МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А.И. ГЕРЦЕНА»

Институт компьютерных наук и технологического образования

Кафедра компьютерных технологий и электронного обучения

КУРСОВАЯ РАБОТА

ФРАКТАЛЫ. ИССЛЕДОВАНИЕ И МОДЕЛИРОВАНИЕ

Направление подготовки: «Технологии компьютерного моделирования»

.

Руководитель:
Доктор педагогических наук, профессор,
_____________________ Власова Е.З.
«___»_______________ 2019 г.

Автор работы:
Студент 2 курса группы ИВТ
_____________________ Ефимова В.С.
«___»_______________ 2019 г.

Санкт-Петербург
2019

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Данная работа посвящена исследованию фракталов, как математических объектов и явлений природы, определению их области применения и моделированию фракталов средствами языка программирования Python.

Актуальность темы заключается в следующем. Евклидова геометрия описывает такие простые фигуры как линия, круг, треугольник, квадрат, сфера, конус и так далее. С помощью этих фигур ученые описывают многие процессы и применяют их для проектирования мира вокруг нас. Однако эта геометрия не может описать сложные структуры и формы природы. Так, облака не являются сферами, а горы - конусами. Моделирование кровеносной системы человека, фактуры коры дерева, природных ландшафтов инструментами евклидовой геометрии невозможно. В то время как фрактальная геометрия способна описать явления природы, а также применима в науке, компьютерных технологиях, медицине, экономике и многих других областях.

Объектом исследования являются фракталы.

Предмет исследования – фракталы.

Цель работы – рассмотреть особенности фракталов, их область применения и разработать программный продукт, генерирующий фракталы.

Для достижения указанной цели в курсовой работе решаются следующие **исследовательские задачи**:

1. Рассмотреть примеры самоподобных множеств с необычными свойствами в математике (Множество Мандельброта, Ковер Серпинского, Кривая Коха, Кривая Пеано).
2. Рассмотреть природные объекты, обладающее фрактальными свойствами.
3. Проанализировать применение фракталов.

Структура исследования. Курсовая работа включает в себя введение, 2 раздела, в которых решаются поставленные исследовательские задачи, заключение, список источников и литературы.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Понятие «фрактал».

Слово “фрактал” образовано от латинского причастия fractus. Соответствующий глагол frangere переводится, как ломать, разламывать, то есть создавать фрагменты неправильной формы. Его предложил Бенуа Мандельброт в 1977 г. в своей книге “Фрактальная геометрия природы”. Этим словом он описывал множества, обладающие свойством самоподобия.

Самоподобный объект – объект, в точности или приближённо совпадающий с частью себя самого, то есть целое имеет ту же форму, что и одна или более частей. Таким образом, небольшая часть фрактала содержит информацию о всем фрактале.

Слово «фрактал» употребляется не только в качестве математического термина. Фракталом может называться предмет, обладающий, по крайней мере, одним из указанных ниже свойств:

1. Обладает нетривиальной структурой на всех масштабах. В этом отличие от регулярных фигур (таких как окружность, эллипс, график гладкой функции): если рассмотреть небольшой фрагмент регулярной фигуры в очень крупном масштабе, то он будет похож на фрагмент прямой. Для фрактала увеличение масштаба не ведёт к упрощению структуры, то есть на всех шкалах можно увидеть одинаково сложную картину.
2. Является самоподобным или приближённо самоподобным.
3. Обладает дробной метрической размерностью или метрической размерностью, превосходящей топологическую.

Самоподобные фигуры, повторяющиеся конечное число раз, называются предфракталами.

