. Текстовый разбор домашней работы

DZ_6_1

Задача № 1 (4936)

Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Черепаха находится в начале координат, её голова направлена вдоль положительного направления оси ординат, хвост опущен. При опущенном хвосте Черепаха оставляет на поле след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существует две команды:
Вперёд n (где n – целое число), вызывающая передвижение Черепахи на n единиц в том направлении, куда указывает её голова, и
Направо m (где m – целое число), вызывающая изменение направления движения на m градусов по часовой стрелке.
Запись Повтори k [Команда1 Команда2 ... КомандаS] означает, что последовательность из S команд повторится k раз.
Черепахе был дан для исполнения следующий алгоритм:
Повтори 70 [Вперёд 8 Направо 30].
Определите периметр фигуры, построенной Черепахой после выполнения данного алгоритма.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h0m0s

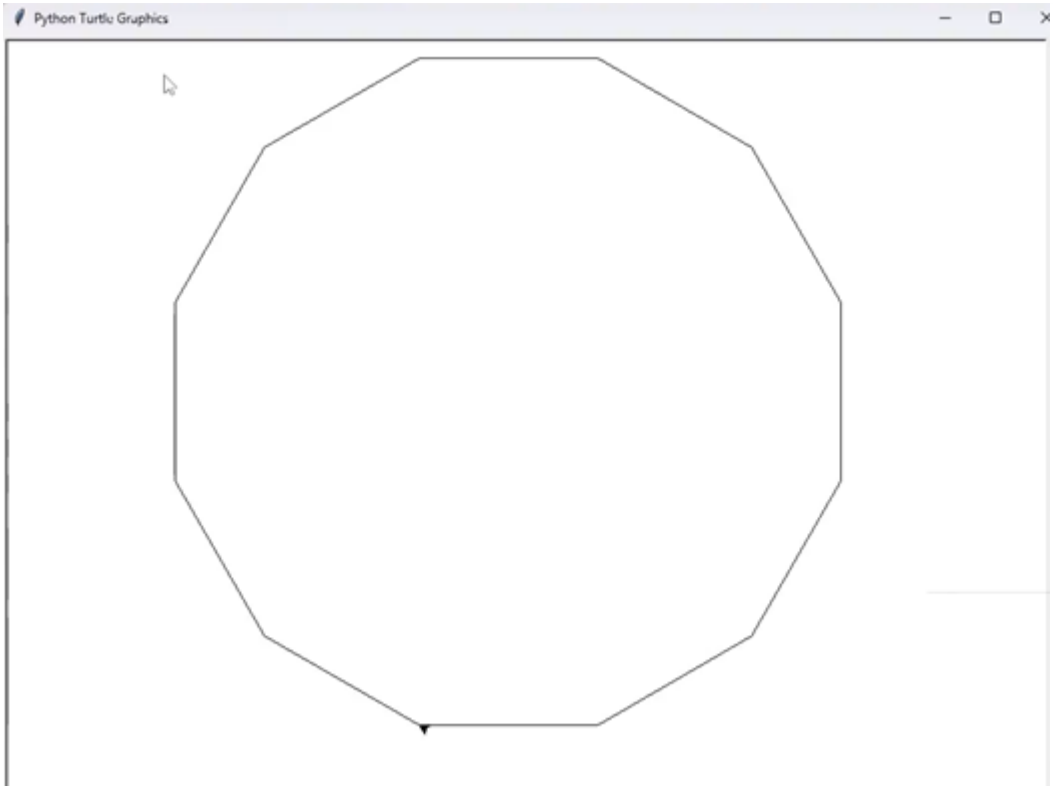
Нам нужно определить периметр фигуры, которую рисует черепаха по этому алгоритму. Давай создадим программу и посмотрим, что нарисует черепаха.
Сначала импортируем библиотеку turtle. Для начала отключим анимацию (используем команду tracer(0)), чтобы сразу увидеть результат. Затем поворачиваем черепашку на 90 градусов, чтобы она была направлена вверх, и устанавливаем размер экрана (screensize) на 10 000 x 10 000, чтобы удобно просматривать изображение.
Далее установим масштаб: пусть каждая единица равна 20 пикселям. Теперь перейдем к написанию основного цикла. Черепашка должна двигаться вперед на 8 единиц, затем повернуть направо на 30 градусов. Повторяем эти действия 70 раз.

Решение

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
for i in range(70):
    fd(8*r)
    rt(30)
update()
```

Теперь проверим, какую фигуру нарисовала черепашка.

Результат работы программы:



Получилась фигура с 12 углами – двенадцатиугольник. Длина каждой стороны составляет 8 единиц, следовательно, периметр равен 12 умножить на 8, то есть 96. Это правильный ответ.

Ответ: 96
Telegram: @fast_ege

DZ_6_2

Задача №2 (6013)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 2 [Направо 120 Вперёд 7]
Направо 300
Повтори 2 [Направо 120 Вперёд 7].
Определите, сколько точек с целочисленными координатами будут находиться внутри области, ограниченной линией, заданной данным алгоритмом. Точки на линии учитывать не следует.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h2m15s

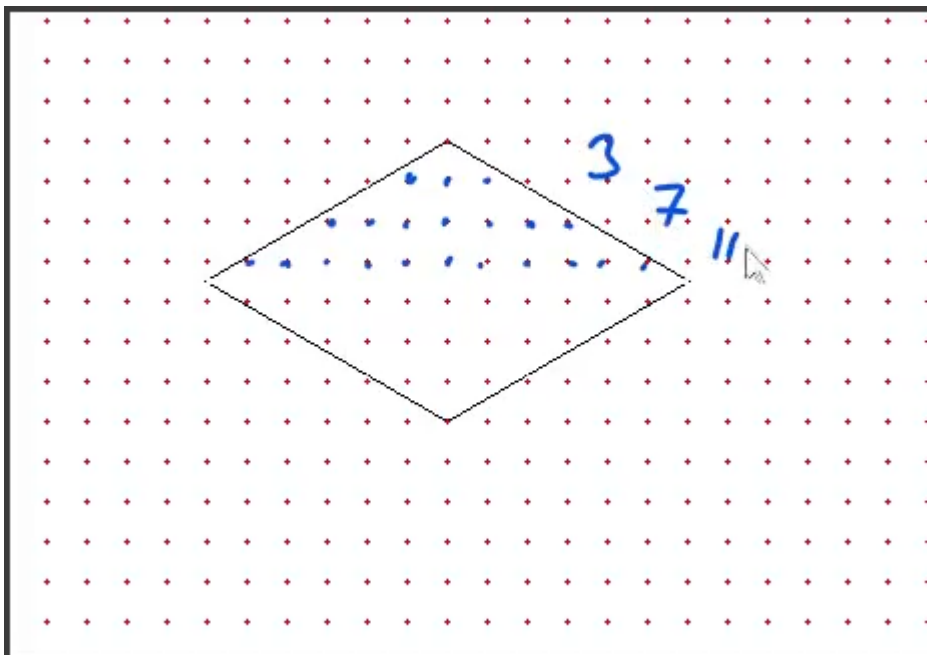
Решение

В этой задаче также требуется использовать черепаху для выполнения алгоритма. Необходимо определить количество точек с целыми координатами, которые находятся внутри области, ограниченной линией, описанной данным алгоритмом. Точки, лежащие непосредственно на границе, не учитываются.
Сначала подключаем библиотеку turtle и пишем программу:

