Akademia WSB

Dąbrowa Górnicza, Cieszyn, Olkusz, Żywiec, Kraków

Student: Marcin Czernek

Numer albumu: 39924

Semestr: 6

Kierunek: Informatyka

Studia: Niestacjonarne

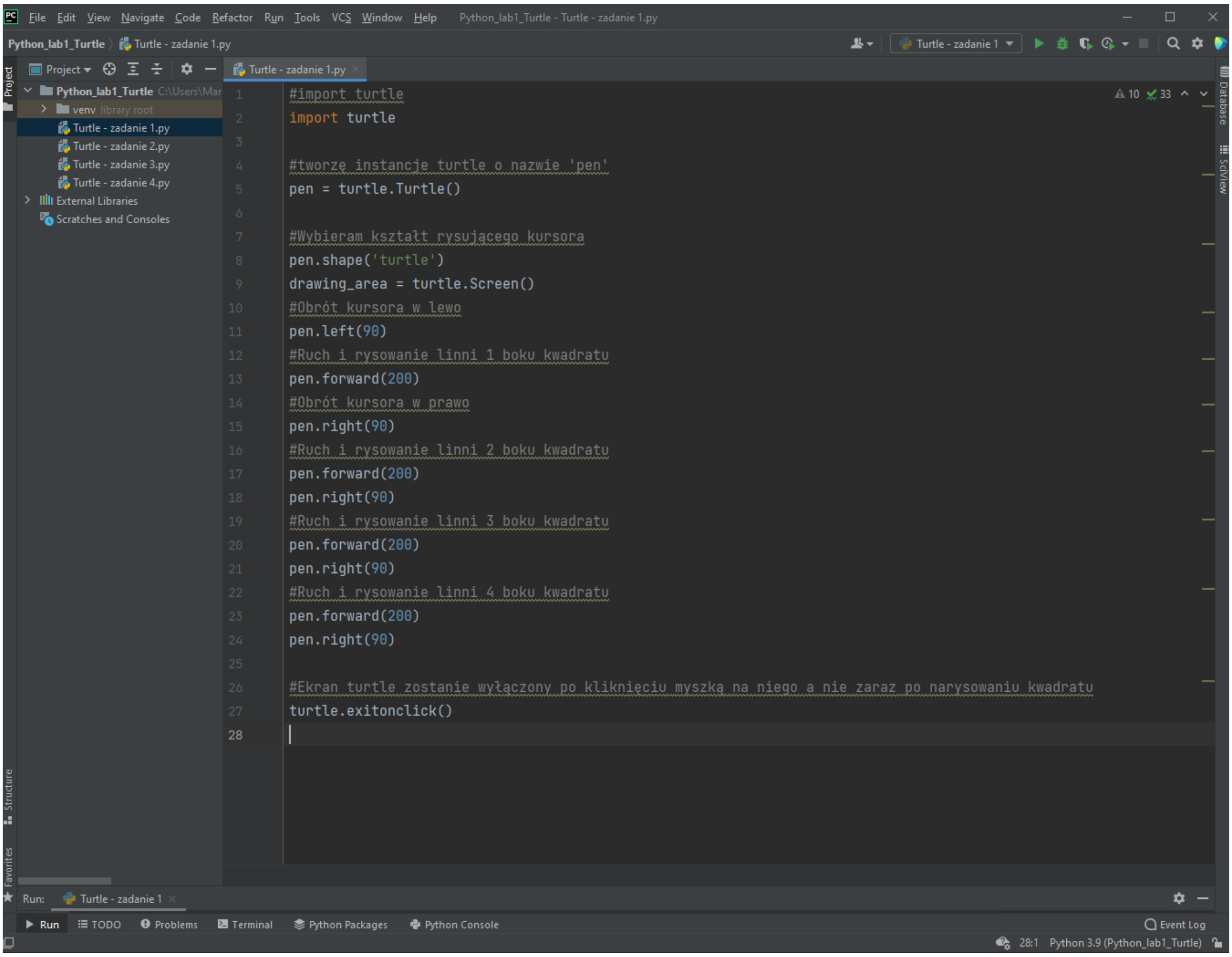
Języki skryptowe

Laboratorium - Python 1 - Turtle

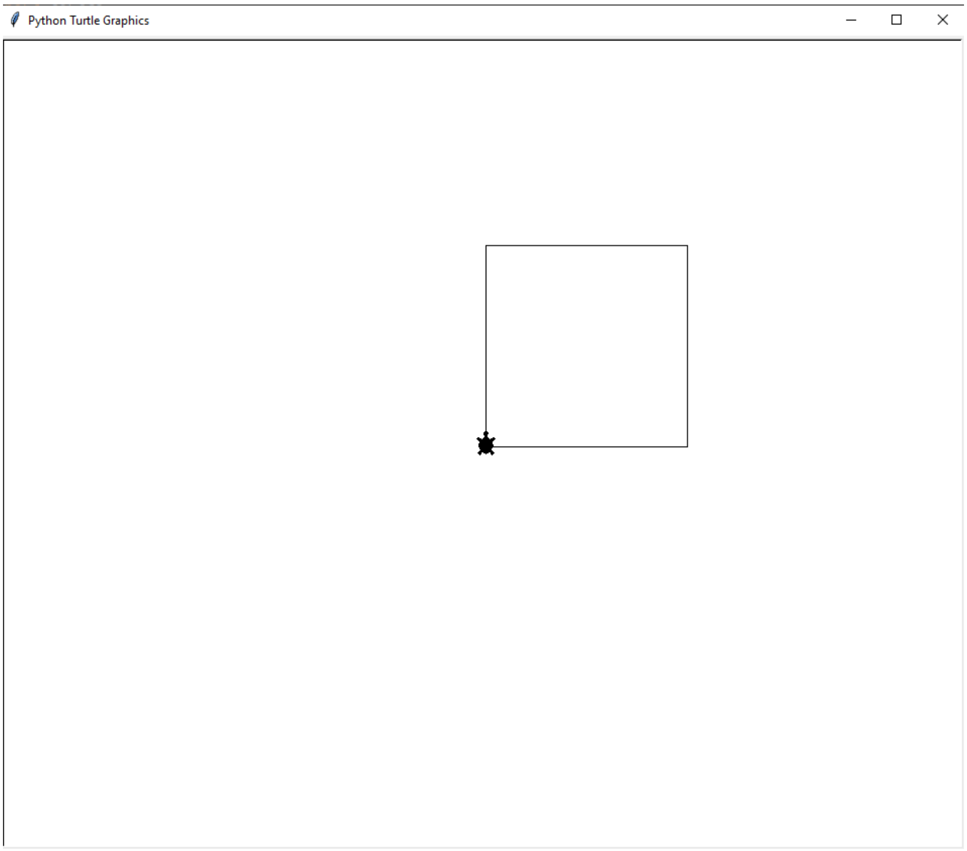
CIESZYN, 29 KWIECIEŃ 2022

Zadanie 1

Napisz program zawierający instrukcję rysowania kwadratu.