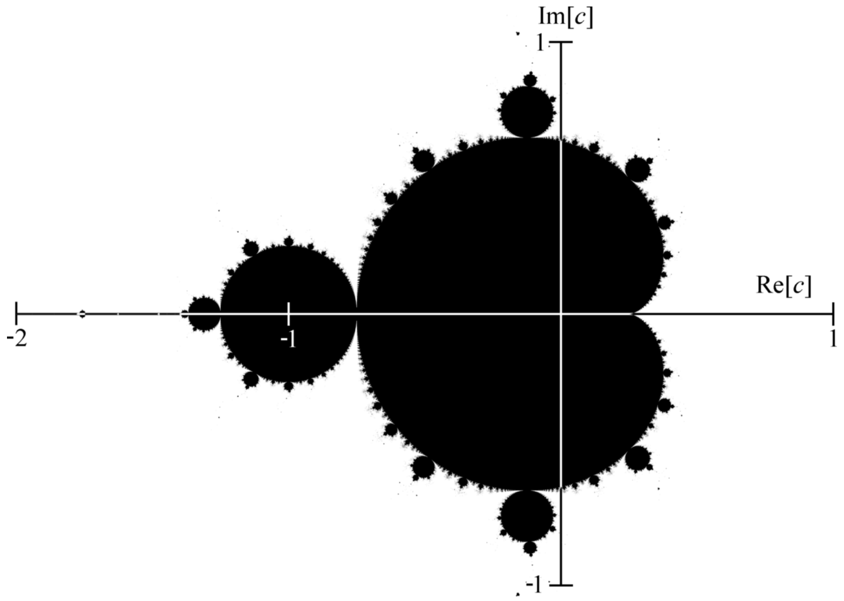
Многие объекты в природе обладают свойствами фрактала, например: побережья, облака, кроны деревьев, снежинки, система кровообращения, альвеолы.

Примеры фракталов.

Множество Мандельброта.

Множество Мандельброта является одним из самых известных фракталов, в том числе за пределами математики, благодаря своим цветным визуализациям. Его фрагменты не строго подобны исходному множеству, но при многократном увеличении определённые части всё больше похожи друг на друга.

Множество Мандельброта — это множество точек c на комплексной плоскости, для которых последовательность z_n, определяемая итерациями $z_0 = 0$, $z_1 = z_0^2 + c$, ..., $z_{n+1} = z_n^2 + c$, конечна (то есть не уходит в бесконечность). Визуально, внутри множества Мандельброта можно выделить бесконечное количество элементарных фигур, причём самая большая в центре представляет собой кардиоиду (она похожа на стилизованное изображение сердца и получила свое название от двух греческих слов — «сердце» и «вид»). Также есть набор овалов, касающихся кардиоиды, размер которых постепенно уменьшается, стремясь к нулю. Каждый из этих овалов имеет свой набор меньших овалов, диаметр которых также стремится к нулю и т. д. Этот процесс продолжается бесконечно, образуя фрактал. Также важно, что эти процессы ветвления фигур не исчерпывают полностью множество Мандельброта: если рассмотреть с увеличением дополнительные «ветки», то в них можно увидеть свои кардиоиды и круги, не связанные с главной фигурой. Самая большая фигура (видимая при рассматривании основного множества) из них находится в области от −1,78 до −1,75 на отрицательной оси действительных значений.



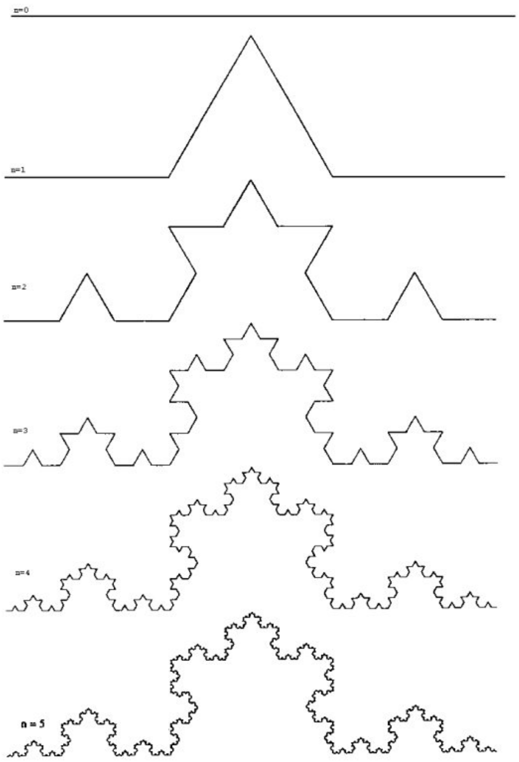
Кривая Коха (Снежинка Коха).

Снежинка Коха, построенная в виде замкнутой кривой на базе равностороннего треугольника, впервые была описана шведским математиком Хельге фон Кохом в 1904 году.

Кривая Коха является типичным геометрическим фракталом. Процесс её построения выглядит следующим образом: берём единичный отрезок, разделяем на три равные части и заменяем средний интервал равносторонним треугольником без этого сегмента. В результате образуется ломаная, состоящая из четырёх звеньев длины 1/3. На следующем шаге повторяем операцию для каждого из четырёх получившихся звеньев и т. д. Предельная кривая и есть кривая Коха.