```
from turtle import *
#инициализируем среду рисования
tracer(0)
```

```
lt(90)
screensize(10000,10000)
r = 20
#реализуем основной цикл
for i in range(2):
# Поворот вправо на 120 градусов
    rt(120)
# Движение вперёд на 7 единиц (с учётом масштаба)
    fd(7*r)
# Поворот вправо на 300 градусов
rt(300)
for i in range(2):
# Поворот вправо на 120 градусов
    rt(120)
# Движение вперёд на 7 единиц (с учётом масштаба)
    fd(7*r)
# отключаем перо, чтобы не оставлять следов при дальнейшем перемещении:
up()
#Далее создаём сетку для подсчёта точек:
for x in range(-50,50):
    for y in range(-50,50):
# Перемещаемся к точке с координатами (x*r, y*r)
        goto(x*r,y*r)
# Рисуем красную точку радиусом 3 пикселя
        dot(3,'red')
# Обновляем экран, чтобы увидеть результат:
update()
```

Результат работы программы:



Запускаем программу и смотрим на полученную картинку. Теперь необходимо подсчитать количество красных точек внутри фигуры. Видно, что фигура представляет собой ромб. Подсчитываем точки рядами:

1-й ряд: 3 точки 2-й ряд: 7 точек 3-й ряд: 11 точек

Суммируя их, получаем 21 точку. Так как таких групп рядов две, общее количество точек будет равно 21 * 2 = 42.

Ответ: 42

DZ_6_3

Задача № 3 (7585)

Черепахе был дан для исполнения следующий алгоритм:
Вправо 315
Повтори 7 [Вперёд 16 Направо 45 Вперёд 8 Направо 135]
Определите, сколько точек с целочисленными координатами будут находиться внутри области, ограниченной линией, заданной данным алгоритмом. Точки на линии учитывать не следует.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h4m55s

Решение

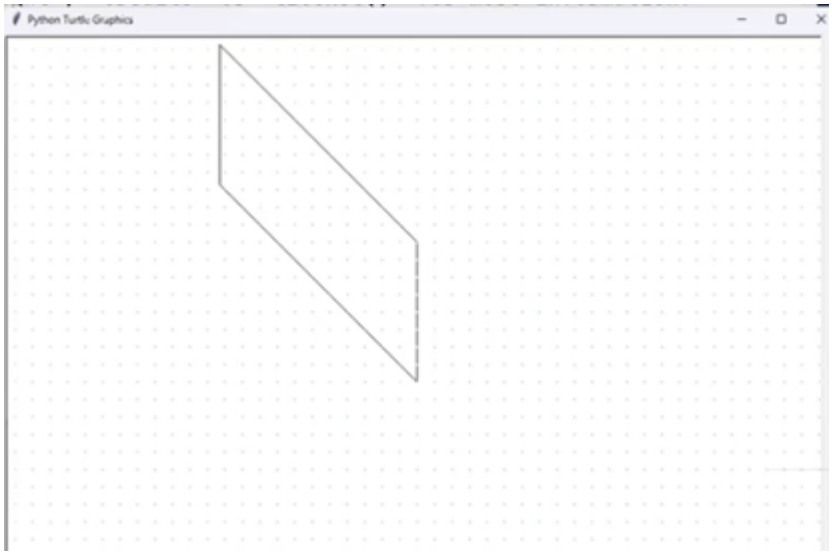
Необходимо определить количество точек с целочисленными координатами, которые находятся внутри фигуры, образованной указанным алгоритмом для Черепахи. Запишем алгоритм:
Сначала подключаемся к модулю Черепаха и устанавливаем начальные параметры. Далее поворачиваем Черепашку вправо на 315 градусов.
Затем запускаем цикл for, который будет выполняться 7 раз. Внутри цикла выполняем следующие действия:

1. Двигаемся вперёд на расстояние 16*r (где r – масштабный коэффициент).
2. Поворачиваем направо на 45 градусов.
3. Снова двигаемся вперёд, но уже на 8*r.
4. Поворачиваем направо на 135 градусов.

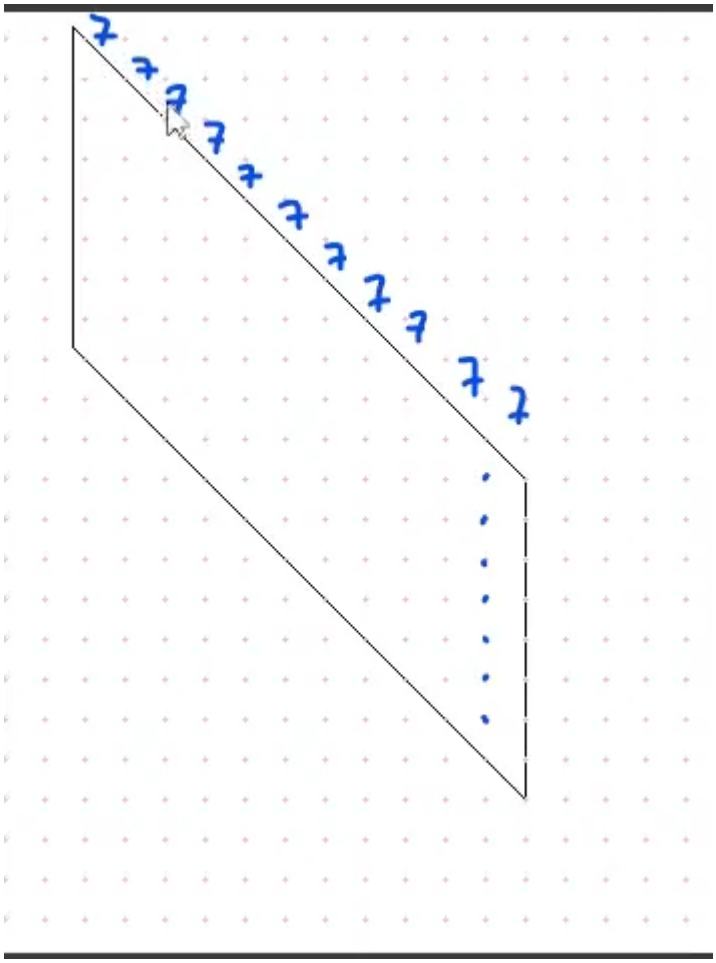
После завершения цикла поднимаем перо, чтобы не рисовать лишние линии при подсчёте точек.
Затем используем два вложенных цикла for для перебора всех возможных координат от -50 до 49 (по оси X) и от -50 до 49 (по оси Y). Для каждой пары координат перемещаем Черепашку в соответствующую точку и ставим там розовую точку размером 3 пикселя.

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
rt(315)
for i in range(7):
    fd(16*r)
    rt(45)
    fd(8*r)
    rt(135)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'pink')
update()
```

Результат работы программы:



Видно, что фигура представляет собой параллелограмм. Теперь считаем точки внутри этой фигуры, исключая те, которые лежат на границах.



В каждом горизонтальном ряду по 7 точек. Всего таких рядов 11. Умножаем количество точек в одном ряду на количество рядов: 7 *11 = 77.