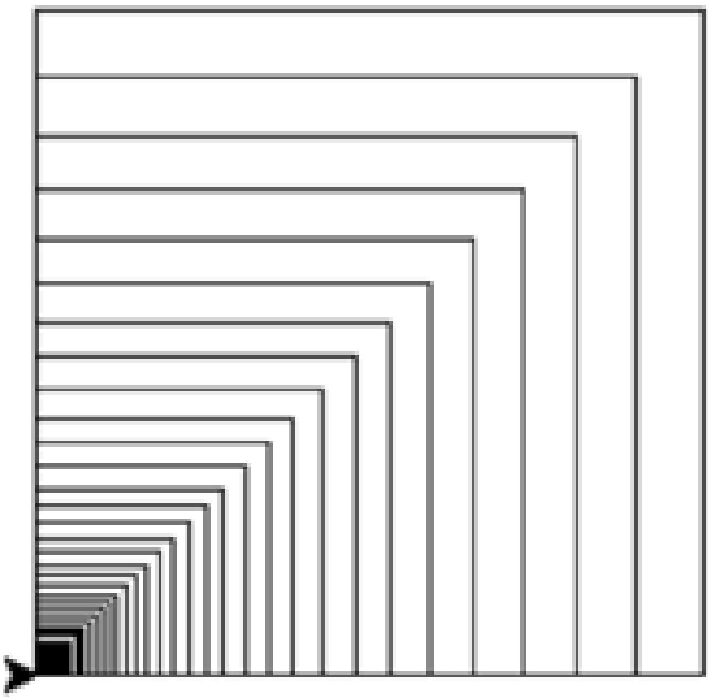


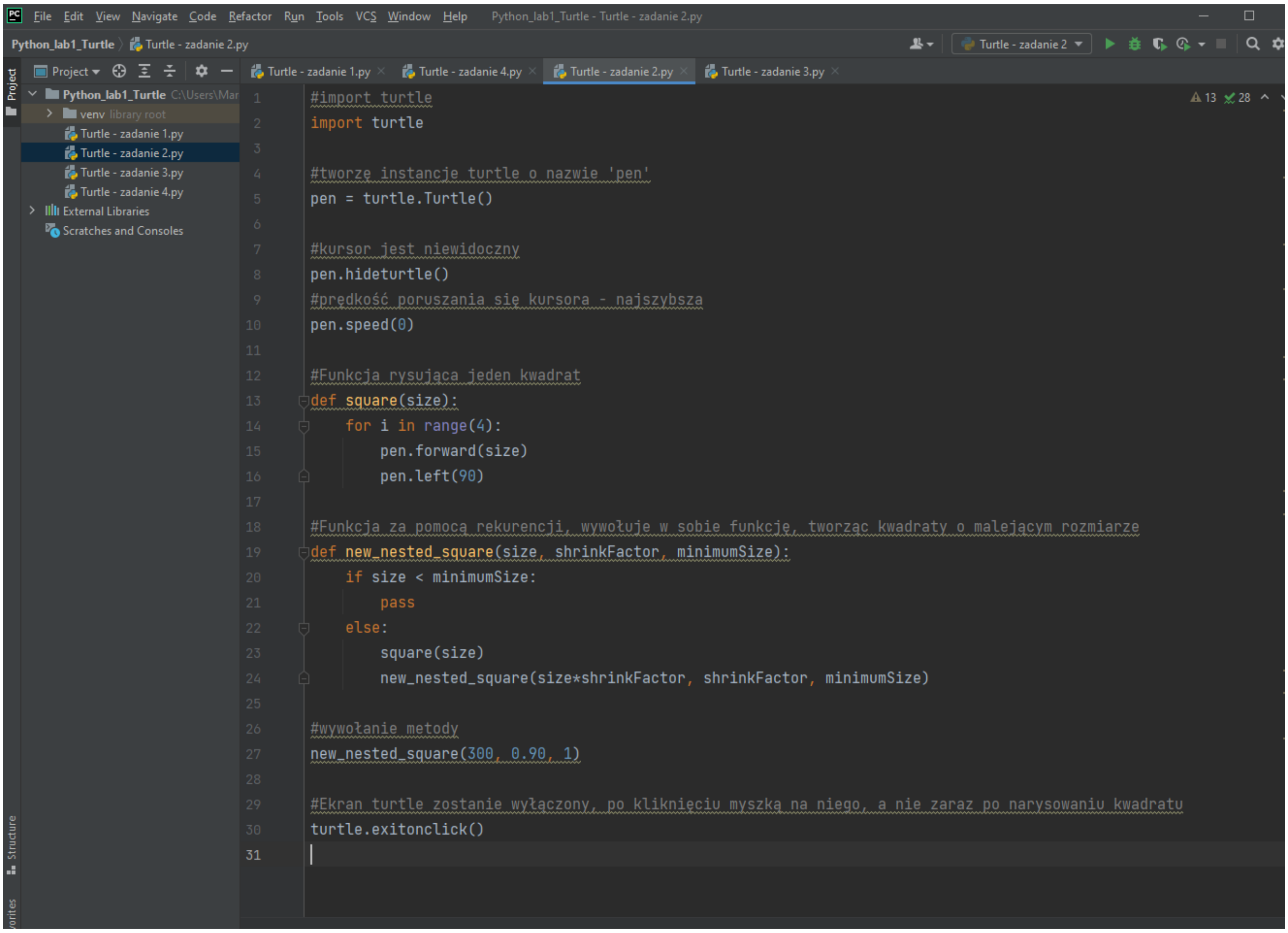
```python
#import turtle
import turtle

#tworze instancje turtle o nazwie 'pen'
pen = turtle.Turtle()

#Wybieram kształt rysującego kursora
pen.shape('turtle')
drawing_area = turtle.Screen()
#Obrót kursora w lewo
pen.left(90)
#Ruch i rysowanie linni 1 boku kwadratu
pen.forward(200)
#Obrót kursora w prawo
pen.right(90)
#Ruch i rysowanie linni 2 boku kwadratu
pen.forward(200)
pen.right(90)
#Ruch i rysowanie linni 3 boku kwadratu
pen.forward(200)
pen.right(90)
#Ruch i rysowanie linni 4 boku kwadratu
pen.forward(200)
pen.right(90)

#Ekran turtle zostanie wyłączony po kliknięciu myszką na niego a nie zaraz po narysowaniu kwadratu
turtle.exitonclick()