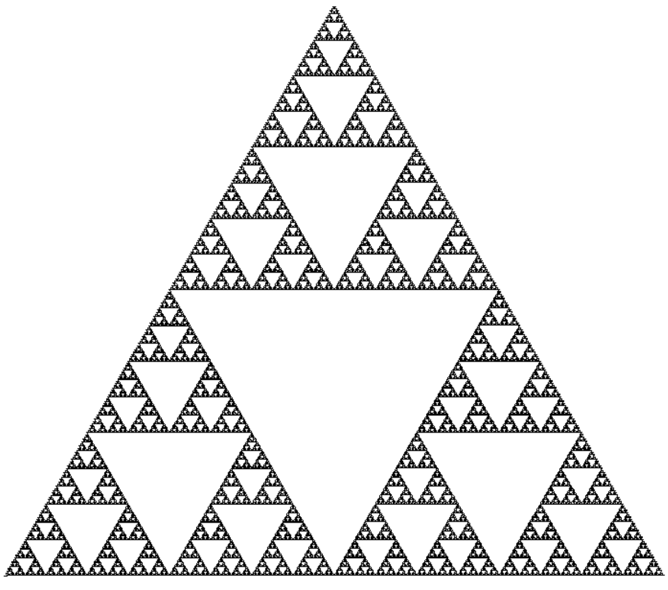
Свойства:

1. Кривая Коха нигде не дифференцируема и не спрямляема.
2. Кривая Коха имеет бесконечную длину.
3. Кривая Коха не имеет самопересечений.
4. Плоскость допускает замощение снежинками Коха двух размеров. При этом не существует замощения снежинками одного размера.

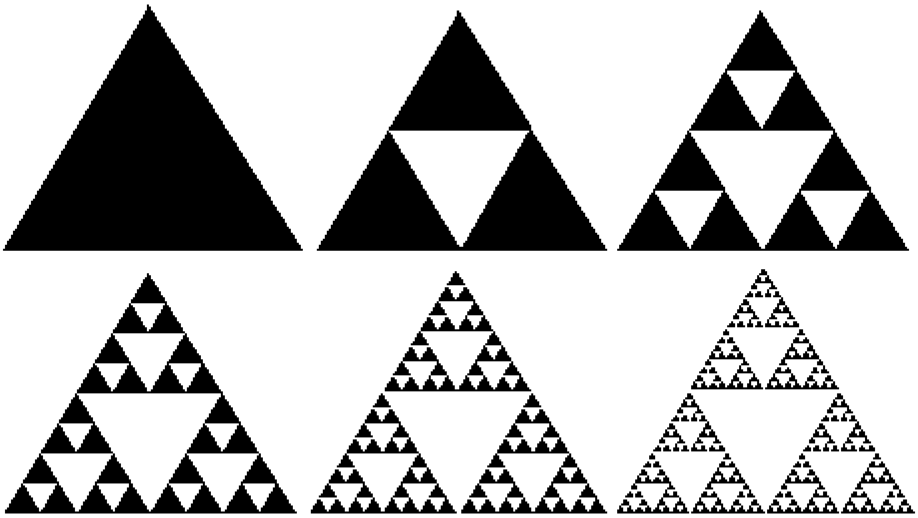


Ковер Серпинского.

Пример простого самоподобного фрактала – ковер Серпинского , придуманный польским математиком Вацлавом Серпинским в 1915 году.



Пусть начальное множество S_0 – равносторонний треугольник вместе с областью, которую он замыкает. Разобьем S_0 на четыре меньшие треугольные области, соединив отрезками середины сторон исходного треугольника. Удалим внутренность маленькой центральной треугольной области. Назовем оставшееся множество S_1 . Затем повторим процесс для каждого из трех оставшихся маленьких треугольников и получим следующее приближение S_2. Продолжая таким образом, получим последовательность вложенных множеств S_n, чье пересечение образует ковер S.



Примеры фракталов в природе.

Квазифракталы.

От ветки, как и от ствола дерева, отходят отростки поменьше, от них – еще меньшие, и т. д., то есть ветка подобна всему дереву. Похожим образом устроена и кровеносная система: от артерий отходят артериолы, а от них – мельчайшие капилляры, по которым кислород поступает в органы и ткани.

