**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Институт физики

Направление подготовки: 10.03.05 – информационная безопасность автоматизированных систем

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

Цифровая обработка изображений:

Фрактальные изображения

Студент 3 курса

группы 06-952 Глазков А.Ю.

Научный руководитель Корчагин П.А.

Казань – 2022

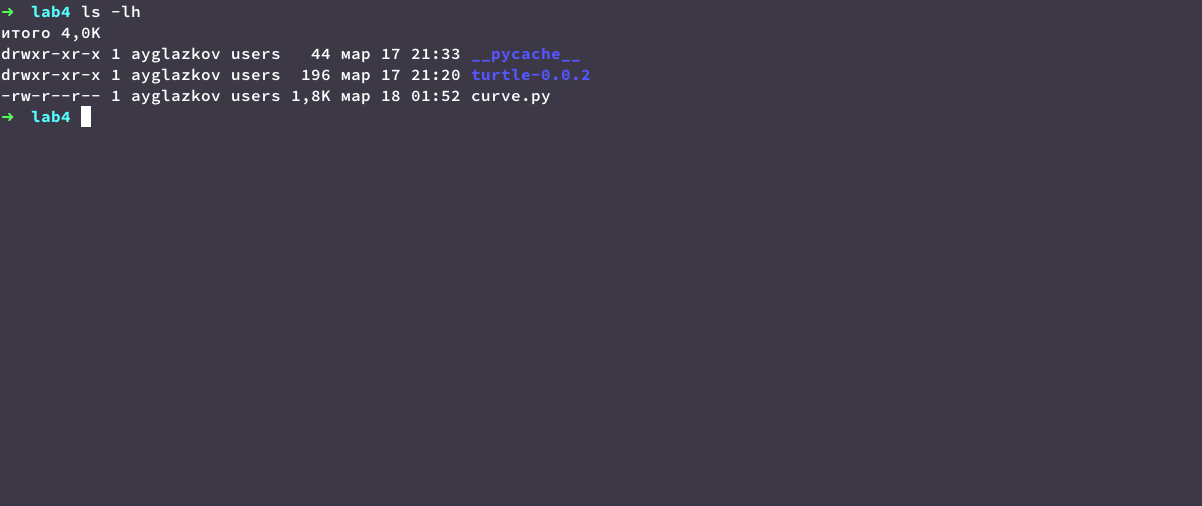
# Лабораторная работа №4

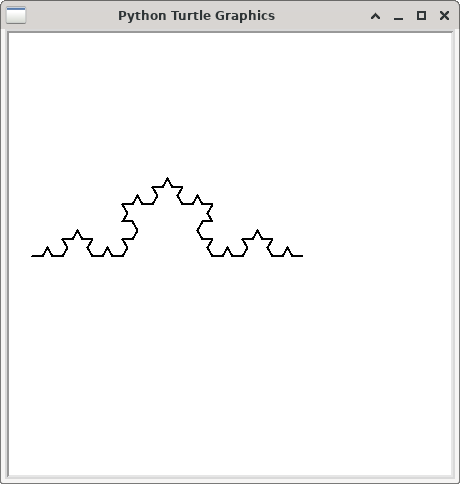
## Построение кривой Коха и расчет фрактальной размерности

Цель: Разработать приложение для построения кривой Коха и расчета ее фрактальной размерности по покрытию.

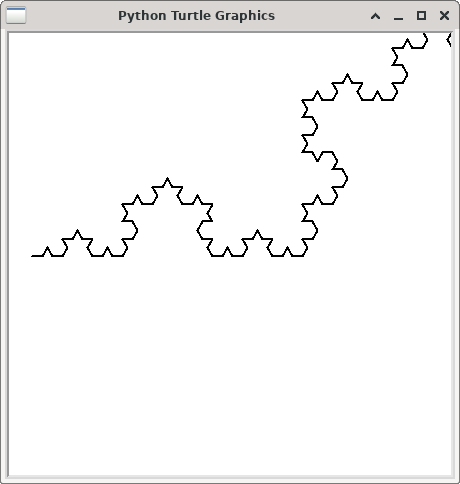
Использованный язык программирования: python.