```



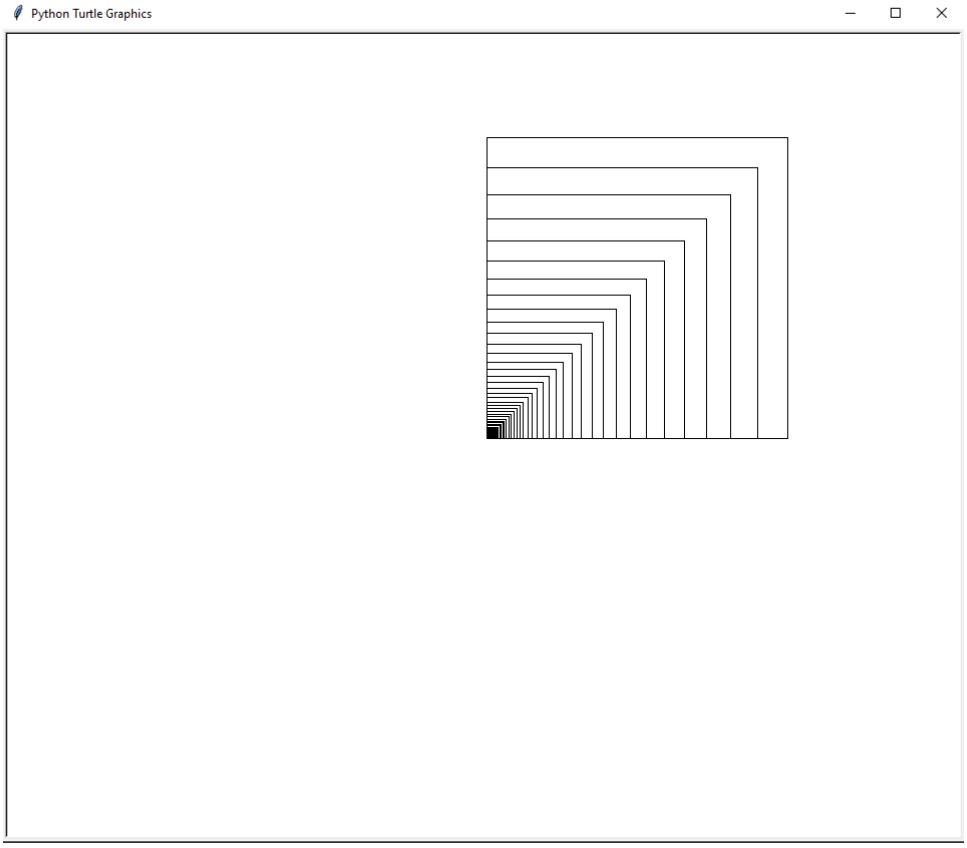
Zadanie 2

Narysuj poniższy fraktal:



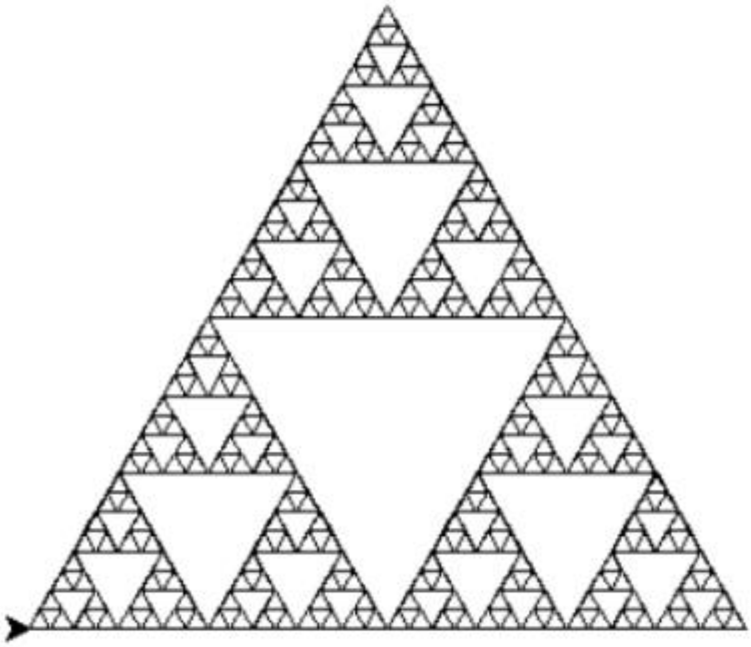


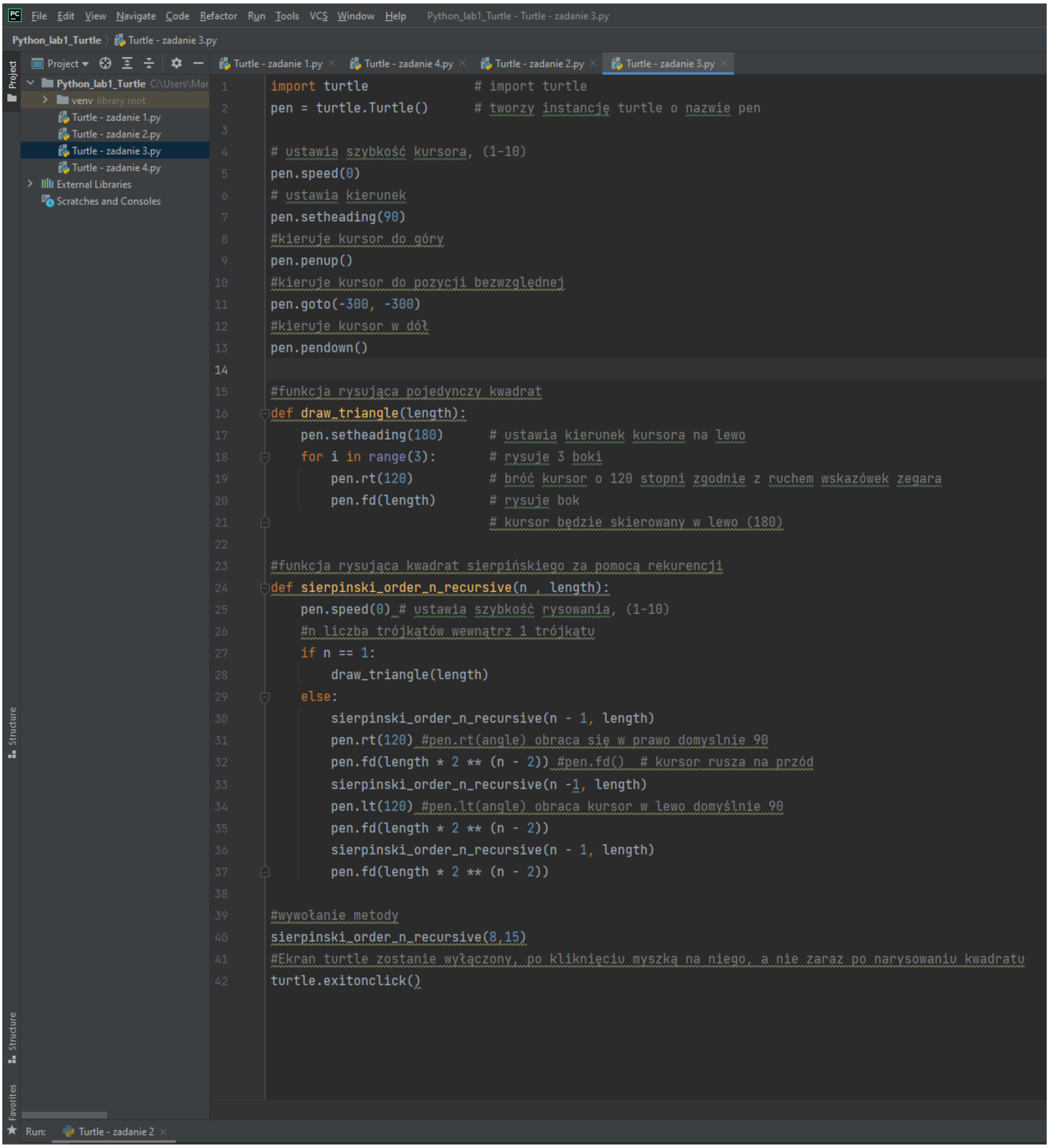
```python
#import turtle
import turtle

#tworzę instancję turtle o nazwie 'pen'
pen = turtle.Turtle()

#kursor jest niewidoczny
pen.hideturtle()
#prędkość poruszania się kursora - najszybsza
pen.speed(0)

#Funkcja rysująca jeden kwadrat
def square(size):
    for i in range(4):
        pen.forward(size)
        pen.left(90)

#Funkcja za pomocą rekurencji, wywołuje w sobie funkcję, tworząc kwadraty o malejącym rozmiarze
def new_nested_square(size, shrinkFactor, minimumSize):
    if size < minimumSize:
        pass
    else:
        square(size)
        new_nested_square(size*shrinkFactor, shrinkFactor, minimumSize)

#wywołanie metody
new_nested_square(300, 0.90, 1)

#Ekran turtle zostanie wyłączony, po kliknięciu myszką na niego, a nie zaraz po narysowaniu kwadratu
turtle.exitonclick()
```



