**Функции**

**setheading(угол), seth(угол) - курс**

Встать головой (угол)



|  |  |  |
| --- | --- | --- |
| [**Угол**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonTurtleBeginFunction?sortcol=0&table=1&up=0#sorted_table) | [**Куда (рус)**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonTurtleBeginFunction?sortcol=1&table=1&up=0#sorted_table) | [**Куда(eng)**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonTurtleBeginFunction?sortcol=2&table=1&up=0#sorted_table) |
| 0. | восток | east |
| 90 | север | north |
| 180 | запад | west |
| 270 | юг | south |

**sleep(секунд) - жди**

Функция **sleep**(секунд) - программа ничего не делает это время (секунд).

Эта функция находится не в библиотеке turtle, а в библиотеке time. Нам из всей библиотеки time нужна только одна функция sleep. Чтобы взять из библиотеки только одну функцию, мы пишем:

from time import sleep # из библиотеки time берем только функцию sleep

sleep(5) # программа ничего не делает 5 секунд

Напишем программу полностью

# -\*- coding: utf-8 -\*-

# чтобы можно было писать русские буквы и иероглифы

import turtle # познакомили программу с всей библиотекой turtle (черепаха)

from time import sleep # познакомили программу с функцией sleep из библиотеки time

t = turtle.Turtle() # сделали черепаху, назвали черепаху t

t.shape("turtle") # как черепаха выглядит

t.forward(75) # вперед 75

sleep(3) # жди 3 секунды

t.forward(75) # вперед 75

turtle.done() # чтобы окно не закрывалось, на repl.it не нужно

**Новая функция**

Можно научить черепаху новой команде.

Для этого надо придумать имя новой функции и записать в нее известные функции.

DEA! Пишем новую функцию **после** того, как сделали черепаху.

# -\*- coding: utf-8 -\*-

# чтобы можно было писать русские буквы и иероглифы

import turtle # познакомили программу с пакетом turtle (черепаха)

from time import sleep # короткая форма import, можно дальше писать sleep

# это новая функция

# Она называется leaf (лист дерева)

# def пишется с начала строки без пробелов и <Tab>

# рисовать лист будет черепаха t,

# поэтому пишем функцию там, где уже знаем о черепахе (после того, как ее сделали)

def leaf ():

# здесь пишем код для рисования листа

# все строки должны начинаться с <Tab>

t.color("green")

t.begin\_fill()

t.forward(150)

sleep(1)

t.right(45)

sleep(1)

t.forward(150)

sleep(1)

t.right(135)

sleep(1)

t.forward(150)

sleep(1)

t.right(45)

sleep(1)

t.forward(150)

sleep(1)

t.end\_fill()

# Здесь новая команда leaf закончилась.

# Закончили ставить <Tab>

# Тут место для выполнения старых и новых команд.

# выполняем команды. Пишем БЕЗ пробелов и <Tab>

t = turtle.Turtle() # сделали черепаху, назвали черепаху t

t.shape("turtle") # как черепаха выглядит

t.width(3)

leaf() # вызвали (использовали) функцию leaf (выполнили все ее команды)

t.seth(90)

leaf() # вызвали функцию leaf (выполнили все ее команды)

turtle.done() # чтобы окно не закрывалось, на repl.it не нужно

Русский и английский: вызвать (call), использовать (use) функцию (function)

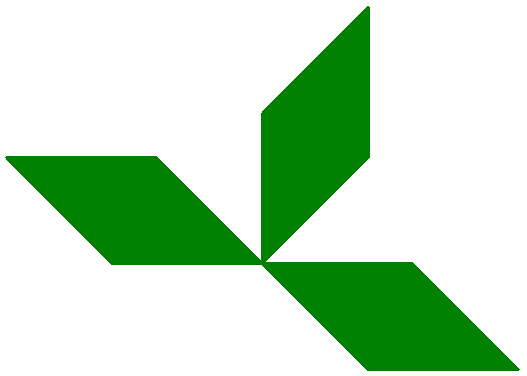
**Задача 1**

Записать этот код в файл **fun1.py** (Новый урок в новой директории!)