Природные объекты называют квазифракталами, так как они отличаются от идеальных абстрактных фракталов неполнотой и неточностью повторений структуры. Большинство встречающихся в природе фракталоподобных структур являются квазифракталами, поскольку на некотором малом масштабе фрактальная структура исчезает. Природные структуры не могут быть идеальными фракталами из-за ограничений, накладываемых размерами живой клетки и, в конечном итоге, размерами молекул.

Примерами фракталов в природе являются:

1. Кораллы
2. Морские звезды и ежи
3. Морские раковины
4. Пчелиные соты
5. Цветы и растения
6. Кроны деревьев и листья растений
7. Система кровообращения и бронхи людей и животных
8. Горные хребты
9. Снежинки
10. Облака
11. Молнии
12. Кристаллы
13. Сталактиты, сталагмиты, геликтиты
14. Морозные узоры

и тд.

Парадокс береговой линии.

Противоречивое наблюдение в географических науках, связанное с невозможностью точно определить длину линии побережья из-за ее фракталоподобных свойств.

Впервые этот вопрос поднял Льюис Ричардсон, а позже Бенуа Мандельброт дополнил его в статье “Какова длина побережья Великобритании? Статистическое самоподобие и фрактальная размерность”, опубликованной в 1967 году.

В 1951 году Льюис Фрай Ричардсон обнаружил: Португалия заявила, что ее сухопутная граница с Испанией равна 987 км, а Испания определила ее равной 1214 км. Такая значительная ошибка показалась ему подозрительной. Это стало отправной точкой в исследовании проблемы береговой линии.

Парадокс заключается в том, что длина береговой линии зависит от способа ее измерения. Если оценка длины производится путем наложения равных отрезков длиной N на карту, то чем меньше длина отрезка измерений, тем больше становится конечная измеряемая длина. И наоборот, чем больше отрезок, тем меньше длина. При этом если длина отрезка стремиться к нулю, то длина береговой линии стремиться к бесконечности.

Применение фракталов.

Информатика.

Фракталы используются для генерирования природных объектов в компьютерной графике, таких как горные ландшафты, деревья, поверхность океана и другие. Впервые фрактальная графика в кино была применена для генерирования лавы в третьем эпизоде фильма “Звездные Воины”.

Также существуют алгоритмы сжатия изображения с помощью фракталов. Они основаны на идее о том, что вместо самого изображения можно хранить сжимающее отображение, для которого это изображение (или некоторое близкое к нему) является неподвижной точкой.

Радиотехника.

Радиоастроном Натан Коэн использовал математику фракталов чтобы совершить прорыв в области электронных средств связи. Он соорудил антенну в форме кривой Коха, что позволило значительно уменьшить размер антенны. Более того фрактальная форма антенн позволяет не только сделать их меньше, но и расширяет доступный диапазон радиоволн. Сегодня повсюду в мире фрактальные антенны используются в телефонах и других беспроводных устройствах связи.

Естественные науки.

В физике фракталы естественным образом возникают при моделировании нелинейных процессов, таких как турбулентное течение жидкости, сложные процессы диффузии-адсорбции, пламя, облака и тому подобное. Фракталы используются при моделировании пористых материалов, например, в нефтехимии. В биологии они применяются для моделирования популяций и для описания систем внутренних органов (система кровеносных сосудов).

ПРАКТИЧЕСКАЯ ЧАСТЬ

Постановка задачи:

Необходимо спроектировать и разработать программный продукт, при помощи которого возможно наглядно посмотреть изображения фрактальной графики.

Решение:

Решением данной задачи является программный продукт, с помощью которого можно увидеть несколько частных примеров фрактальной графики: Ковер Серпинского, Шкура Двойного Дракона (разновидность Кривой Коха) и Множество Мандельброта. Приложение реализовано на языке программирования Python. Программный продукт представляет из себя многооконное приложение, состоящее из меню и окон, в которых демонстрируется построение фракталов.

Меню позволяет выбрать один из трех фракталов для построения или покинуть приложение. В результате нажатия на кнопки «Dragon Koch», «Sierpinski triangle», «Mandelbrot set», открываются окна с изображением соответствующего фрактала.

Разработка графического интерфейса и алгоритма:

Графический интерфейс пользователя и построение фракталов реализованы с помощью библиотеки Tkinter. Полный код программы см. в Приложении 1.