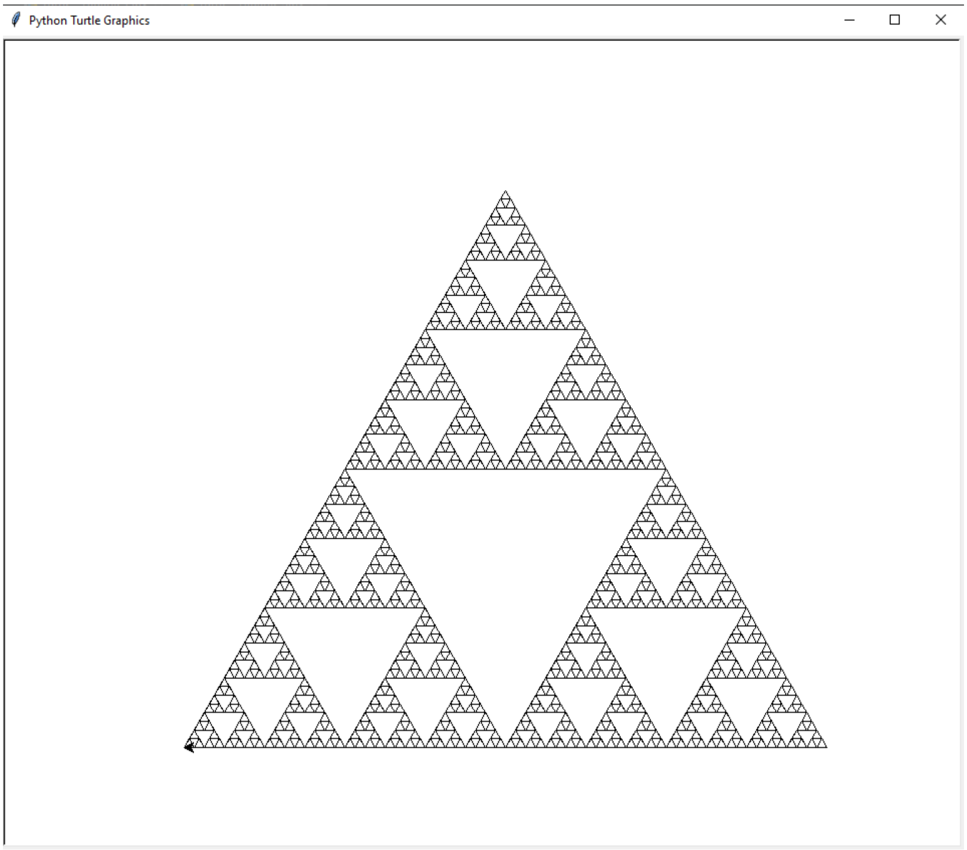
2.3 Zadanie 3

Narysuj poniższy fraktal:





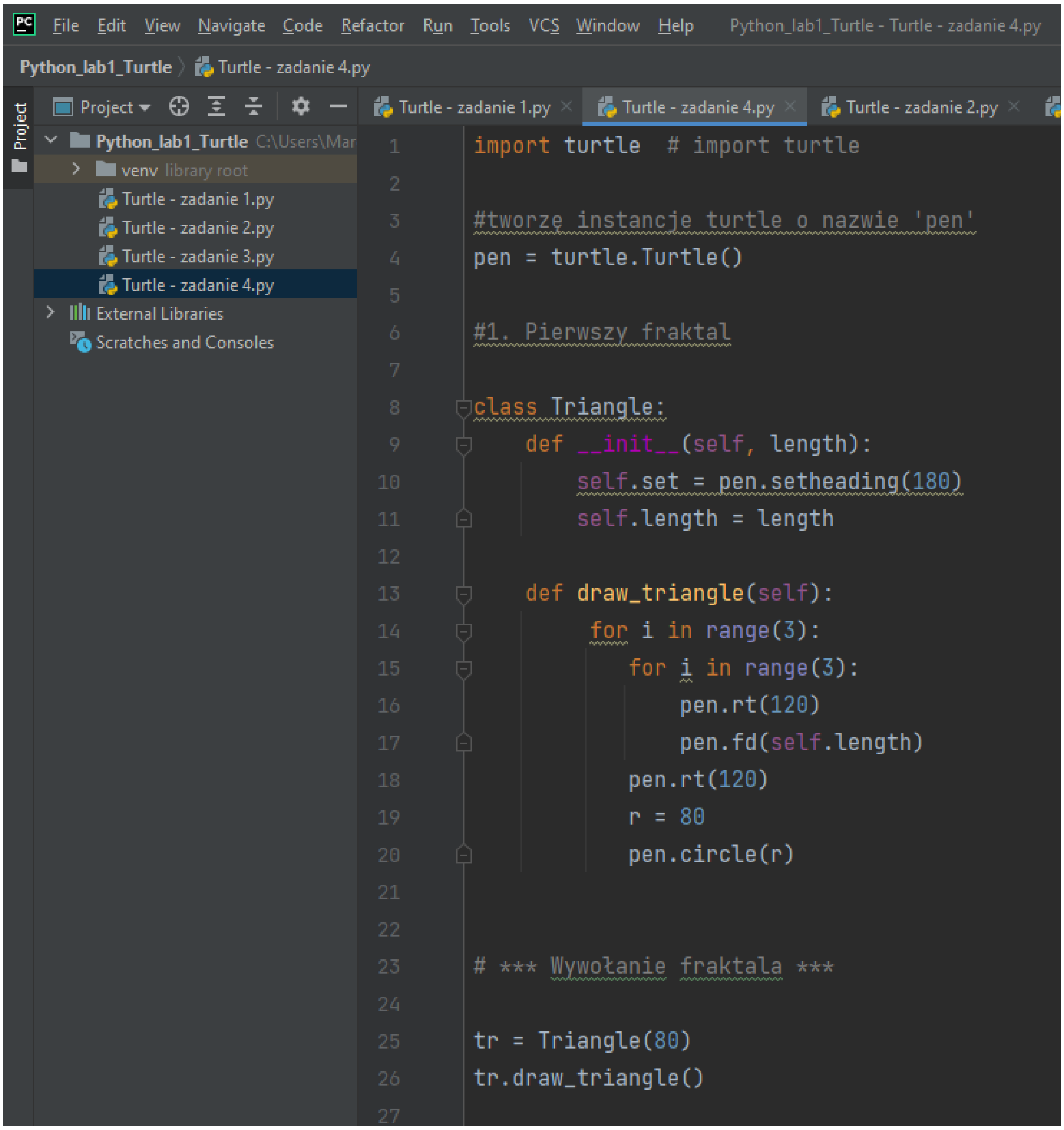
```python
import turtle              # import turtle
pen = turtle.Turtle()      # tworzy instancję turtle o nazwie pen

# ustawia szybkość kursora, (1-10)
pen.speed(0)
# ustawia kierunek
pen.setheading(90)
#kieruje kursor do góry
pen.penup()
#kieruje kursor do pozycji bezwzględnej
pen.goto(-300, -300)
#kieruje kursor w dół
pen.pendown()

#funkcja rysująca pojedynczy kwadrat
def draw_triangle(length):
    pen.setheading(180)      # ustawia kierunek kursora na lewo
    for i in range(3):       # rysuje 3 boki
        pen.rt(120)          # bróć kursor o 120 stopni zgodnie z ruchem wskazówek zegara
        pen.fd(length)       # rysuje bok
                             # kursor będzie skierowany w lewo (180)

#funkcja rysująca kwadrat sierpińskiego za pomocą rekurencji
def sierpinski_order_n_recursive(n , length):
    pen.speed(0) # ustawia szybkość rysowania, (1-10)
    #n liczba trójkątów wewnątrz 1 trójkątu
    if n == 1:
        draw_triangle(length)
    else:
        sierpinski_order_n_recursive(n - 1, length)
        pen.rt(120) #pen.rt(angle) obraca się w prawo domyslnie 90
        pen.fd(length * 2 ** (n - 2)) #pen.fd()  # kursor rusza na przód
        sierpinski_order_n_recursive(n -1, length)
        pen.lt(120) #pen.lt(angle) obraca kursor w lewo domyślnie 90
        pen.fd(length * 2 ** (n - 2))
        sierpinski_order_n_recursive(n - 1, length)
        pen.fd(length * 2 ** (n - 2))

#wywołanie metody
sierpinski_order_n_recursive(8,15)
#Ekran turtle zostanie wyłączony, po kliknięciu myszką na niego, a nie zaraz po narysowaniu kwadratu
turtle.exitonclick()
```



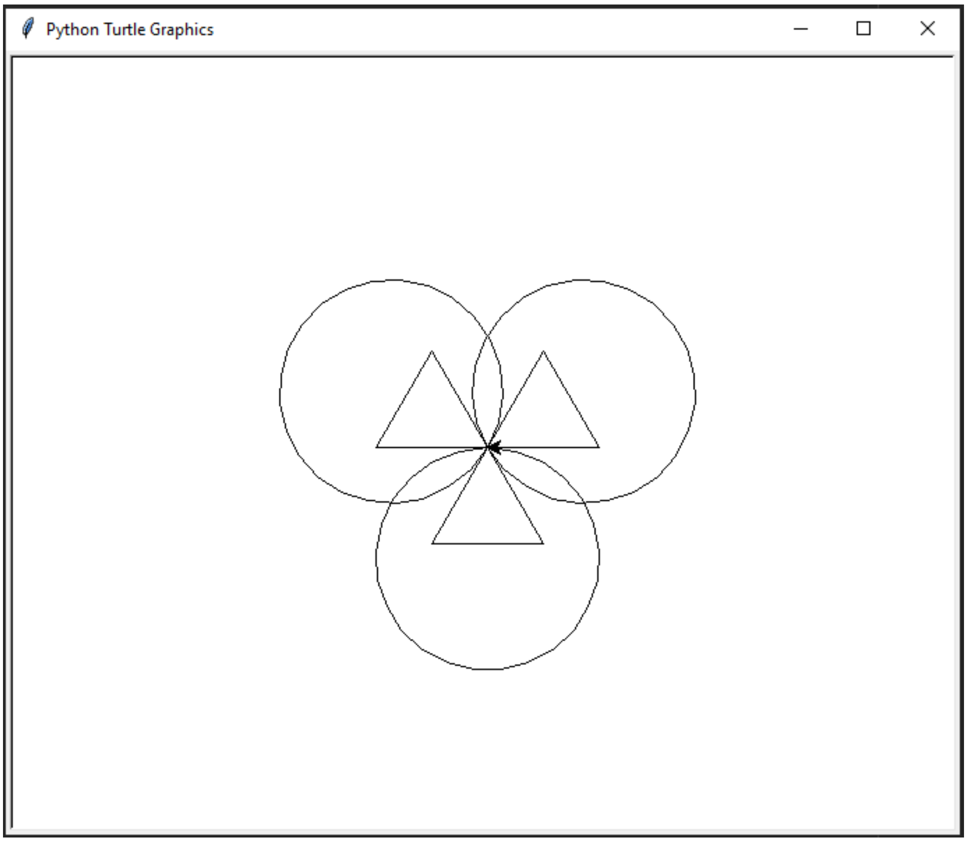
2.4 Zadanie 4

Narysuj 4 dowolne fraktale. W programach wykorzystaj klasy oraz metody klas do generowania fraktali.

