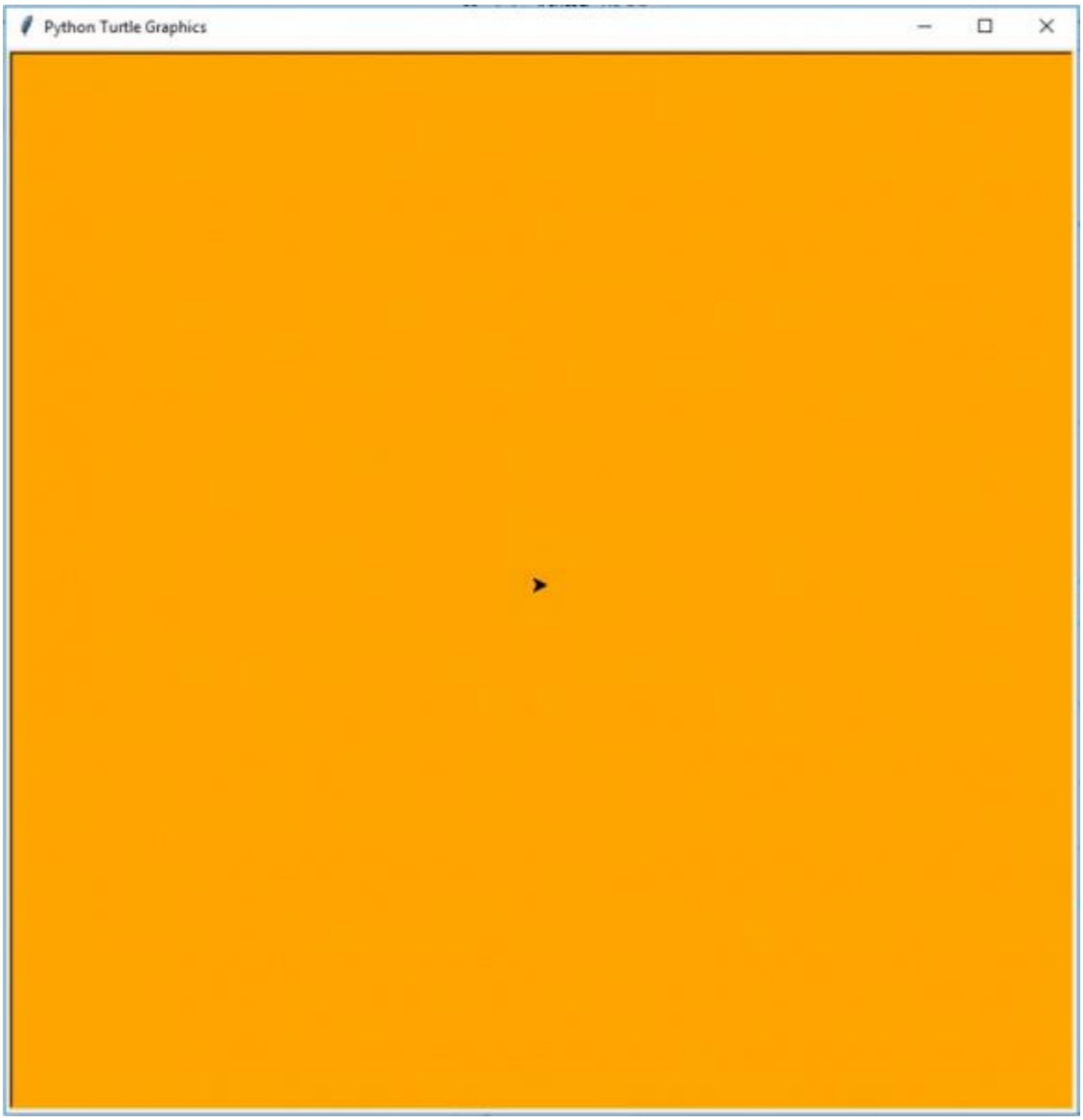


Если с помощью **модуля turtle в Питоне** вы создали большой рисунок, то **черепашка** будет рисовать его слишком долго. Чтобы ускорить **черепашку**, используйте команду t.speed(n), n – скорость черепашки. n должно быть целым числом больше или равным нулю. Ноль – самая большая скорость, доступная для использования. Менять можно не только **черепашку из модуля turtle** и линию, но и некоторые свойства окна для графики в **Python**.

Чтобы изменить цвет фона окна для графики, используется команда t.screen.bgcolor(""), в кавычках пишется цвет заднего фона.

Пример. Задаём оранжевый цвет заднего фона.

```
from turtle import *
    t = Turtle()
    t.screen.setup(800, 800)
    t.screen.bgcolor("orange")
    t.screen.exitonclick()
    t.screen.mainloop()
```



Чтобы очистить окно от всего, что было нарисовано **черепашкой**, используйте команду t.clear(). Команда t.reset() не только очищает экран от рисунков, но и перемещает **черепашку** в центр.

Команды перемещения черепашки

forward(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.

fill(f) Используется для рисования закрашенных областей. Начиная рисовать закрашенную область, дайте команду t.fill(1), а закончив рисование области — t.fill(0).

Прочие команды

reset() Возврат черепашки в исходное состояние: очищается экран, сбрасываются все параметры, черепашка устанавливается в начало координат, глядя вправо.

clear() Очистка экрана.

write(s) Вывести текстовую строку s в точке нахождения черепашки.

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.

Упражнения для самостоятельной работы

1. Нарисуйте на экране равносторонний треугольник. Нарисуйте желтый равносторонний треугольник. Нарисуйте закрашенный красный равносторонний треугольник.
2. Нарисуйте на экране квадрат с диагоналями. Нарисуйте только диагонали квадрата (два пересекающихся отрезка). Нарисуйте квадрат, стороны которого не параллельны осям координат.
3. Нарисуйте две касающиеся окружности. Нарисуйте две пересекающиеся окружности.
4. Нарисуйте сами какую-нибудь цветную картинку (дом, дерево, рожицу, компьютер, ...).