Ответ: 77

DZ_6_4

Задача № 4 (5189)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 2 [Вперёд 9 Направо 90 Вперёд 15 Направо 90]
Поднять хвост
Вперёд 12 Направо 90
Опустить хвост
Повтори 2 [Вперёд 6 Направо 90 Вперёд 12 Направо 90]
Определите, сколько точек с целочисленными координатами будут находиться внутри пересечения фигур, ограниченных заданными алгоритмом линиями, включая точки на границах этого пересечения.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h7m50s

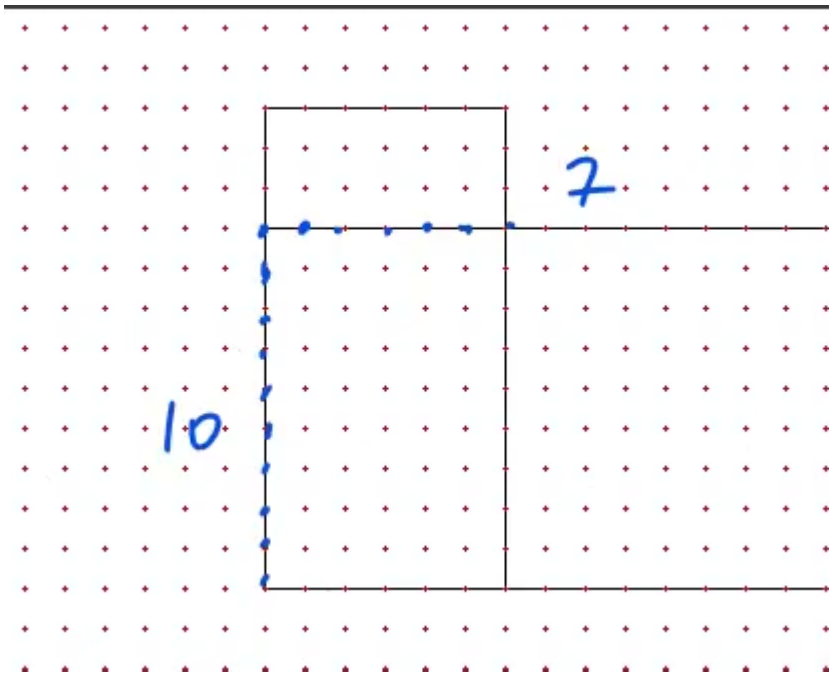
Решение

Сначала подключаемся к модулю черепахи, устанавливаем начальные параметры и создаем основной алгоритм. Используем цикл for для выполнения действий. На каждом шаге цикла черепаха выполняет следующие действия: движется вперед на расстояние 9r, затем поворачивает направо на 90 градусов, после чего проходит вперед на 15r и снова поворачивает направо на 90 градусов. Когда первый цикл завершен, поднимаем хвост черепахи (up()), перемещаемся вперед на 12 единиц и поворачиваем направо на 90 градусов. После этого опускаем хвост (down()) и запускаем второй цикл, где выполняются аналогичные действия, но с измененными значениями: сначала проходим 6 клеток, поворачиваем на 90 градусов, затем 12 клеток и снова поворот на 90 градусов. Далее рисуется сетка для визуализации результата. В завершение программы вызывается команда update, чтобы обновить экран и отобразить результат. Полный код программы:

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
for i in range(2):
    fd(9*r)
    rt(90)
    fd(15*r)
    rt(90)
up()
fd(12*r)
rt(90)
down()
for i in range(2):
    fd(6*r)
    rt(90)
    fd(12*r)
    rt(90)
up()
for x in range(-20,20):
    for y in range(-20,20):
        goto(x*r,y*r)
        dot(3,'red')
```

```
update()
```

Результат работы программы:



После запуска программы получаем два пересекающихся прямоугольника. Область пересечения — это та часть, которую необходимо проанализировать. Подсчитываем количество точек внутри этой области, включая ее границы.
Один прямоугольник имеет 7 точек вдоль одной стороны и 10 точек вдоль другой. Следовательно, общее количество точек в области пересечения составляет 7 * 10 = 70.

Ответ: 70
Telegram: @fast_ege

DZ_6_5

Задача №5 (5281)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 6 [Вперёд 25 Направо 120]
Поднять хвост
Вперёд 20 Налево 90 Назад 5
Опустить хвост
Повтори 2 [Вперёд 20 Налево 90 Вперёд 10 Налево 90]

Определите, сколько точек с целочисленными координатами будут находиться внутри пересечения фигур, ограниченных заданными алгоритмом линиями, включая точки на границах этого пересечения.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h10m45s

Решение

Для решения этой задачи необходимо определить количество точек с допустимыми координатами, которые находятся внутри пересечения двух фигур, включая границы этого пересечения.
Начнем с подключения модуля "turtle" и установки начальных параметров. Далее запускается первый цикл, который выполняет следующие действия

```
for i in range(6):
# Переместиться вперед на 25 единиц
   fd(25*r)
# Повернуть направо на 120 градусов
    rt(120)
```

После завершения первого цикла выполняем следующие шаги:

```
up()
# Поднять перо и переместиться вперед на 20 единиц
fd(20*r)
#Повернуть влево на 90 градусов
lt(90)
 # Отступить назад на 5 единиц
bk(5*r)
 # Опустить перо
down()
```

второй цикл:

```
for i in range(2):
 # Переместиться вперед на 20 единиц
fd(20*r)
 # Повернуть влево на 90 градусов
    lt(90)
 # Переместиться вперед на 10 единиц
    fd(10*r)
 # Повернуть влево еще раз на 90 градусов
    lt(90)
#поднимаем перо, чтобы не рисовать лишние линии при подсчёте точек.
up()
```

Затем используем два вложенных цикла for для перебора всех возможных координат от -50 до 49 (по оси X) и от -50 до 49 (по оси Y).

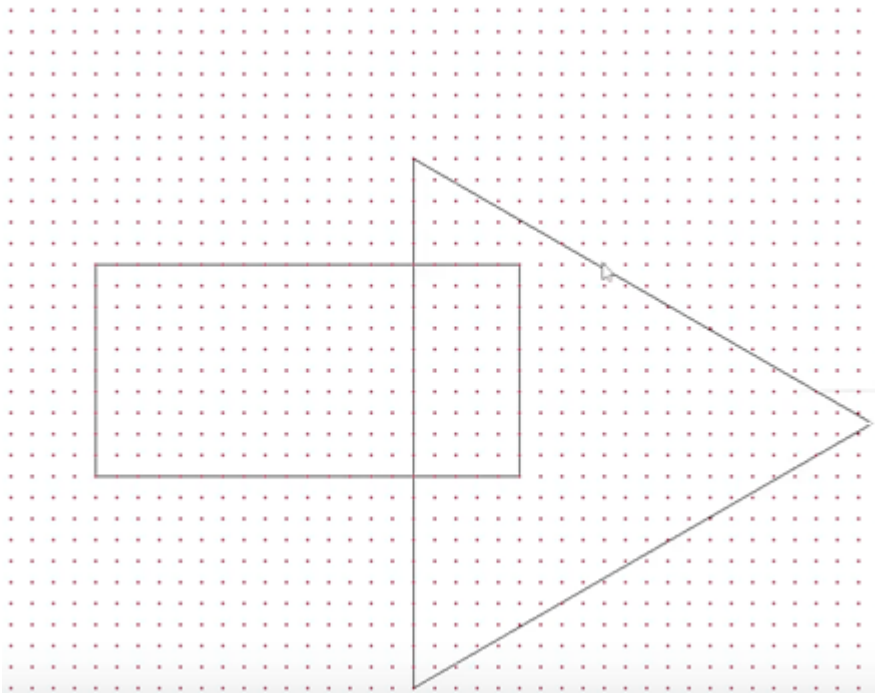
```
for x in range(-50,50):
    for y in range(-50,50):
# Перейти к точке (x*r, y*r)
        goto(x*r,y*r)
# Нарисовать красную точку радиусом 3 пикселя
        dot(3,'red')
```

Полный код программы:

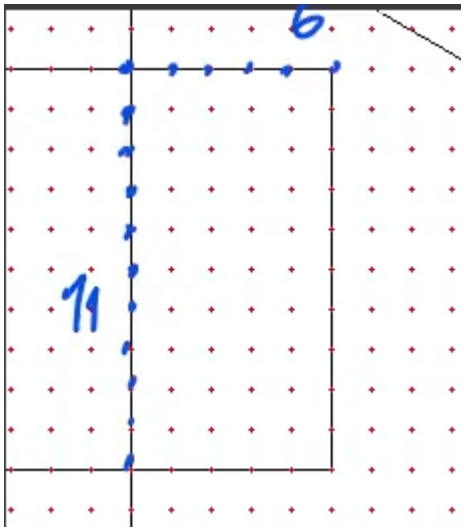
```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20

for i in range(6):
    fd(25*r)
```

```
    rt(120)
up()
fd(20*r)
lt(90)
bk(5*r)
down()
for i in range(2):
    fd(20*r)
    lt(90)
    fd(10*r)
    lt(90)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