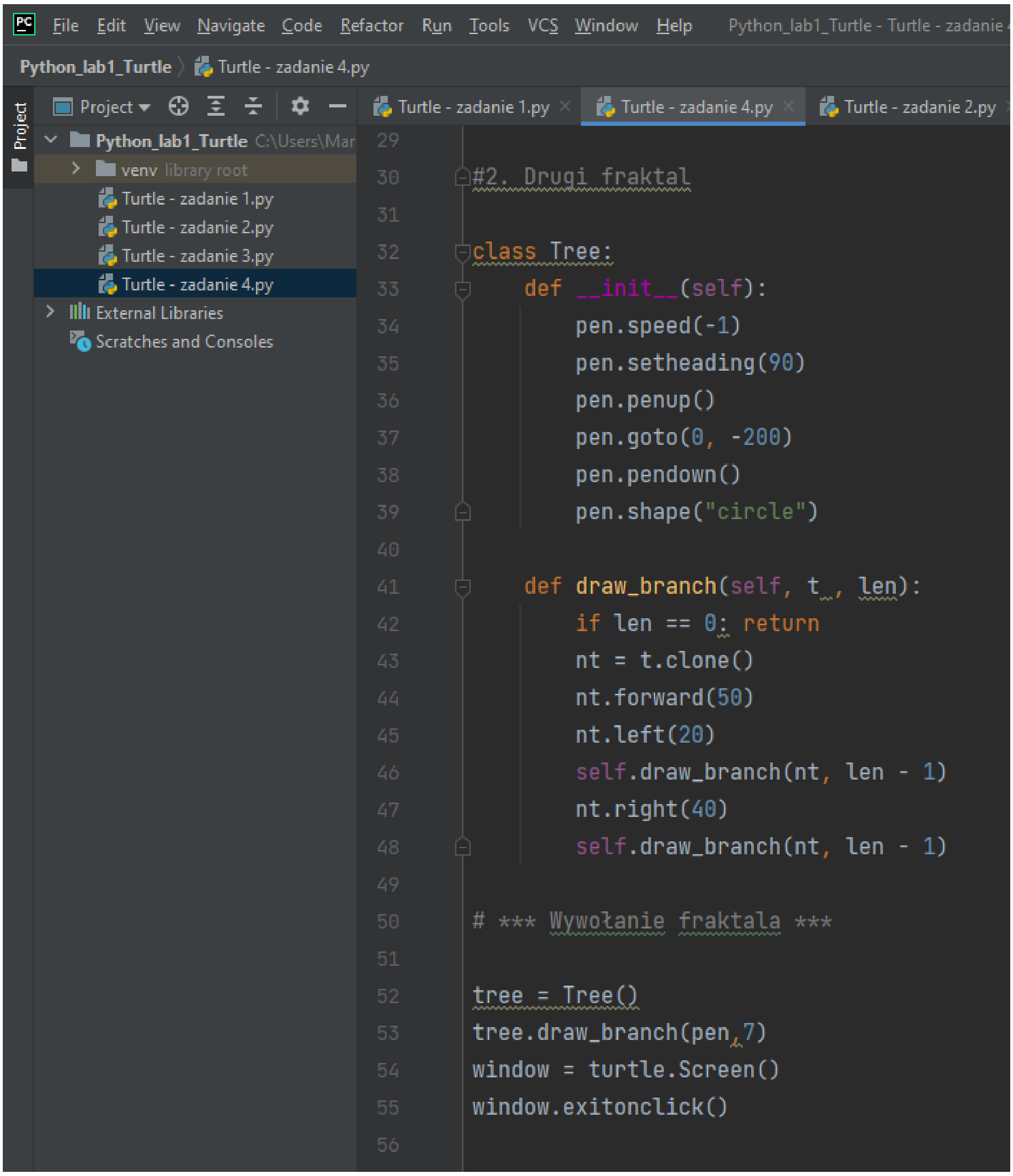
Pierwszy fraktal:



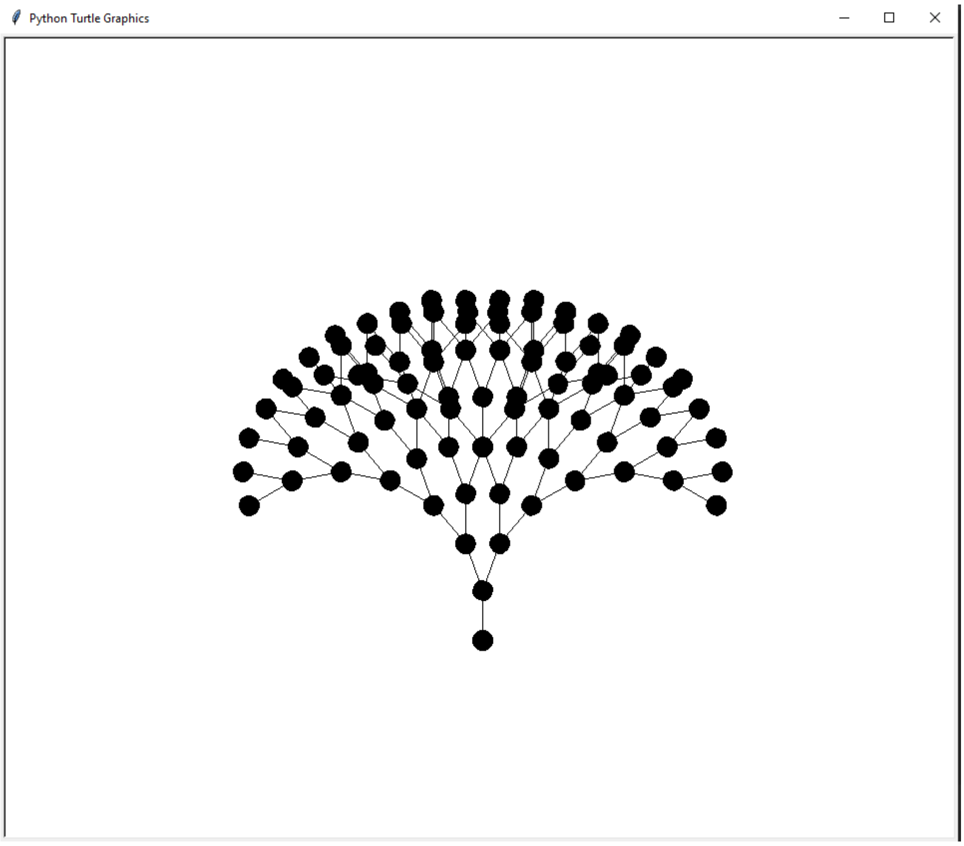
```python
import turtle  # import turtle

#tworze instancje turtle o nazwie 'pen'
pen = turtle.Turtle()

#1. Pierwszy fraktal

class Triangle:
    def __init__(self, length):
        self.set = pen.setheading(180)
        self.length = length

    def draw_triangle(self):
         for i in range(3):
            for i in range(3):
                pen.rt(120)
                pen.fd(self.length)
            pen.rt(120)
            r = 80
            pen.circle(r)


# *** Wywołanie fraktala ***

tr = Triangle(80)
tr.draw_triangle()
```