Окно Menu:

Создание окна, задание его размера и названия

```
root = Tk()
root.geometry('400x250')
root.title("Menu")
```

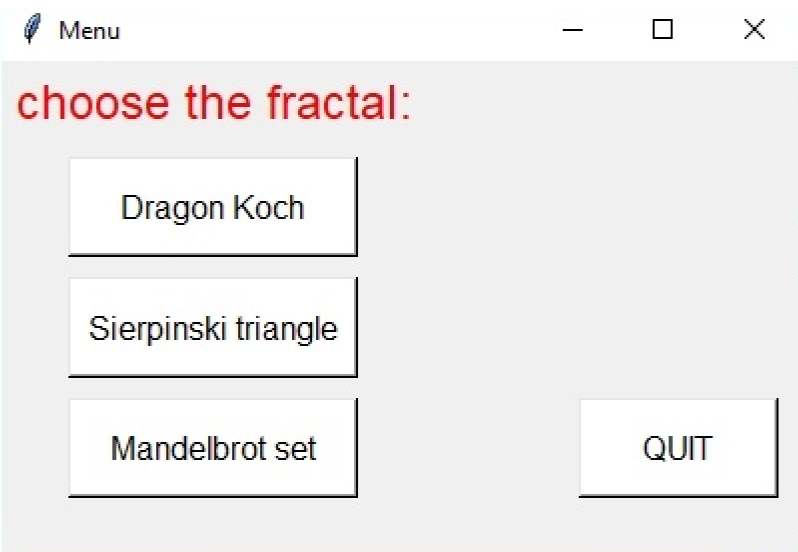
Создание виджетов: текст (Label) и кнопки (Button). У каждого задаются свойства: текст (text), размер (width, height), цвет фона и текста(bg, fg), шрифт(font). Каждая кнопка привязываться к функции (command).

```
label = Label(root, text="choose the fractal:", fg='red', font='robotomono
18')
button1 = Button(root,
                text="Dragon Koch",
                width=15, height=2,
                bg='white', fg='black',
```

```
                font='robotomono 12',
                command=open_window)
button2 = Button(root,
                text="Sierpinski triangle",
                width=15, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=open_window2)
button3 = Button(root,
                text="Mandelbrot set",
                width=15, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=open_window3)
button4 = Button(root,
                text="QUIT",
                width=10, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=root.destroy)
```

Определение положения виджета в окне. Упаковщик grid().

```
label.grid(row=0, column=0, padx=5, pady=5)
button1.grid(row=1, column=0, padx=5, pady=5)
button2.grid(row=2, column=0, padx=5, pady=5)
button3.grid(row=3, column=0, padx=5, pady=5)
button4.grid(row=3, column=1, padx=75, pady=5)
root.mainloop()
```



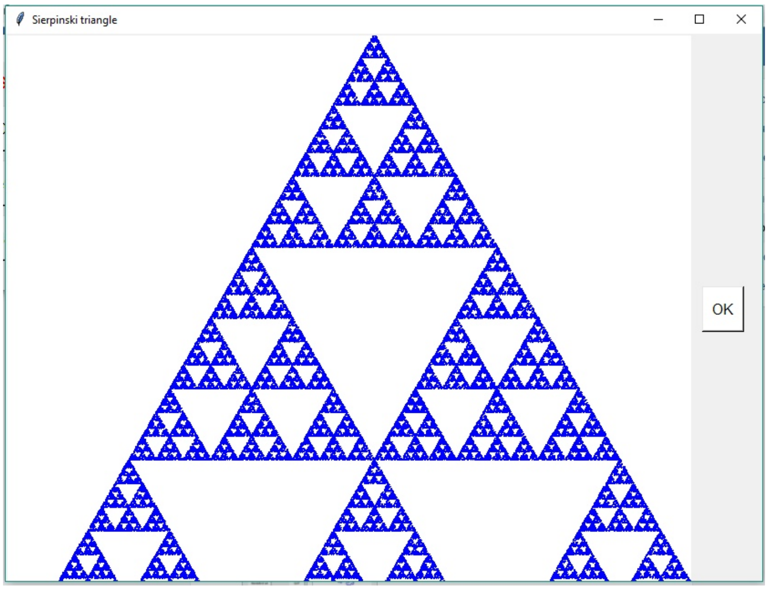
Окна с изображениями (на примере функции для Треугольника Серпинского):