Запустить и посмотреть, что нарисует черепаха.

**Задача 2 - трилистник**

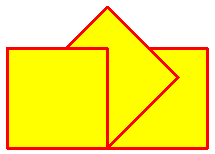
Убрать из функции **leaf()** все sleep.

Функцией leaf нарисовать: 

**Задача 3 - квадрат**

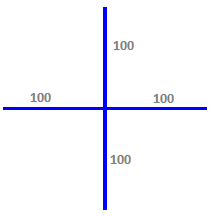
Написать функцию sq() (square - квадрат), которая рисует желтый квадрат с красной линией.

Вызвать функцию 3 раза так, чтобы нарисовать:



**Задача 4 - крест**

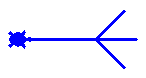
Написать функцию cross() (cross - крест), которая рисует 1 крест.



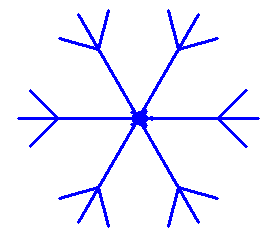
Придумать узор из крестов и написать программу для этого узора.

**Задача 5 - снежинка**

Написать функцию snowline() (snow - снег), которая рисует 1 часть снежинки.



Потом нарисовать такую снежинку.



**Функция в функции**

Новые функции можно писать в других новых функциях.

# -\*- coding: utf-8 -\*-

# чтобы можно было писать русские буквы и иероглифы

import turtle # познакомили программу с пакетом turtle (черепаха)

from time import sleep # короткая форма import, можно дальше писать sleep

# это новая функция

# Она называется sq (квадрат)

# def пишется с начала строки без пробелов и <Tab>

# рисовать лист будет черепаха t,

# поэтому пишем функцию там, где уже знаем о черепахе (после того, как ее сделали)

def sq ():

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

# Здесь новая команда sq закончилась.

# Началась другая новая функция sq2 (два квадрата)

def sq2():

t.color("red")

sq() # это наша функция sq

t.back(50)

t.left(45)

t.color("blue")

sq() # это наша функция sq

# Здесь новая команда sq2 закончилась.

# Тут место для выполнения старых и новых команд.

# выполняем команды. Пишем БЕЗ пробелов и <Tab>

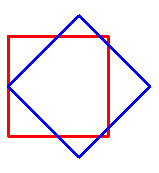
t = turtle.Turtle() # сделали черепаху, назвали черепаху t

t.shape("turtle") # как черепаха выглядит

t.width(3)

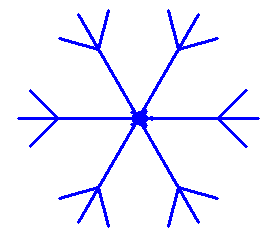
sq2()

turtle.done() # чтобы окно не закрывалось, на repl.it не нужно



**Задача 6 - снежинка функцией**

Напишите **фукцию** snowflake() для рисования снежинки.

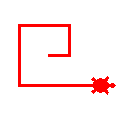


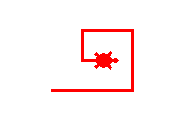
**Задача 7 - горы**

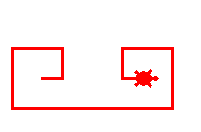
Напишите фукнцию peak(), которая рисует 1 гору.   

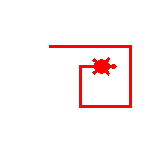

Напишите функцию gory(), которая рисует 3 горы.  