Ход работы:

1. Выводим содержимое папки до исполнения программы:
2. Запустим разработанное приложение с параметром n = 3:  
   
3. При помощи черепашки строится фрактальное изображение кривой Коха и сохраняется в файл “turtle\_img.png”:



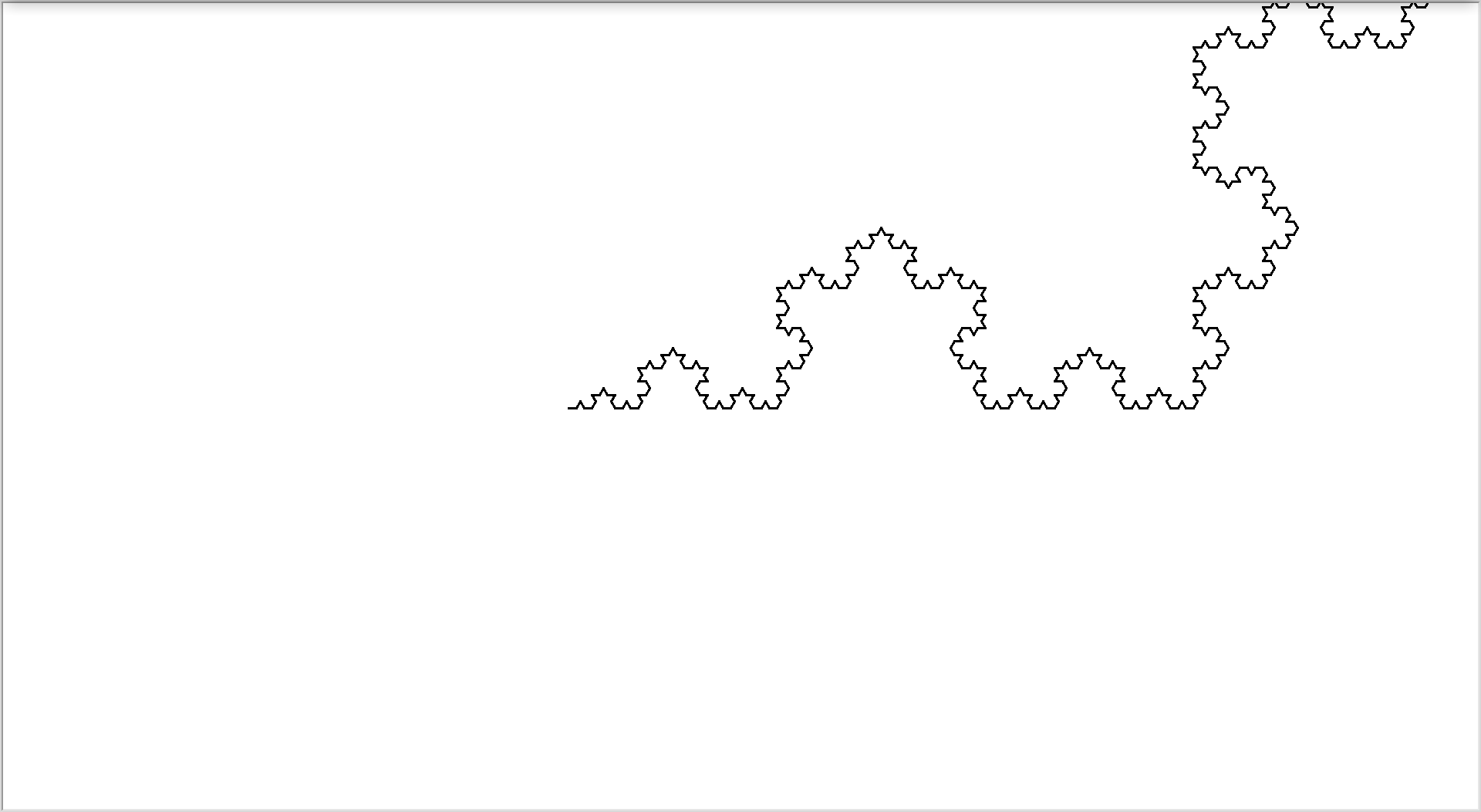
1. После закрытия приложения в терминале появляется информация о фрактальной размерности по покрытию:  
    
2. Запустим разработанное приложение с параметром n = 4:



1. После закрытия приложения в терминале появляется информация о фрактальной размерности по покрытию:



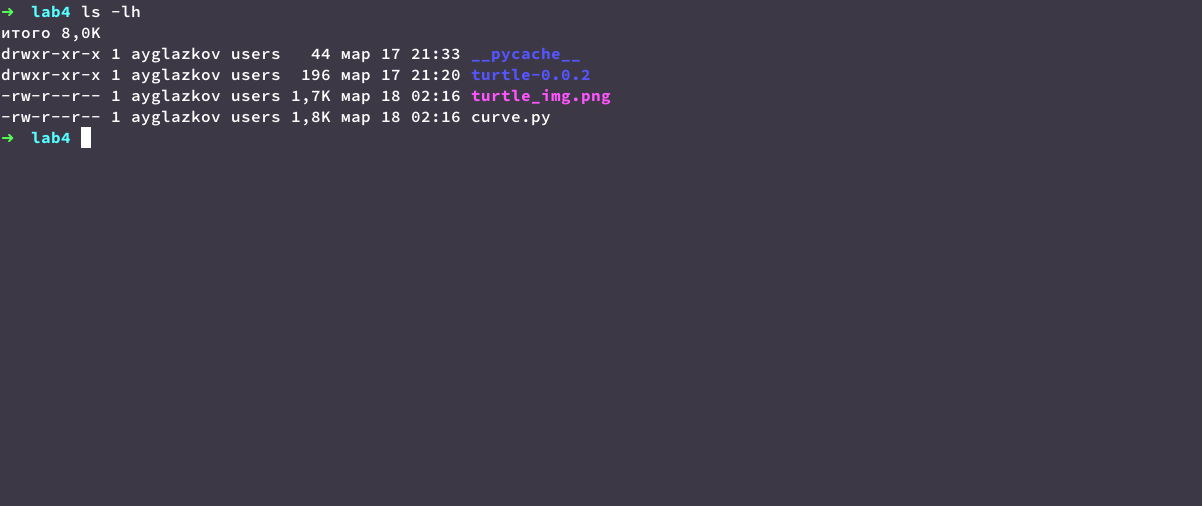
1. Запустим разработанное приложение с параметром n = 5:



1. После закрытия приложения в терминале появляется информация о фрактальной размерности по покрытию:



1. Просмотрим содержимое директории после отработки приложения:



## 

## Листинг написанной программы:

import turtle

import numpy as np

from PIL import Image

import cv2

import os

def get\_threshold(Z: np.ndarray):

return Z.mean()

def boxcount(Z: np.ndarray, k: int):

S = np.add.reduceat(

np.add.reduceat(Z, np.arange(0, Z.shape[0], k), axis=0),

np.arange(0, Z.shape[1], k), axis=1)

return len(np.where((S > 0) & (S < k \* k))[0])

def ft\_fractalDimension(Z: np.ndarray):

assert (len(Z.shape) == 2)

Z = (Z < get\_threshold(Z))

p = min(Z.shape)

n = 2 \*\* (np.log(p) / np.log(2))

n = int(np.log(n) / np.log(2))

sizes = 2 \*\* np.arange(n, 1, -1)

counts = []

for size in sizes:

counts.append(boxcount(Z, size))

coeffs = np.polyfit(np.log(sizes), np.log(counts), 1)

return -coeffs[0]

def ft\_kochTurns(n):

route = []

for i in range(n):

for j in range(4\*\*i):

route.insert(j\*4, 60)

route.insert(j\*4, -120)

route.insert(j\*4, 60)

return route

def ft\_turtleKoch(n, line\_lenght=10, width=450,

height=450, fileName = "turtle\_img"):

wn = turtle.Screen()

wn.setup(width, height)

turtle.setx(-200)

turtle.clear()

turtle.pensize(2)

turtle.speed(0)

for move in ft\_kochTurns(n):

turtle.forward(line\_lenght)

turtle.left(move)

turtle.forward(line\_lenght)

turtle.hideturtle()

canvas = wn.getcanvas()

canvas.postscript(file= fileName+'.eps', width=width, height=height)

img = Image.open(fileName + '.eps')

img.save(fileName + '.png')

wn.exitonclick()

os.remove(fileName+'.eps')

def main():

n = int(input("n = "))

ft\_turtleKoch(n)

image = cv2.imread('turtle\_img.png', 0)

fd = np.around(ft\_fractalDimension(image), decimals=4)

print(f"Фрактальная размерность: {fd}")

main()

# 

# Вывод

В ходе выполненной лабораторной работы было разработано приложение, позволяющий построить фрактальное изображение кривой Коха и выполнить расчет её фрактальной размерности по полученному изображению. В результате получили фрактальную размерность для кривой Коха, стремящуюся к 1.2614 – что соответствует теории. Кривая Коха имеет промежуточную (то есть не целую) хаусдорфову размерность, которая равна ln ⁡(4) / ln(3) ≈ 1.26, поскольку она состоит из четырёх равных частей, каждая из которых подобна всей кривой с коэффициентом подобия 1/3.