В результате выполнения программы получаем изображение, где область пересечения представлена маленьким прямоугольником. Теперь подсчитаем количество точек внутри этого прямоугольника:



Прямоугольник имеет размеры 6x11 точек, следовательно, общее количество точек равно $6 * 11 = 66$.
Ответ: 66

DZ_6_6

Задание №6 (6996)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 2 [Вперёд 24 Направо 90 Вперёд 16 Направо 90]
Поднять хвост
Вперёд 10 Направо 90 Вперёд 8 Налево 90
Опустить хвост
Повтори 2 [Вперёд 15 Направо 90 Вперёд 28 Направо 90]

Определите, сколько точек с целочисленными координатами будут находиться внутри пересечения фигур, ограниченных заданными алгоритмом линиями, не включая точки на линиях.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h13m40s

Решение

Мы снова работаем с алгоритмом для Черепахи. Нужно определить количество точек с целочисленными координатами, которые находятся внутри области пересечения двух фигур, при этом, мы не учитываем точки, лежащие непосредственно на границах этих фигур. Подключаемся к Черепахе и начинаем писать программу. Используем цикл for i in range(2), который означает, что Черепаха должна пройти следующие шаги:

- Вперед на 24 единицы.
- Повернуть направо на 90 градусов.
- Снова вперед на 16 единиц.
- Еще раз повернуть направо на 90 градусов.

После завершения этого цикла Черепаха поднимает хвост (команда up) и проходит еще несколько шагов:

- Вперед на 10 клеток.
- Поворот направо на 90 градусов.
- Вперед на 8 клеток.
- Поворот налево на 90 градусов.

Далее начинается второй цикл for, где Черепаха выполняет такие действия:

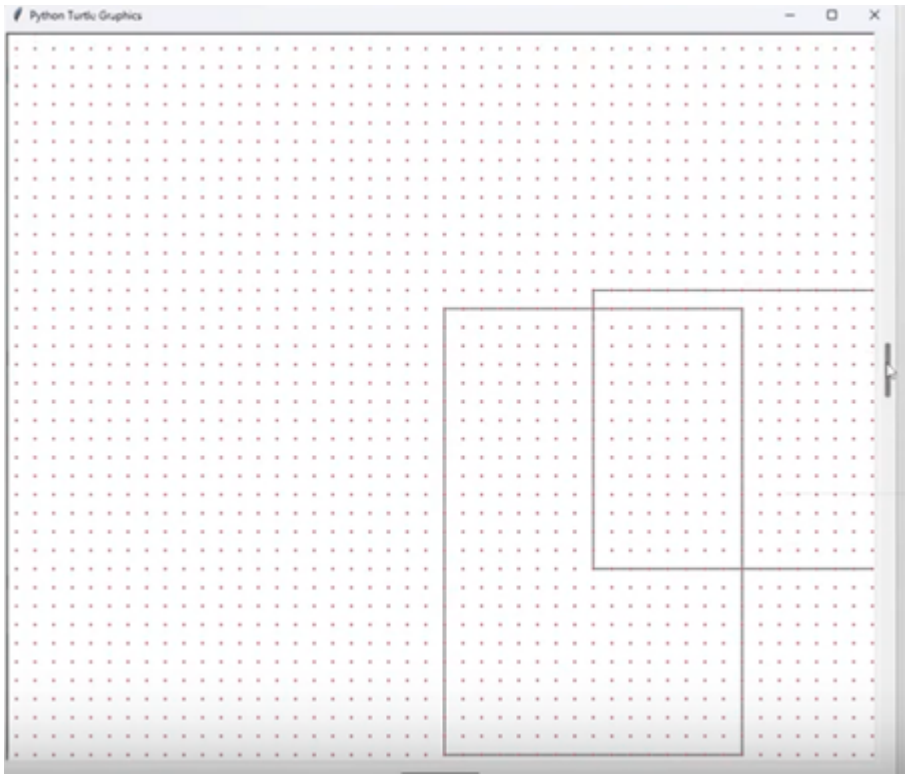
- Проходит вперед на 15 единиц.
- Поворачивает направо на 90 градусов.
- Проходит вперед на 28 единиц.
- Завершает цикл поворотом направо на 90 градусов.

Теперь Черепаха рисует сетку точек, используя вложенные циклы for x in range(-50, 50) и for y in range(). Она перемещается в точку с координатами (x, y) и ставит там красную точку размером 3 пикселя.

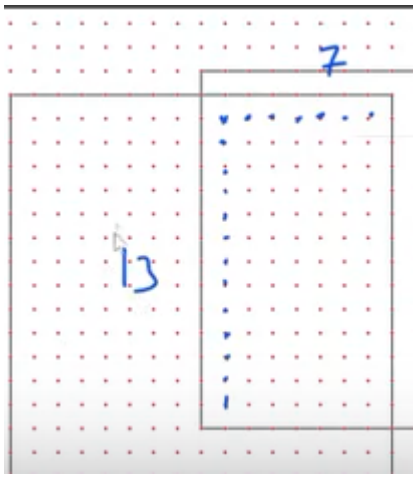
Вот итоговый код программы:

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
for i in range(2):
    fd(24*r)
    rt(90)
    fd(16*r)
    rt(90)
up()
fd(10*r)
rt(90)
fd(8*r)
lt(90)
down()
for i in range(2):
    fd(15*r)
    rt(90)
    fd(28*r)
    rt(90)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



Нас интересуют точки, находящиеся строго внутри пересечения фигур, исключая те, что лежат на их границах. Таким образом, нужно учитывать только внутреннюю область фигуры. По горизонтали у нас 7 точек, а по вертикали – 13. Получаем прямоугольник размером 7x13,



что дает нам 91 точку.

Ответ: 91

DZ_6_7

Задание № 7(6741)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 4 [Вперёд 10 Направо 270]
Поднять хвост

Вперёд 3 Направо 270 Вперёд 5 Направо 90
Опустить хвост
Повтори 2 [Вперёд 10 Направо 270 Вперёд 12 Направо 270]

Определите, сколько точек с целочисленными координатами будут находиться внутри объединения фигур, ограниченных
заданными алгоритмом линиями, включая точки на линиях.
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h16m50s

Решение
Необходимо определить количество точек с целыми координатами, которые попадают внутрь фигуры, ограниченной заданными линиями, включая границы. Для начала запускаем Черепаху, устанавливаем начальные параметры и начинаем выполнение программы.

```
for i in range(4):
    fd(10*r)
#Это эквивалентно повороту налево на 90 градусов
    rt(270)
```

После этого поднимаем хвост Черепахи (up()).