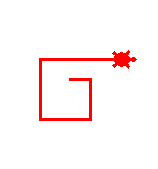

**Задача 8 - Орнамент**

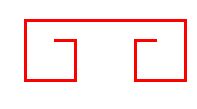
1. Напишите функцию спиралПраво(), которая рисует такую спираль   


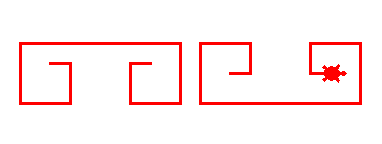
2. Напишите функцию спиралЛево(), которая рисует такую спираль   


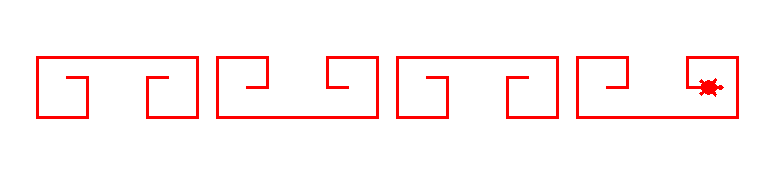
3. Напишите функцию узор1(), которая рисует   


4. Напишите функцию спиральВнизПраво(), которая рисует   


5. Напишите функцию спиральВнизЛево(), которая рисует   


6. Напишите функцию узор2(), которая рисует   


7. Напишите функцию узор3(), которая рисует   


8. Напишите функцию узор4(), которая рисует   


**Параметры функции (аргументы функции)**

Функциям черепахи можно дать цвет, угол, размер линии. Это **параметры** (или **аргументы** ).

Для новых функций тоже можно указать параметры.

Пример:

# -\*- coding: utf-8 -\*-

# чтобы можно было писать русские буквы и иероглифы

import turtle # познакомили программу с пакетом turtle (черепаха)

from time import sleep # короткая форма import, можно дальше писать sleep

# параметр col запомнит цвет квадрата

# и задаст его в функции

def sq (col):

t.color(col)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t = turtle.Turtle()

t.shape("turtle")

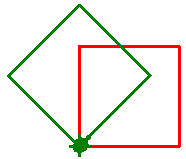
t.width(3)

sq("red") # передаем в функцию sq параметр "red"

t.left(45)

sq("green") # передаем в функцию sq параметр "green"

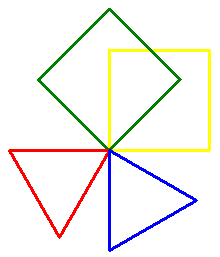
turtle.done()



**Задача 9 - цветные фигуры**

Дописать в программу функцию для треугольника.  
**Цвет треугольника - параметр** .

Нарисовать 2 квадрата: желтый и зеленый. Нарисовать 2 треугольника: красный и синий.



**Несколько параметров. Параметры-числа.**

Много параметров пишутся через запятую (,)

Каждый параметр имеет свое имя.

В параметры можно передавать числа.

Напишем функцию **rect(col, size)**, которая рисует прямоугольник. У него линии всегда красные и одна сторона в 3 раза больше другой.

В функцию передадим цвет внутри прямоугольника **col** и размер большой стороны **size** .

Размер маленькой стороны посчитаем size/3 .

С параметрами можно делать разные математические действия: +, -, \* (умножить), / (разделить), // (взять целую часть от деления)

# -\*- coding: utf-8 -\*-

import turtle

from time import sleep

# Это прямоугольник (rectangle).

# Одна сторона у него в 3 раза больше другой.

# Линии красные.

# 2 параметра: первый col (цвет внутри) , второй size (размер)

def rect (col, size):

t.color("red", col) # "red" - цвет линии, col - цвет внутри

t.begin\_fill()

t.forward(size)

t.left(90)

t.forward(size/3)

t.left(90)

t.forward(size)

t.left(90)

t.forward(size/3)

t.left(90)

t.end\_fill()

t = turtle.Turtle()

t.shape("turtle")

t.width(3)

t.speed(0)

rect("yellow", 300)

t.left(90)

rect("violet", 120)

t.hideturtle()

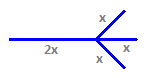
turtle.done()



**Задача 10 - луч снежинки**

У луча снежинки ветки в 3 раза меньше, чем длина луча.