Функция, в которой создается дополнительное окно с помощью виджета Toplevel. Внутри создается область для рисования (Canvas) и кнопка для закрытия окна. В данной функции вызывается другая функция с алгоритмом построения фрактала.

```
def open_window():
   top = Toplevel()
   top.title("Sierpinski triangle")
   top.geometry("830x600")
   canv = Canvas(top, width=750, height=600, bg='white')
   serpinski(canv)
   canv.grid(row=0, column=0)
   button = Button(top,
                   text="OK",
                   width=4, height=2,
                   bg='white', fg='black',
                   font='robotomono 12',
                   command=top.destroy)
   button.grid(row=0, column=1, padx=10, pady=10)
```

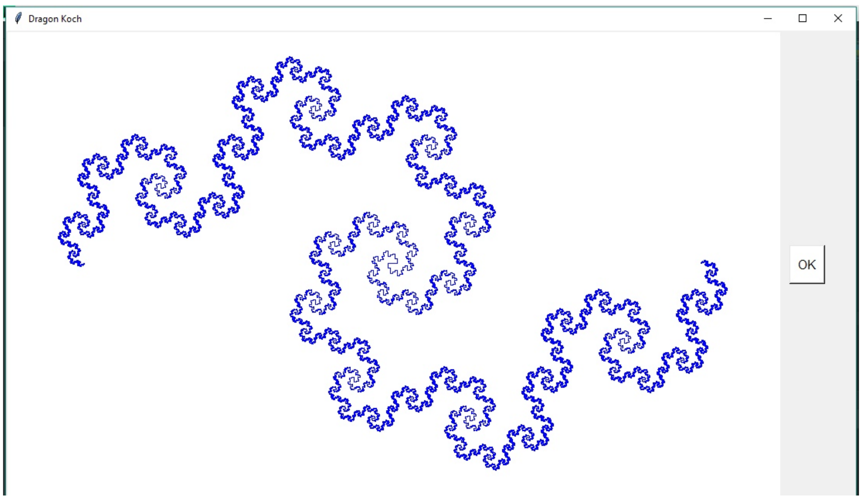
Построение фрактала Треугольник Серпинского:

```
def serpinski(canv):
   x = 0
   y = 0
   mod = 3
   points = 100000
   for t in range(points):
       while True:
           a = random.randint(0, mod - 1)
           b = random.randint(0, mod - 1)
           if a + b < mod:
               break
       x = x / mod + a / mod + b / 2 / mod
       y = y / mod + b / mod

       X = math.floor(x * 808)
       Y = math.floor((1 - y) * 700)

       canv.create_oval(X, Y, X, Y, fill="blue", outline="blue")
```



Построение фрактала Шкура Двойного Дракона:

```
def dragon(canv, x1, y1, x2, y2, n):
   if n:
       dx = (x2 - x1) / 4
       dy = (y2 - y1) / 4
       x3 = x1 + dx + dy
       y3 = y1 + dy - dx
       x4 = x2 - dx - dy
       y4 = y2 - dy + dx
       dragon(canv, x1, y1, x3, y3, n - 1)
       dragon(canv, x3, y3, x4, y4, n - 1)
       dragon(canv, x4, y4, x2, y2, n - 1)
   else:
       canv.create_line(x1, y1, x2, y2, width=0.3, fill='blue')
```