Далее двигаемся вперед на 3 единицы, поворачиваемся направо на 270 градусов, а затем снова идем вперед на 5 единиц и поворачиваемся направо на 90 градусов.

```
fd(3*r)
rt(270)
fd(5*r)
rt(90)
```

Опускаем хвост Черепахи (down()) и продолжаем.

```
for i in range(2):
    fd(10*r)
    rt(270)
    fd(12*r)
    rt(270)
```

Создаем сетку, для каждого значения x в диапазоне от -50 до 49 рисуем зеленую точку размером 3 пикселя.

```
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'green')
```

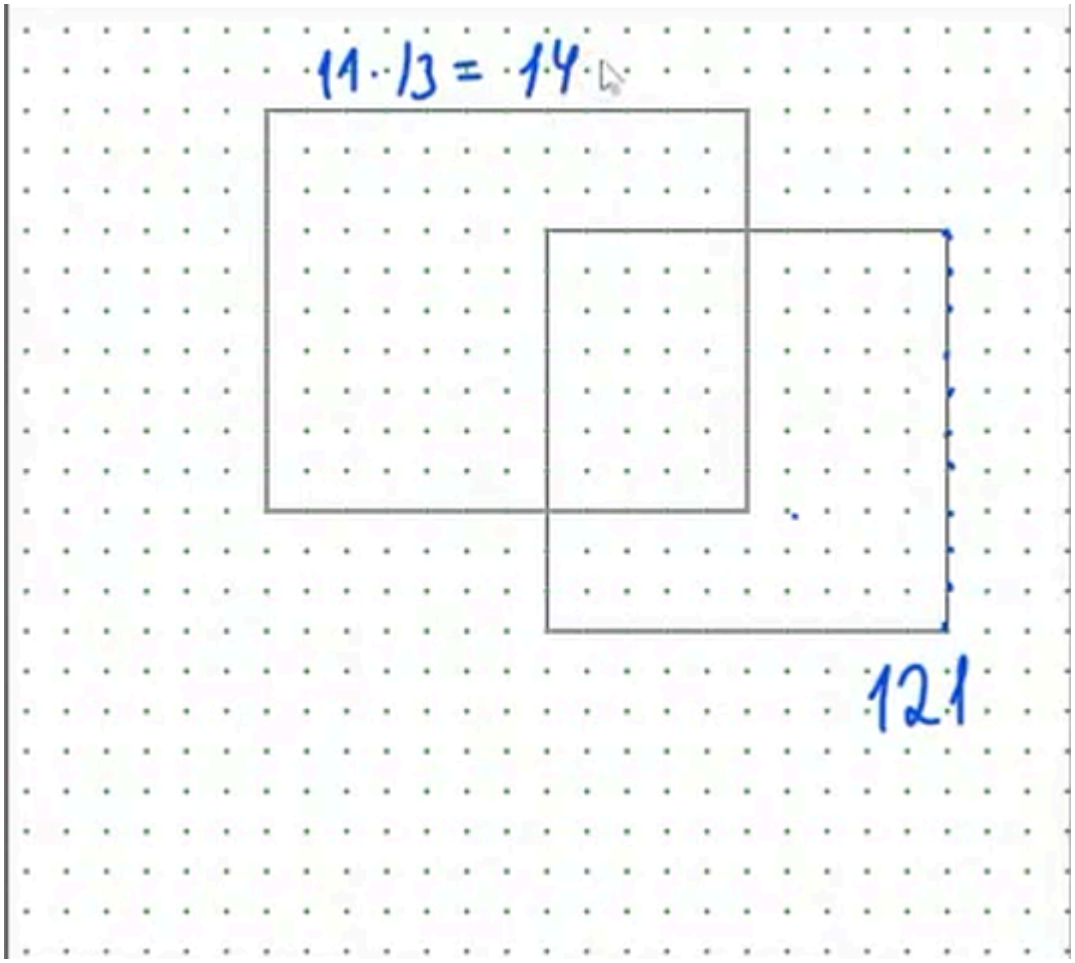
В конце обновляем экран update(), чтобы увидеть результат.

Полный код программы:

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
```

```
for i in range(4):
    fd(10*r)
#Это эквивалентно повороту налево на 90 градусов
    rt(270)
up()
fd(3*r)
rt(270)
fd(5*r)
rt(90)
down()
for i in range(2):
    fd(10*r)
    rt(270)
    fd(12*r)
    rt(270)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'green')
update()
```

Результат работы программы:



Запустив программу, мы получим изображение двух прямоугольников. Теперь подсчитаем количество точек внутри их объединения. Начнем с первого квадрата 10×10. Количество точек внутри него (включая границы) равно 11×11=121. Второй прямоугольник имеет размеры 10×12, поэтому количество точек внутри него составляет 11×13=143. Пересечение этих прямоугольников содержит 6×8=48 точек. Таким образом, общее количество точек в объединении двух прямоугольников будет равно 143+121-48=216.

Ответ: 216

DZ_6_8

Задание №8 (8150)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 2 [Вперёд 5 Направо 90 Вперёд 15 Направо 90]
Поднять хвост
Вперёд -7 Направо 90 Вперёд 12 Налево 90
Опустить хвост
Повтори 2 [Вперёд 65 Направо 90 Вперёд 120 Направо 90]
Определите периметр пересечения фигур, ограниченного заданными алгоритмом линиями.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h21m0s

Решение

Исполнителю Черепаха необходимо выполнить следующий алгоритм: определить периметр фигуры, образованной пересечением двух других фигур. Для этого нужно сначала запустить Черепаху и выполнить необходимые действия.
Сначала инициализируем среду:

```
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
```

Далее следуем инструкциям алгоритма:

1. Первый цикл:

```
for i in range(2):
```

```
# Переместиться вперед на 5 единиц (с учетом масштаба r)
```

```
    fd(5*r)
```

```
 # Повернуть направо на 90 градусов
```

```
rt(90)
```

```
# Переместиться вперед на 15 единиц
```

```
fd(15*r)
```

```
# Повернуть направо еще раз на 90 градусов
```

```
rt(90)
```

Поднимаем "хвост" Черепахи, чтобы она не оставляла следов при перемещении и перемещаем черепаху.

```
up()
# Перемещаемся назад на 7 единиц
fd(-7*r)
 # Поворачиваем направо на 90 градусов
rt(90)
 # Перемещаемся вперед на 12 единиц
fd(12*r)
 # Поворачиваем налево на 90 градусов
```

Опускаем "хвост", чтобы снова начать рисовать.

2. Второй цикл:

```
for i in range(2):
```

```
 # Перемещаемся вперед на 65 единиц
```

```
    fd(65*r)
```

```
 # Поворот направо на 90 градусов
```

```
    rt(90)
```

```
 # Перемещаемся вперед на 120 единиц
```

```
    fd(120*r)
```

```
 # Еще один поворот направо на 90 градусов
```

```
    rt(90)
```

3. Снова поднимаем "хвост" и рисуем сетку:

```
for x in range(-50,50):
```

```
    for y in range(-50,50):
```

```
 # Переходим к координатам (x*r, y*r)
```

```
        goto(x*r,y*r)
```

```
 # Ставим красную точку радиусом 3 пикселя
```

```
        dot(3,'red')
```

```
# Обновляем экран
```

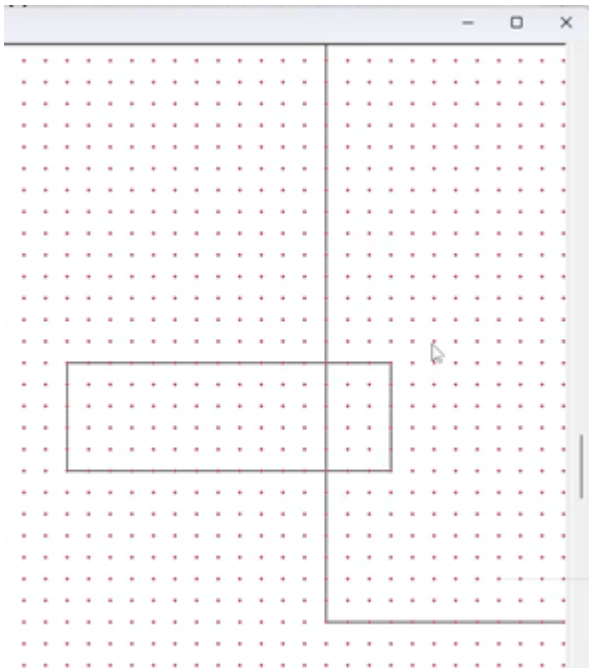
```
update()
```

```

```

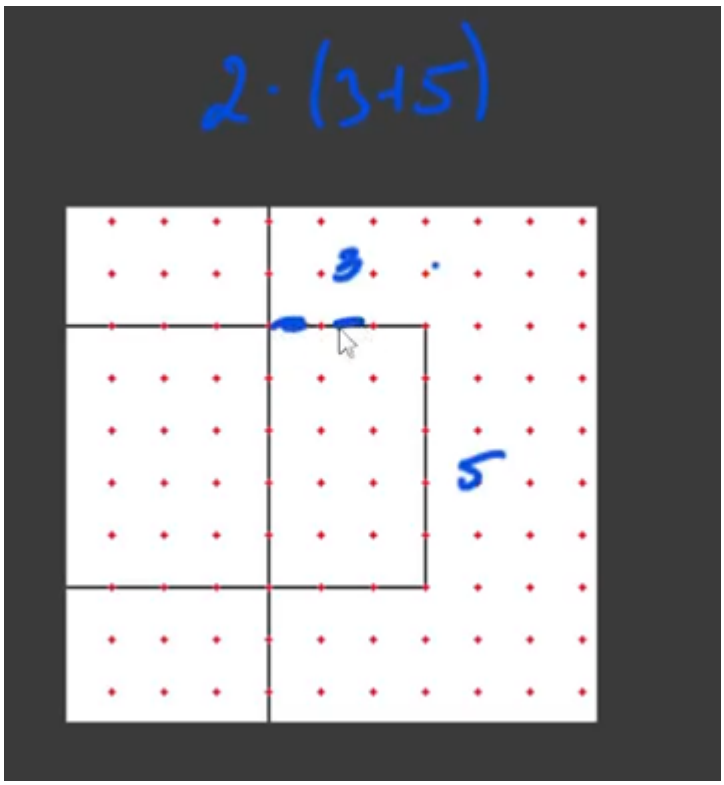
Полный код программы:

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
for i in range(2):
    fd(5*r)
    rt(90)
    fd(15*r)
    rt(90)
up()
fd(-7*r)
rt(90)
fd(12*r)
lt(90)
down()
for i in range(2):
    fd(65*r)
    rt(90)
    fd(120*r)
    rt(90)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



Теперь, когда фигура нарисована, можно вычислить периметр пересечения. Нам нужны длины сторон пересечения:



- Одна сторона имеет длину 3 единицы,
- Другая сторона — 5 единиц.

Будем считать именно в единичных отрезках. Очень важно не путать точки и отрезки.
Суммарная длина периметра будет равна удвоенной сумме этих значений:
$P = 2 * (3 + 5) = 16$.

Ответ: 16
Telegram: @fast_ege

DZ_6_9

Задание №9(12460)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 3 [Опустить хвост
Повтори 2 [Вперёд 7 Направо 90 Вперёд 7 Направо 90]
Поднять хвост
Вперёд 6 Направо 90 Вперёд 6 Налево 90]

Определите длину замкнутой ломаной, которая является границей объединения фигур, очерченных заданными алгоритмом линиями. В ответе укажите только число. Единицы измерения указывать не нужно.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h24m0s

Решение

Здесь необходимо вычислить длину замкнутого многоугольника, образованного границами фигур, созданных данным алгоритмом. Другими словами, требуется найти периметр объединения этих фигур.

Обратите внимание на важный нюанс: внешний цикл завершается после выполнения внутреннего цикла. Внутри основного цикла находится меньший цикл. Этот аспект следует учитывать при написании программы.

Мы начинаем с внешнего цикла for i in range(3), где опускается "хвост" черепахи. Затем идет внутренний цикл for g in range(2):

```
    for j in range(2):
        fd(7*r)
        rt(90)
        fd(7*r)
        rt(90)
```

После завершения внутреннего цикла "хвост" поднимается, и выполняются следующие действия:

```
    fd(6*r)
    rt(90)
    fd(6*r)
    lt(90)
```

Этот блок команд выполняется трижды (по количеству итераций во внешнем цикле). Внутренний цикл повторяет свои шаги дважды каждый раз.

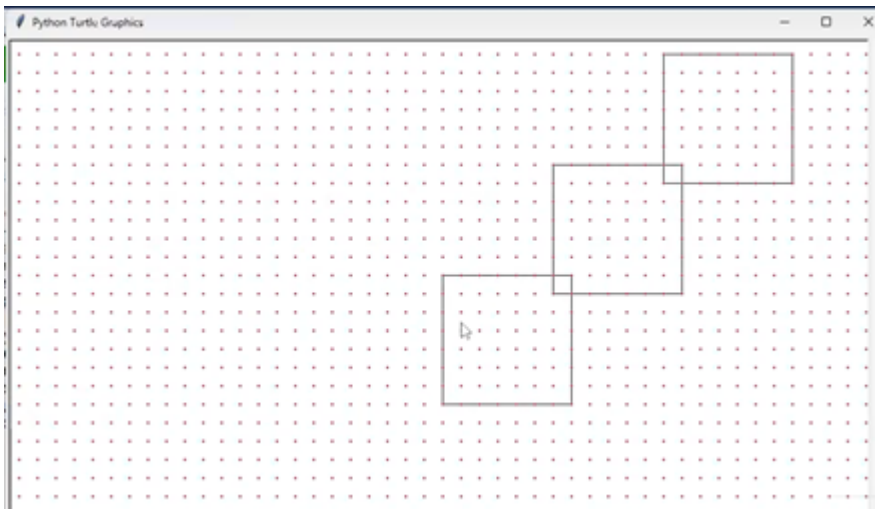
Теперь рисуем сетку:

```
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
```

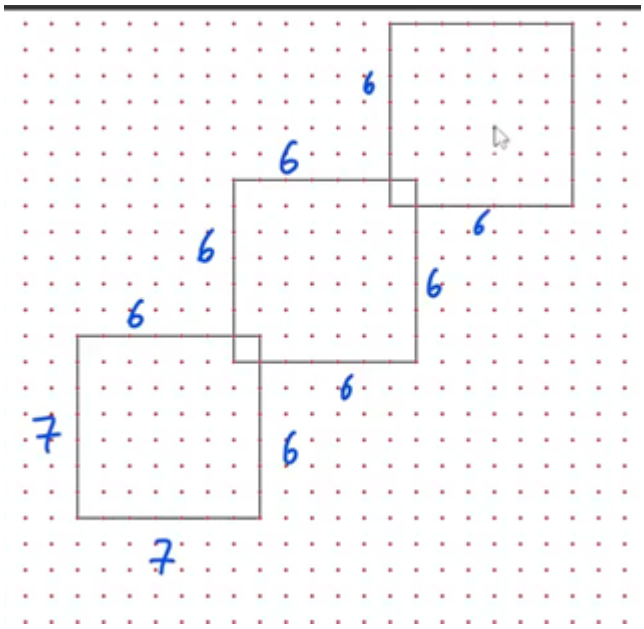
Полный код программы

```
from turtle import *

tracer(0)
lt(90)
screensize(10000,10000)
r = 20

for i in range(3):
    down()
    for j in range(2):
        fd(7*r)
        rt(90)
        fd(7*r)
        rt(90)
    up()
    fd(6*r)
    rt(90)
    fd(6*r)
    lt(90)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



Программа завершена, и мы видим две фигуры. Периметр их объединения равен сумме длин всех сторон, составляющих границу. Стороны квадратов имеют длину 7 единиц, а стороны прямоугольников – 6 единиц.



Подсчитав количество каждой из них, получаем:
4 *7 + 8 * 6 = 28 + 48 = 76
Таким образом, периметр объединения фигур составляет 76 единиц.

