8 класс. Практическая работа №5 (Turtle)

Тема: Рекурсия. Условная инструкция if

Алгоритм выполнения работы

1. В модуле Turtle существуют два общедоступных класса с помощью которых можно создавать свои объекты черепахи и холста с персональными настройками — это:

   1. Turtle()

   2. Screen()

   Первый класс предоставляет методы управления черепахой, а второй предоставляет методы управления холстом (оконные методы). Построим два объекта этих классов с именами **window** и **t**. Определим с помощью методов этих классов настройки холста и черепахи (первая часть программы):

```
import turtle
window = turtle.Screen() # Объект - окно
window.title('Рекурсия')
window.bgcolor('black')
window.setup(width=600, height=600)
t = turtle.Turtle() # Объект - черепаха
t.speed(10)
t.color('red')
t.pensize(2)
t.hideturtle()
t.up()
t.goto(-250, -250)
t.down()
```

   Вы можете использовать, по своему усмотрению, и другие методы, известные нам по предыдущим работам. Обратите внимание на то, как в этот раз мы импортировали модуль. В дальнейшем, будет использоваться только такой подход!

2. Теперь скажем, что такое **рекурсия**. Мы уже знаем из предыдущего занятия, что такое функция. Рекурсия состоит в том, что созданная вами функция вызывает саму себя. Рекурсивный вариант реализации алгоритма обычно выглядит изящнее, делает текст программы более компактным и понятным, но исполняется медленнее. Для рекурсивного варианта программы всегда существует её нерекурсивная реализация. Для того, чтобы не происходило

бесконечных вызовов функции, рекурсивный алгоритм содержит условие завершения рекурсии (терминатор) и строится по схеме, в которой рекурсивный вызов находится в одной из ветви **условной инструкции**.

3. Условная инструкция имеет синтаксис:

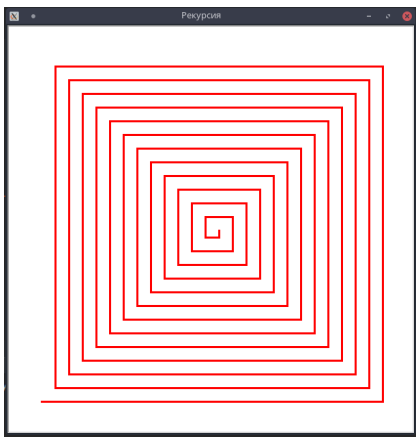
```
if выражение_условие:
    выражение_если_условие_истинно
else:
    выражение_если_условие_ложно
```

Примечание. Отступы в блоках условной инструкции должны быть одинаковой длины!

Выражение_условие можно сформировать путем использования операций сравнения: >, <, >=, <=, != и ==, а также логических операций or, and, not, например, a > 5 или b != a. Результатом вычисления выражения_условия (далее — логического выражения) являются истина (True) или ложь (False).

4. Составим программу рисования «квадратной» спирали (вторая часть программы):

```
x = int(input('x = '))
def kvsp(y):
    if y <= 0:
        return
    else:
        t.fd(y)
        t.lt(90)
        kvsp(y - 10)
kvsp(x)
window.exitonclick()
```