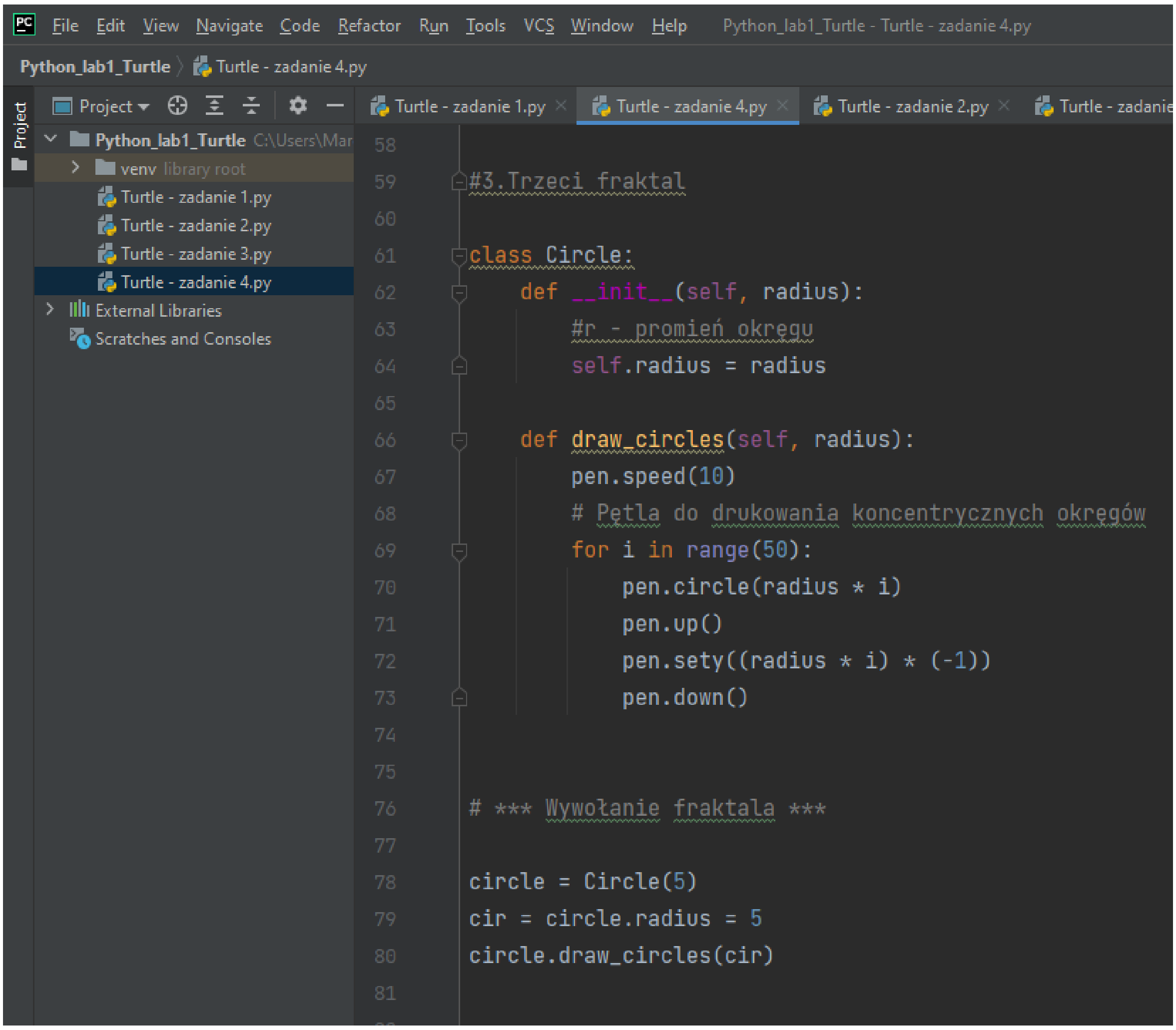
Drugi fraktal



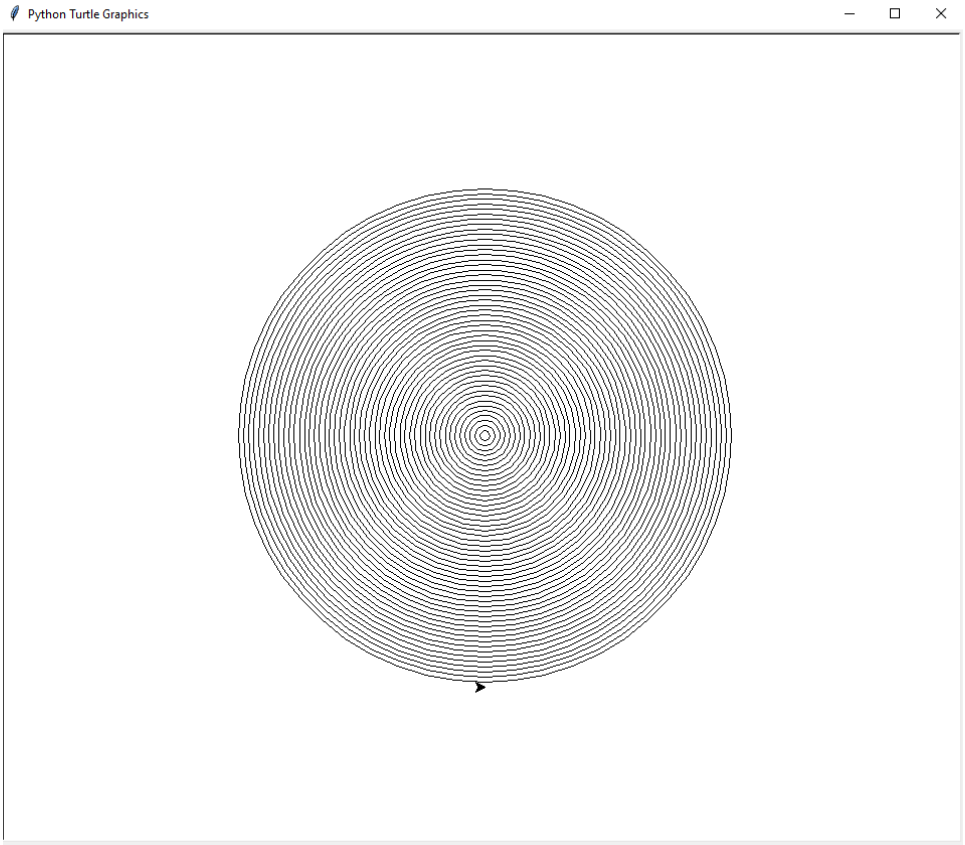
```

#2. Drugi fraktal

class Tree:
    def __init__(self):
        pen.speed(-1)
        pen.setheading(90)
        pen.penup()
        pen.goto(0, -200)
        pen.pendown()
        pen.shape("circle")

    def draw_branch(self, t , len):
        if len == 0: return
        nt = t.clone()
        nt.forward(50)
        nt.left(20)
        self.draw_branch(nt, len - 1)
        nt.right(40)
        self.draw_branch(nt, len - 1)

# *** Wywołanie fraktala ***

tree = Tree()
tree.draw_branch(pen,7)
window = turtle.Screen()
window.exitonclick()