Ответ: 76
Telegram: @fast_ege

DZ_6_10

Задание №10 (17669)

Черепахе был дан для исполнения следующий алгоритм:
Повтори 4 [Вперёд 19 Направо 90 Вперед 30 Направо 90]
Поднять хвост
Вперед 2 Направо 90 Вперёд 8 Налево 90
Опустить хвост
Повтори 4 [Вперёд 93 Направо 90 Вперёд 97 Направо 90]
Определите площадь области пересечения фигур, ограниченных заданными алгоритмом линиями.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h28m25s

Решение

Нам необходимо вычислить площадь фигуры, образованной пересечением линий. Для этого нужно сначала построить эти линии, найти область их пересечения и рассчитать её площадь.
Для начала выполним следующие шаги:

```
for i in range(4):
# Двигаемся вперёд на 19 единиц
    fd(19*r)
 # Поворачиваем направо на 90 градусов
    rt(90)
 # Двигаемся вперёд ещё на 30 единиц
    fd(30*r)
 # Снова поворачиваем направо на 90 градусов
    rt(90)
```

Теперь поднимаем "хвост" черепахи, чтобы она не оставляла след при движении:

```
# Перемещаемся вперёд на 2 единицы
 fd(2*r)
 # Поворачиваем направо на 90 градусов
rt(90)
 # перемещаемся вперёд
fd(8*r)
 # Поворачиваем налево на 90 градусов
lt(90)
```

Опускаем "хвост", чтобы снова начать рисование и, затем, копируем предыдущий цикл и выполняем аналогичные действия:

```
for i in range(4):
    fd(93*r)
    rt(90)
    fd(97*r)
    rt(90)
```

Чтобы лучше визуализировать результат, добавим сетку:

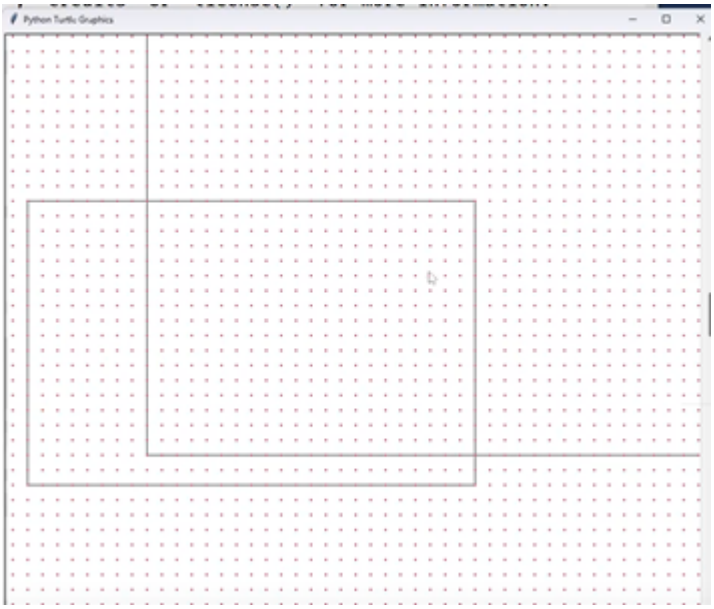
```
for x in range(-50,50):
    for y in range(-50,50):
 # Переходим в точку (x, y)
        goto(x*r,y*r)
 # Рисуем красную точку радиусом 3 пикселя
        dot(3,'red')
# Обновляем экран
update()
```

Полный код программы:

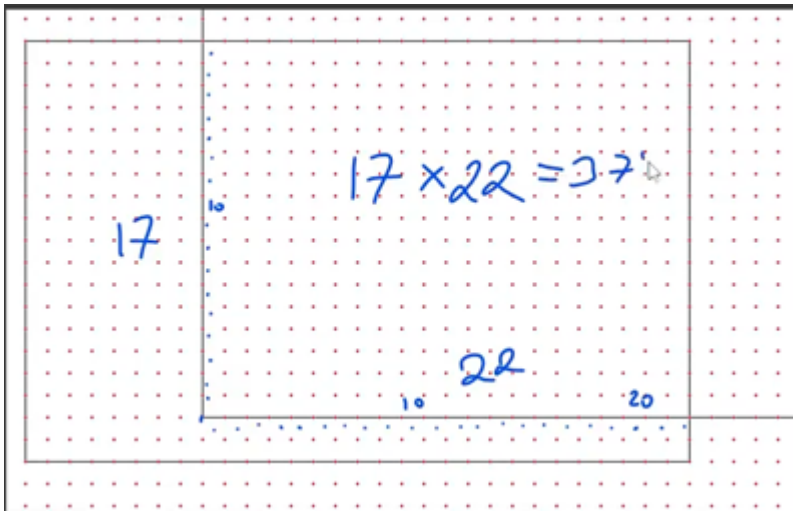
```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20

for i in range(4):
    fd(19*r)
    rt(90)
    fd(30*r)
    rt(90)
up()
```

```
fd(2*r)
rt(90)
fd(8*r)
lt(90)
down()
for i in range(4):
    fd(93*r)
    rt(90)
    fd(97*r)
    rt(90)
up()
for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



Теперь, когда изображение построено, можно оценить размеры области пересечения. Мы видим, что стороны этой области составляют 17 и 22 единицы.



Таким образом, площадь пересечения будет равна произведению этих чисел:
Площадь = 17 *22 = 374

Ответ : 374
Telegram: @fast_ege

DZ_6_11

Задача № 11(4810)

Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Черепаха находится в начале координат, её голова направлена вдоль положительного направления оси ординат, хвост опущен. При опущенном хвосте Черепаха оставляет на поле след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существует две команды:

Вперёд n (где n – целое число), вызывающая передвижение Черепахи на n единиц в том направлении, куда указывает её голова, и Направо m (где m – целое число), вызывающая изменение направления движения на m градусов по часовой стрелке.

Запись Повтори k [Команда1 Команда2 ... КомандаS] означает, что последовательность из S команд повторится k раз.

Черепахе был дан для исполнения следующий алгоритм:

Повтори 8 [Повтори 4 [Вперёд 5 Направо 30 Вперёд 6 Направо 150] Направо 60].

Определите площадь получившейся фигуры в квадратных единицах.

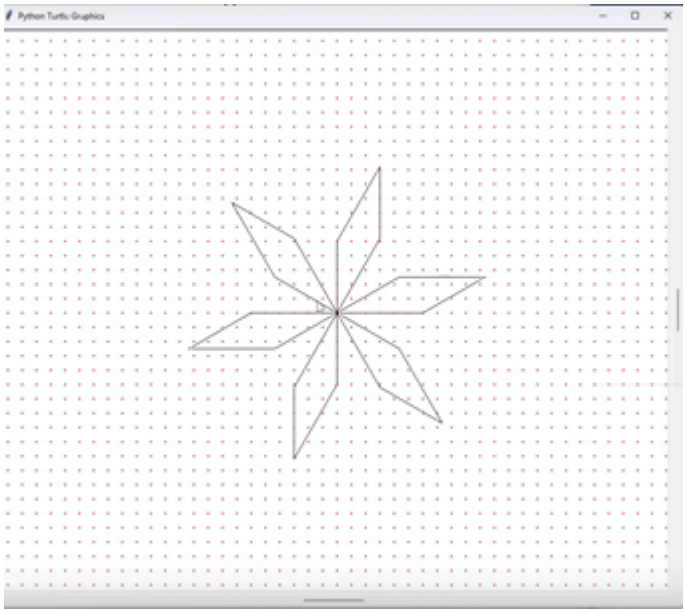
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h32m0s

Решение

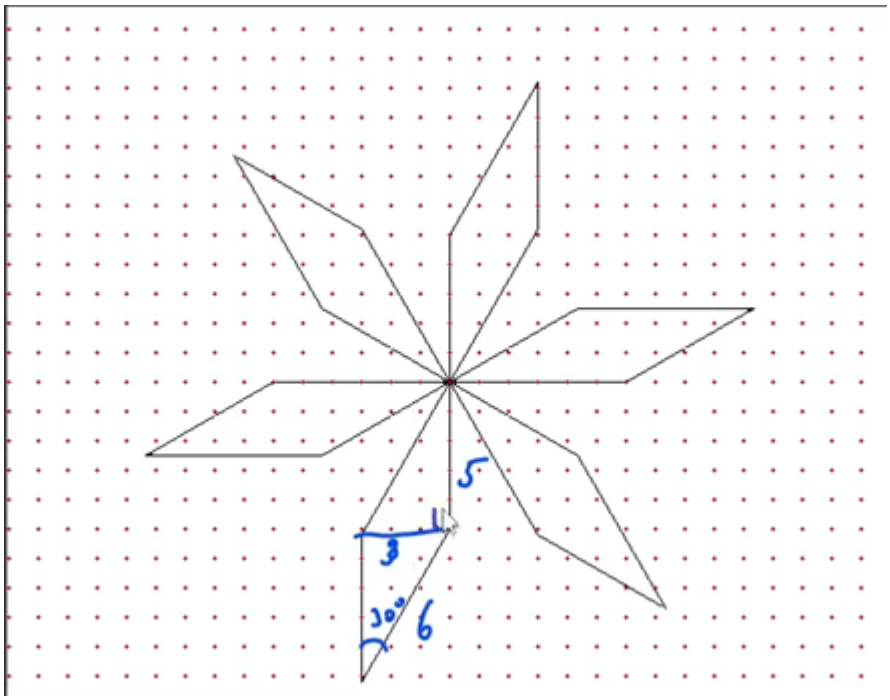
Обратим внимание, что алгоритм, который нам дан достаточно такой вложенный. В процессе выполнения программы создается фигура, состоящая из нескольких частей, которые можно назвать лепестками. Сначала мы подключаем библиотеку turtle, чтобы использовать функции управления "Черепахой". Затем создаем основной цикл, который управляет внешним контуром фигуры, и внутренний цикл, отвечающий за рисование каждого отдельного лепестка. Для создания сетки координат используем два цикла for, чтобы заполнить пространство точками.

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20

for i in range(8):
    for j in range(4):
        fd(5*r)
        rt(30)
        fd(6*r)
        rt(150)
    rt(60)
up()

for x in range(-50,50):
    for y in range(-50,50):
        goto(x*r,y*r)
        dot(3,'red')
update()
```