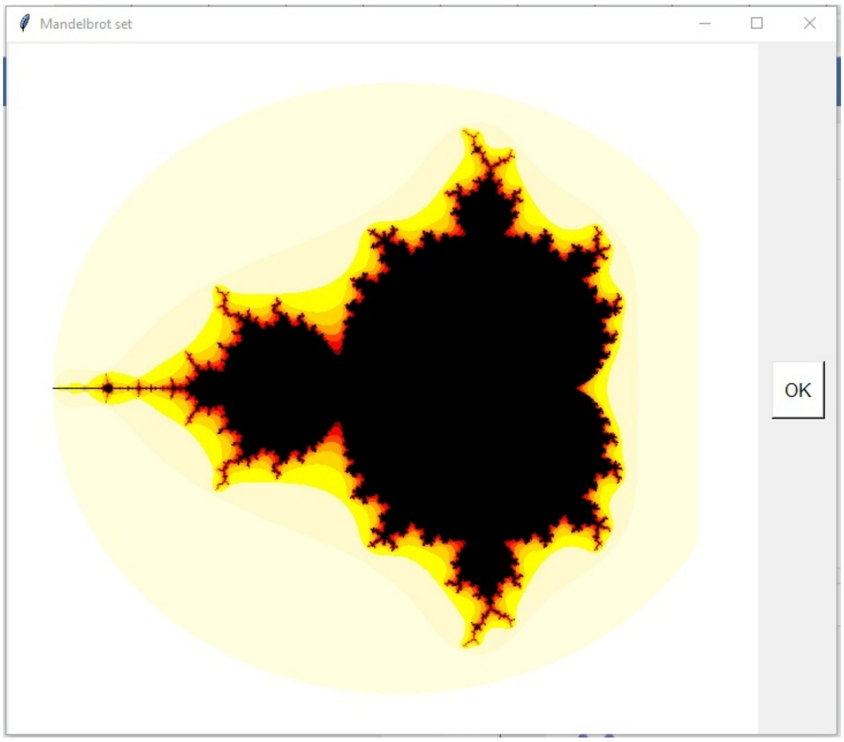
Построение Множества Мандельброта:

Подбор цвета:

```
colors = {
   0: 'white',
   1: 'white',
   2: 'light yellow',
   3: 'light yellow',
   4: 'lemon chiffon',
   5: 'lemon chiffon',
   6: 'yellow',
   7: 'yellow',
   8: 'gold',
   9: 'gold',
   10: 'orange',
   11: 'orange',
   12: 'orange red',
   13: 'orange red',
   14: 'red',
   15: 'red',
   16: 'red',
   17: 'dark red',
   18: 'dark red',
   19: 'dark red',
   20: 'dark red',
   99: 'black'
}

def mandelbrot(c):
   z = 0
   i = 0
   for h in range(0, 20):
       z = z * z + c
       if abs(z) > 2:
           break
       else:
           i += 1
   if abs(z) >= 2:
       return i
   else:
       return 99
```

Отрисовка:

```
for x in range(0, 600):
        real = x / 200.0 - 2.2
        for y in range(0, 600):
            imag = y / 200.0 - 1.5
            c = complex(real, imag)
            p = mandelbrot(c)
            canv.create_line(x, 600 - y, x + 1, 601 - y, fill=colors[p])
```



ЗАКЛЮЧЕНИЕ

Таким образом, выполнение поставленных исследовательских задач позволило получить следующие результаты:

1. Изучено понятие «фрактал» и «самоподобие»;
2. Выявлено, что большинство природных объектов имеют фракталоподобную структуру;
3. Выявлено, что фракталы применяются человеком во многих отраслях науки и искусства;
4. Рассмотрено несколько примеров фракталов с необычными свойствами;
5. Разработан программный продукт, с помощью которого можно увидеть несколько примеров фракталов.

В данной курсовой работе рассмотрена малая часть того, какие бывают фракталы, их особенностей и принципов построения. Мир фракталов обширен и необъятен, а их изучение – это одна из немаловажных задач человека на сегодняшний день.

СПИСОК ЛИТЕРАТУРЫ И ИСТОЧНИКИ

1. Мандельброт Б. Фрактальная геометрия природы. — М.: «Институт компьютерных исследований», 2002.
2. Benoît Mandelbrot (1967). «How Long Is the Coast of Britain? Statistical Self-Similarity and Fractional Dimension», Science, New Series, Vol. 156, No. 3775. (May 5, 1967), pp. 636—638.
3. Маврикиди Ф. И. Фрактальная математика и природа перемен, «Дельфис» — № 54(2) — 2008.
4. Фракталы и хаос в динамических системах. Основы теории. Москва: Постмаркет, 2000. – 352 с.
5. Федер Е. Фракталы. — М: «Мир», 1991.
6. Фракталы. Чудеса природы. Поиски новых размерностей. Документальный фильм, WGBH Educational Foundation and the Catticus Corporation — 2008.