```



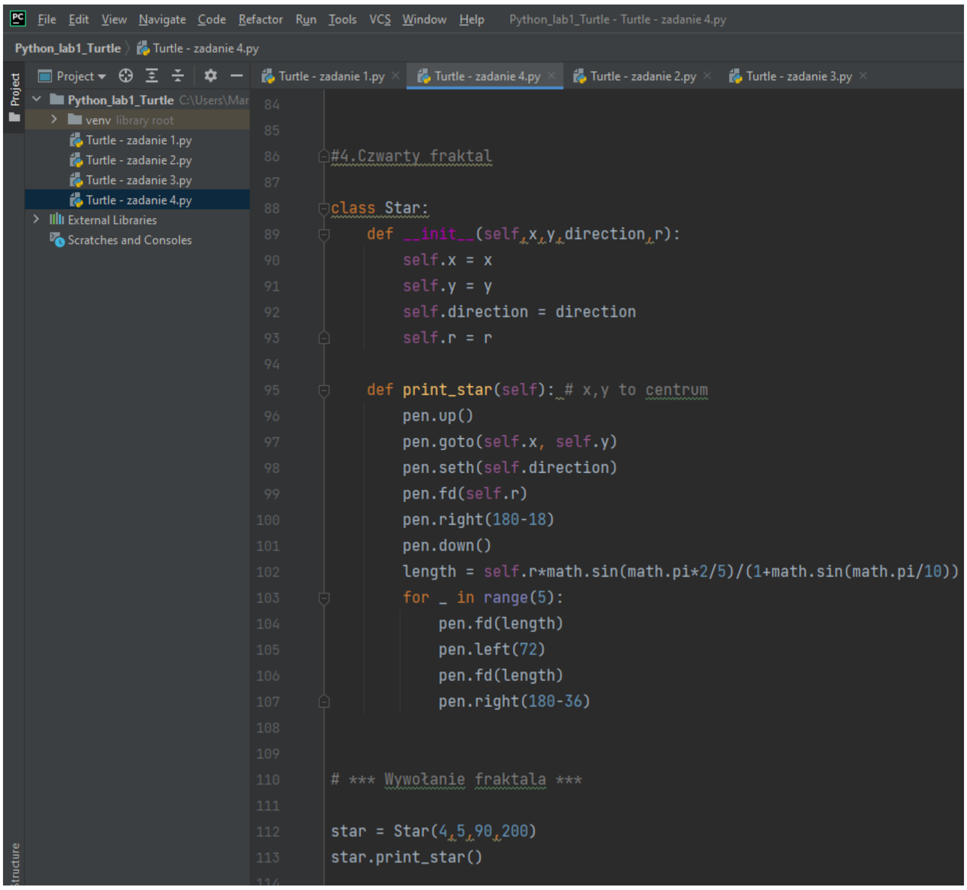
Trzeci fraktal:



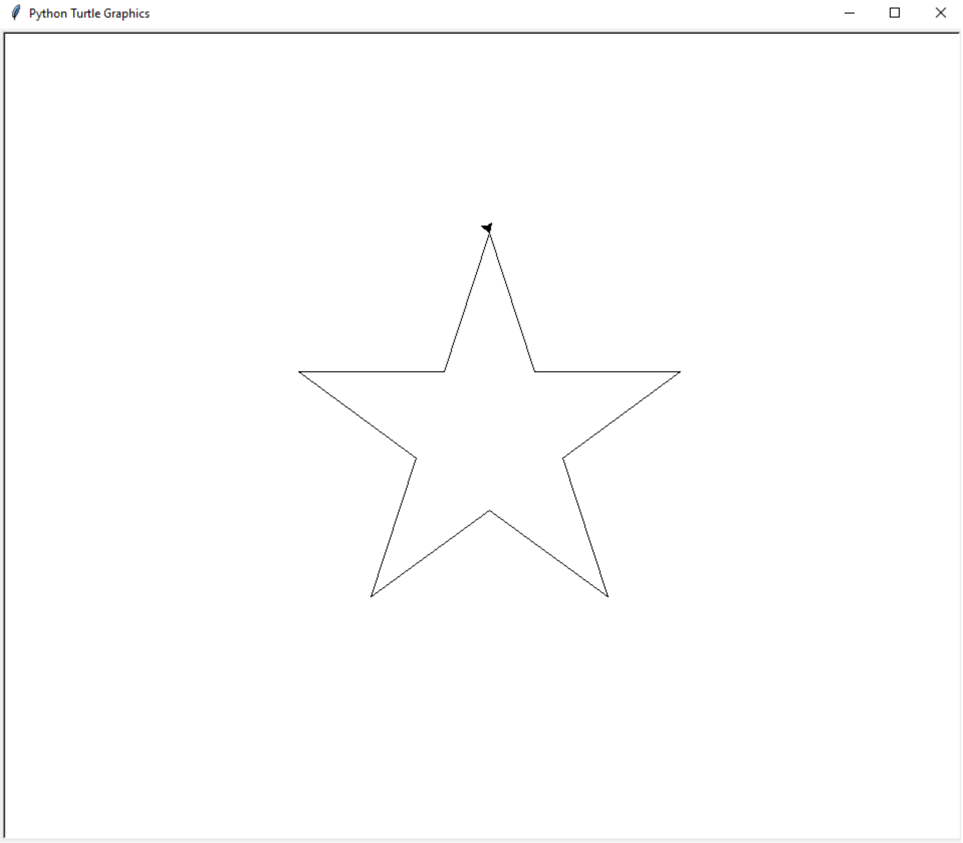
```python

#3.Trzeci fraktal

class Circle:
    def __init__(self, radius):
        #r - promień okręgu
        self.radius = radius

    def draw_circles(self, radius):
        pen.speed(10)
        # Pętla do drukowania koncentrycznych okręgów
        for i in range(50):
            pen.circle(radius * i)
            pen.up()
            pen.sety((radius * i) * (-1))
            pen.down()


# *** Wywołanie fraktala ***

circle = Circle(5)
cir = circle.radius = 5
circle.draw_circles(cir)

```



Czwarty fraktal:



```python


#4.Czwarty fraktal

class Star:
    def __init__(self,x,y,direction,r):
        self.x = x
        self.y = y
        self.direction = direction
        self.r = r

    def print_star(self): # x,y to centrum
        pen.up()
        pen.goto(self.x, self.y)
        pen.seth(self.direction)
        pen.fd(self.r)
        pen.right(180-18)
        pen.down()
        length = self.r*math.sin(math.pi*2/5)/(1+math.sin(math.pi/10))
        for _ in range(5):
            pen.fd(length)
            pen.left(72)
            pen.fd(length)
            pen.right(180-36)


# *** Wywołanie fraktala ***

star = Star(4,5,90,200)
star.print_star()
```


Python Turtle Graphics