Результат работы программы:



После запуска программы мы видим фигуру, напоминающую цветок с шестью лепестками. Каждый лепесток представляет собой параллелограмм. Чтобы вычислить общую площадь фигуры, сначала найдем площадь одного параллелограмма, а затем умножим её на количество лепестков.



Площадь параллелограмма можно найти по формуле S = a *h, где a — основание, h — высота. Основанием является сторона длиной 5 единиц, а высотой — перпендикулярная ей линия, напротив угла в 30 градусов. Высота равна половине гипотенузы треугольника, образованного сторонами 5 и 6, то есть = 3.

Таким образом, площадь одного параллелограмма составляет 3 *5 = 15. Так как всего у нас шесть таких параллелограммов, общая площадь фигуры будет 6 * 15 = 90 квадратных единиц.

Ответ: 90

DZ_6_12

Задача № 12(18612)

Черепаха выполнила следующую программу:
Повтори 2 [Вперёд 24 Направо 90 Вперёд 10 Направо 90]
Вперёд 3 Налево 90 Вперёд 13 Направо 90
Повтори 2 [Вперёд 9 Направо 90 Вперёд 32 Направо 90]
Полученный при выполнении этой программы рисунок можно рассматривать как набор непересекающихся прямоугольников. Определите наибольшую из площадей этих прямоугольников. В ответе запишите только число – наибольшую площадь в условных единицах

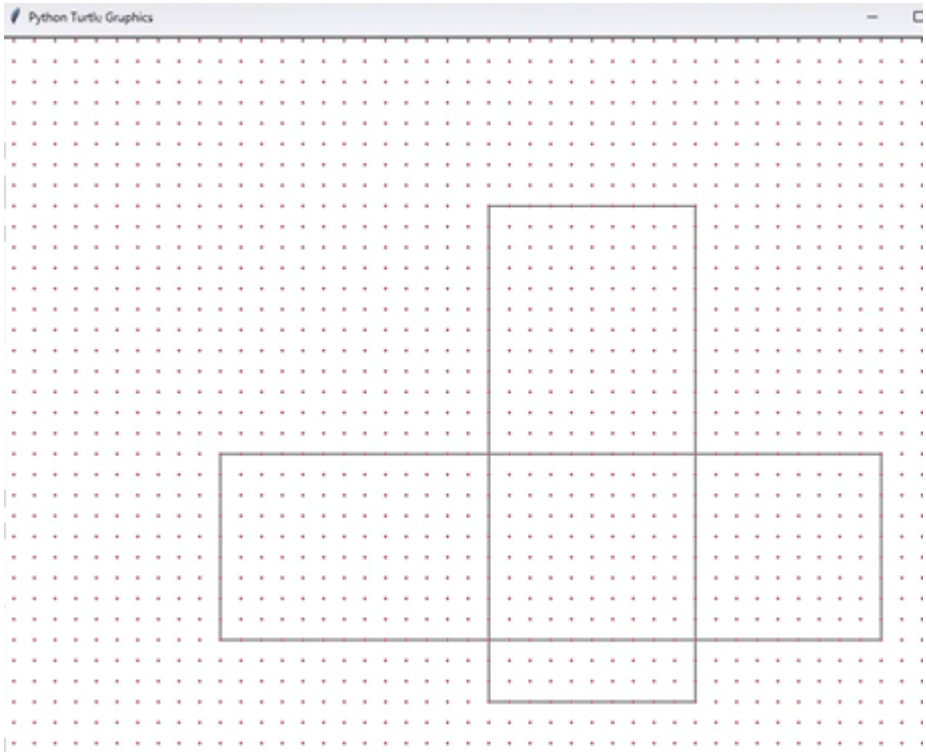
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241241?t=0h34m30s

Решение

Рассмотрим алгоритм выполнения программы:

```
from turtle import *
tracer(0)
lt(90)
screensize(10000,10000)
r = 20
for i in range(2):
 # Двигаемся вперёд на 24 единицы, поворачиваем направо на 90 градусов
    fd(24*r)
 # двигаемся вперёд на 10 единиц, снова поворачиваем направо на 90 градусов
    rt(90)
    fd(10*r)
    rt(90)
 # двигаемся вперёд на 3 единицы, поворачиваем налево на 90 градусов
fd(3*r)
lt(90)
# двигаемся вперёд на 13 единиц, поворачиваем направо на 90 градусов
fd(13*r)
rt(90)
for i in range(2):
    fd(9*r)
    rt(90)
    fd(32*r)
    rt(90)
up()
#Создаём сетку
for x in range(-50,50):
    for y in range(-50,50):
 # Перемещаемся к координатам (x*r, y*r)
        goto(x*r,y*r)
 # Рисуем красную точку радиусом 3 пикселя
        dot(3,'red')
update()
```

Результат работы программы:



Теперь проанализируем полученное изображение. На нём изображены пять прямоугольников, которые имеют следующие размеры:

1. Прямоугольник размером 10 *3 имеет площадь 30.
2. Прямоугольник размером 9 *9 имеет площадь 81.
3. Прямоугольник размером 10 *9 имеет площадь 90.
4. Прямоугольник размером 10 *12 имеет площадь 120.
5. Прямоугольник размером 9 *13 имеет площадь 117.

Из всех перечисленных площадей максимальная равна 120.

Ответ: 120