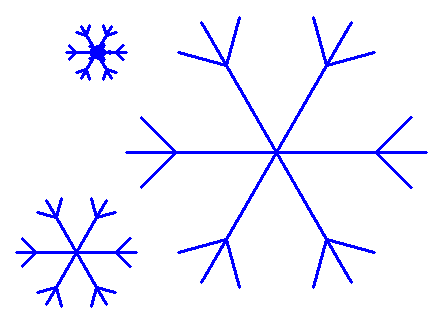
Написать функцию рисования 1 луча с параметром длина ветки.  
snowline(x)



**Задача 11 - снежинка**

Написать функцию рисования снежинки с параметром размер снежинки.  
snowflake(size)

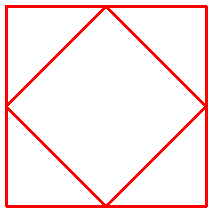
Нарисовать несколько **разных** снежинок.



**Задача 12 - квадрат в квадрате**

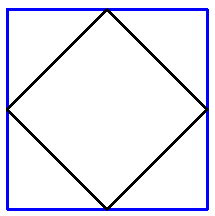
Написать функцию с параметром size (размер) для рисования такой картинки.  
**sqin(size)**

DEA! Чтобы посчитать корень из числа 2, нужно написать в начале программы **import math**, потом в коде **math.sqrt**(2)



**Задача 13 - квадрат в квадрате разных цветов**

Добавьте в предыдущую функцию еще два параметра - цвет одного и цвет другого квадрата.  
**sqin(size, color1, color2)**



**Действия с параметрами**

**goto(x,y), pos(), write(text)**

Черепаха двигается в координатной плоскости (x,y) и каждая точка имеет координату х и у.

* **goto(x,y)** - передвигает черепаху в точку (x,y). Если перо опущено, черепаха при движении рисует.  
  Начинает черепаха в точке (0,0).
* **write(text)** - пишет текст в том месте, где стоит черепаха.
* **pos()** - отдает координаты (x,y) точки, где стоит черепаха (как функции отдают значения - в следующем уроке).

Нарисуем линию от точки с координатами (x1,0) до точки с координатами (x2,0).  
Напишем для этого новую функцию **line(x1,x2)**.

# -\*- coding: utf-8 -\*-

import turtle

from time import sleep

import math

# функция пишет х и у координаты черепахи большими буквами

# я написала вам эту функцию. Скопируйте ее и пользуйтесь ей.

def xy():

x,y = t.pos()

x = int(x)

y = int(y)

t.write((x,y), font=("Arial", 14, "normal"))

# проводит линию от (x1,0) до (x2,0)

def line(x1, x2):

t.penup()

t.goto(x1,0)

xy()

t.color("blue")

t.pendown()

t.goto(x2,0)

xy()

t = turtle.Turtle()

t.shape("turtle")

t.width(3)

line(-200, 200)

t.hideturtle()

turtle.done()

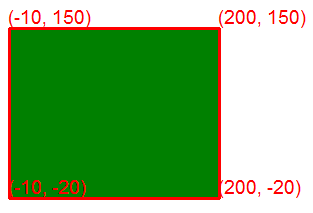
http://acm.mipt.ru/twiki/pub/Cintro/PythonTurtleBeginFunction/func_e13.png

**Задача 14 - прямоугольник по 2 точкам**

В прямоугольнике с параллельными осям сторонами даны две вершины (x1,y1) и (x2, y2), где x1≠x2, y1≠y2.

Напишите функцию **rect(x1, y1, x2, y2, rect\_color, text\_color)**.  
x1, y1, x2, y1 - координаты противоположных вершин   
rect\_color - цвет внутри   
text\_color - цвет текста и линий.

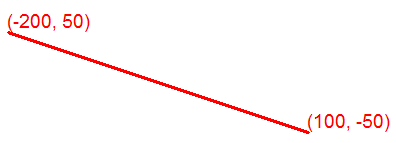
Вызов rect(-10, -20, 200, 150, "green", "red") должен нарисовать:



**Задача 15 - отрезок по 2 точкам**

Напишите функцию **line(x1, y1, x2, y2, col, width)**.  
x1, y1, x2, y2 - координаты концов отрезка  
col - цвет линии и текста   
width - ширина линии.