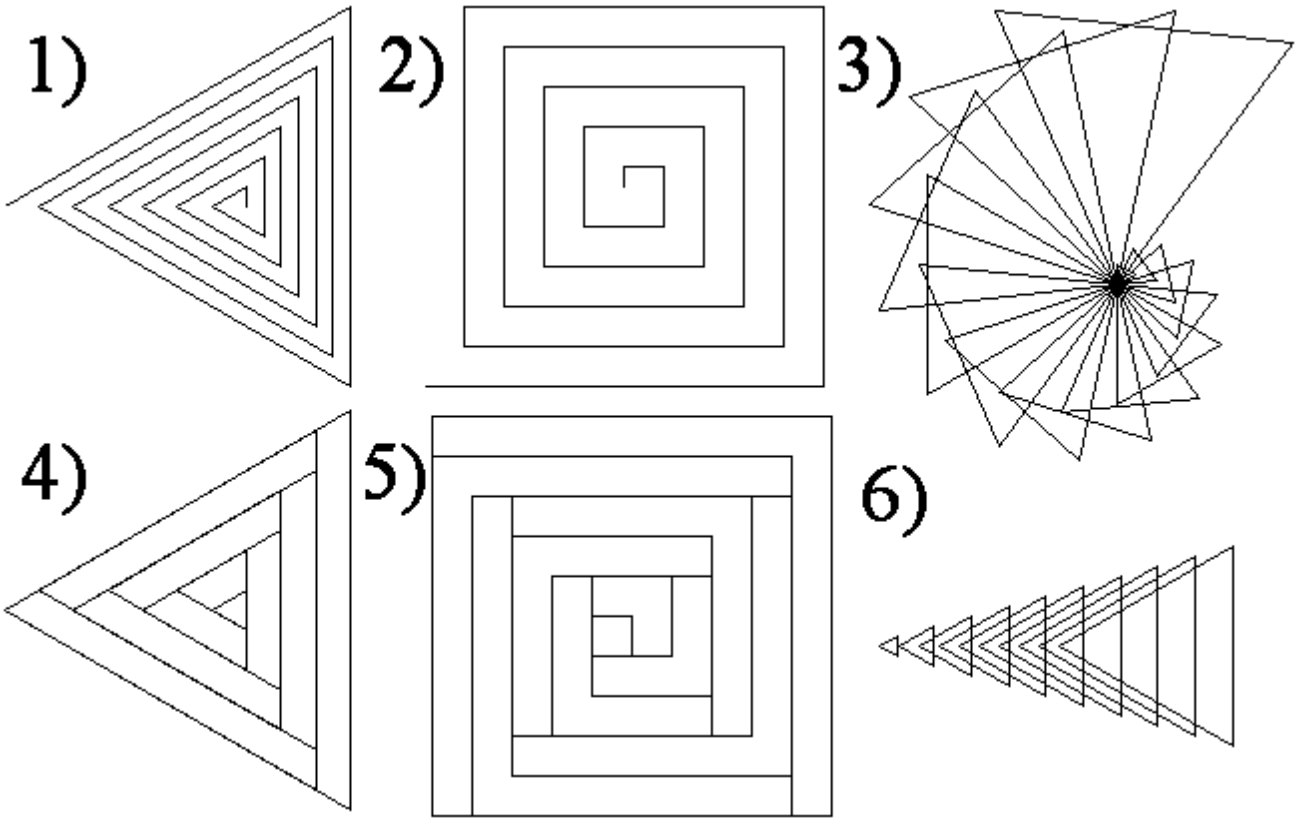


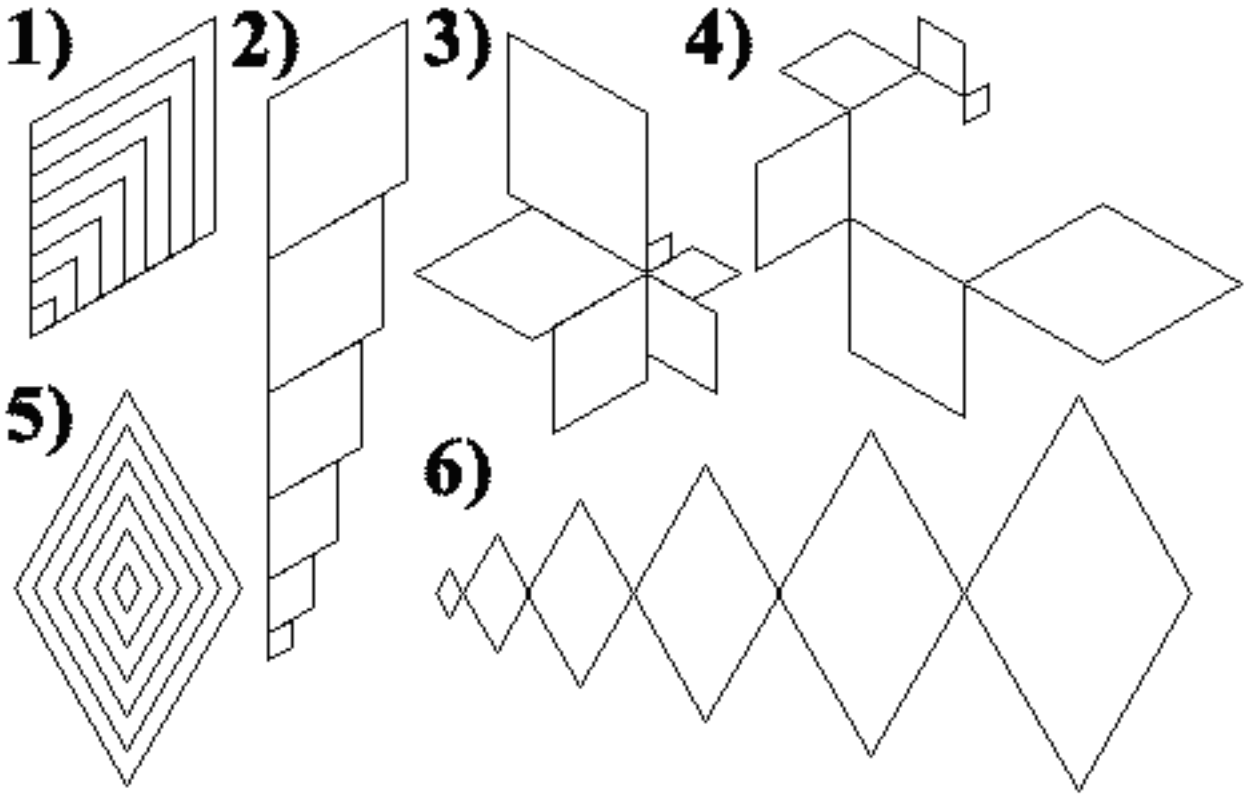
В условной инструкции функции `kvsp()` используется служебное слово return, которое обеспечивает прекращение работы функции (возвращение пустого значения), если терминальное условие приобретет значение True. Обратите внимание, что рекурсивный вызов этой функции уменьшает параметр `kvsp()` на 10 с каждым её вызовом. Это приведет к тому, что, в конце концов, он станет либо равным, либо меньшим нуля, а это, в свою очередь, завершит рекурсивные вызовы.

*Примечание. Для хранения всех отложенных вызовов рекурсивной функции, а также значений всех переменных используется структура данных **стек**. При выполнении операций в стеке расходуется время и заполняется память. Если отложенных вызовов будет слишком много — память будет переполнена и программа завершится с ошибкой.*

Метод `exitonclick()` задержит закрытие окна, пока не будет произведен клик мышью.

5. Выполните задания по вариантам (стр. 3).


1)
2)
3)
4)
5)
6)


1)
2)
3)
4)
5)
6)