ПРИЛОЖЕНИЕ 1.

```
from tkinter import *
import math
import random


colors = {
   0: 'white',
   1: 'white',
   2: 'light yellow',
   3: 'light yellow',
   4: 'lemon chiffon',
   5: 'lemon chiffon',
   6: 'yellow',
   7: 'yellow',
   8: 'gold',
   9: 'gold',
   10: 'orange',
   11: 'orange',
   12: 'orange red',
   13: 'orange red',
   14: 'red',
   15: 'red',
   16: 'red',
   17: 'dark red',
   18: 'dark red',
   19: 'dark red',
   20: 'dark red',
   99: 'black'
}


def serpinski(canv):
   x = 0
   y = 0
   mod = 3
   points = 100000
   for t in range(points):
       while True:
           a = random.randint(0, mod - 1)
           b = random.randint(0, mod - 1)
           if a + b < mod:
               break
       x = x / mod + a / mod + b / 2 / mod
       y = y / mod + b / mod
```

Продолжение приложения 1.

```
      X = math.floor(x * 808)
        Y = math.floor((1 - y) * 700)

        canv.create_oval(X, Y, X, Y, fill="blue", outline="blue")


def dragon(canv, x1, y1, x2, y2, n):
   if n:
       dx = (x2 - x1) / 4
       dy = (y2 - y1) / 4
       x3 = x1 + dx + dy
       y3 = y1 + dy - dx
       x4 = x2 - dx - dy
       y4 = y2 - dy + dx
       dragon(canv, x1, y1, x3, y3, n - 1)
       dragon(canv, x3, y3, x4, y4, n - 1)
       dragon(canv, x4, y4, x2, y2, n - 1)
   else:
       canv.create_line(x1, y1, x2, y2, width=0.3, fill='blue')


def mandelbrot(c):
   z = 0
   i = 0
   for h in range(0, 20):
       z = z * z + c
       if abs(z) > 2:
           break
       else:
           i += 1
   if abs(z) >= 2:
       return i
   else:
       return 99


def open_window():
   top = Toplevel()
   top.title("Dragon Koch")
   top.geometry("1100x600")
   canv = Canvas(top, width=1000, height=600, bg='white')
   N = 12
   dragon(canv, 100, 300, 900, 300, N)
   canv.grid(row=0, column=0)
```

Продолжение приложения 1.

```
    button = Button(top,
                    text="OK",
                    width=4, height=2,
                    bg='white', fg='black',
                    font='robotomono 12',
                    command=top.destroy)
    button.grid(row=0, column=1, padx=10, pady=10)


def open_window2():
    top = Toplevel()
    top.title("Sierpinski triangle")
    top.geometry("830x600")
    canv = Canvas(top, width=750, height=600, bg='white')
    serpinski(canv)
    canv.grid(row=0, column=0)
    button = Button(top,
                    text="OK",
                    width=4, height=2,
                    bg='white', fg='black',
                    font='robotomono 12',
                    command=top.destroy)
    button.grid(row=0, column=1, padx=10, pady=10)


def open_window3():
    top = Toplevel()
    top.title("Mandelbrot set")
    top.geometry("720x600")
    canv = Canvas(top, width=650, height=600, bg='white')
    canv.grid(row=0, column=0)
    for x in range(0, 600):
        real = x / 200.0 - 2.2
        for y in range(0, 600):
            imag = y / 200.0 - 1.5
            c = complex(real, imag)
            p = mandelbrot(c)
            canv.create_line(x, 600 - y, x + 1, 601 - y, fill=colors[p])
            canv.grid(row=0, column=0)

    button = Button(top,
                    text="OK",
                    width=4, height=2,
                    bg='white', fg='black',
```

Продолжение приложения 1.

```
                  font='robotomono 12',
                  command=top.destroy)
   button.grid(row=0, column=1, padx=10, pady=10)


root = Tk()
root.geometry('400x250')
root.title("Menu")

label = Label(root, text="choose the fractal:", fg='red', font='robotomono 
18')
button1 = Button(root,
                text="Dragon Koch",
                width=15, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=open_window)
button2 = Button(root,
                text="Sierpinski triangle",
                width=15, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=open_window2)
button3 = Button(root,
                text="Mandelbrot set",
                width=15, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=open_window3)
button4 = Button(root,
                text="QUIT",
                width=10, height=2,
                bg='white', fg='black',
                font='robotomono 12',
                command=root.destroy)
label.grid(row=0, column=0, padx=5, pady=5)
button1.grid(row=1, column=0, padx=5, pady=5)
button2.grid(row=2, column=0, padx=5, pady=5)
button3.grid(row=3, column=0, padx=5, pady=5)
button4.grid(row=3, column=1, padx=75, pady=5)
root.mainloop()
```