Вызов line(-200, 50, 100, -50, "red", 3) должен нарисовать:

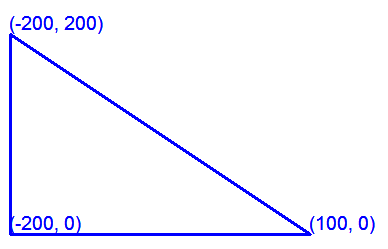


**Задача 16 - треугольник по 3 точкам**

Напишите функцию **tri(x1, y1, x2, y2, x3, y3, col)**.  
x1, y1, x2, y2, x3, y3 - координаты вершин треугольника  
col - цвет линии и текста

**Использовать функцию line(x1, y1, x2, y2, col, width)**

Вызов tri(-200, 200, 100, 0, -200, 0, "blue") должен нарисовать:

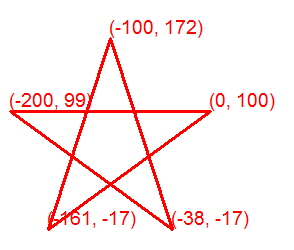


**Задача 17 - звезда**

Напишите функцию **star5(x1, y1, size, col)**.  
x1, y1 - координаты одной вершины звезды  
size - длина стороны  
col - цвет линии и текста

Подумайте, надо ли использовать функцию line(x1, y1, x2, y2, col, width)?

Вызов star5(-200, 100, 200, "red") должен нарисовать:

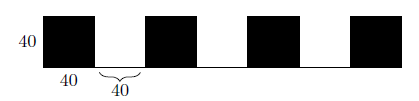


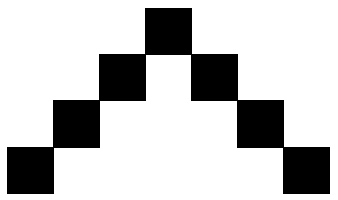
Можно написать star5 по-другому:  
x1, y1 - координаты центра звезды  
size - расстояние от центра до вершины  
col - цвет линии и текста

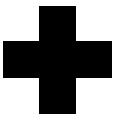
**Задачи (без геометрии)**

Написать функцию **sqb(size)**, которая рисует черный квадрат.

Этой функцией нарисовать:

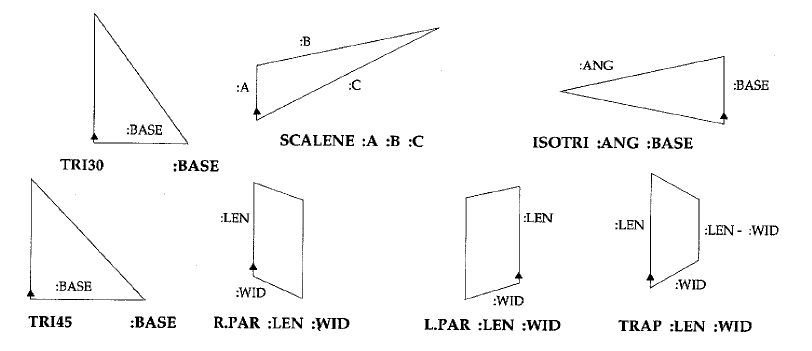








**Задачи (с геометрией)**



Написать функции:

* **tri30(a)** - треугольник с одной стороной *a* и углом 30o
* **tri45(a)** - треугольник с одной стороной *a* и углом 45o
* **tri(a, b, c)** - треугольник со сторонами *a* , *b* и *c*
* **isotri(base, ang)** - равнобедренный треугольник с основанием *base* и углом при вершине *ang*
* **rpar(a, b)** - параллелограмм со сторонами *a* и *b* и углом 30o
* **lpar(a, b)** - параллелограмм со сторонами *a* и *b* и углом 30o
* **par(a, b, ang)** - параллелограмм со сторонами *a* и *b* и углом ango
* **trap(a, b)** - равнобедренную трапецию с боковой стороной *b* и основаниями *a* и *a-b*

Этими функциями нарисовать:

