Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Уфимский государственный авиационный технический университет»

Факультет ИРТ

Кафедра ВВТиС

**ОТЧЕТ**

**по учебной практике**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Группа МКН-315 | Фамилия И.О. | Подпись | Дата | Оценка |
| Студент | Сиротин А.Е. |  |  |  |
| Руководитель  от базы практики | Ямилева А.М. |  |  |  |
| Руководитель от университета | Ямилева А.М. |  |  |  |

Уфа 2022

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Уфимский государственный авиационный технический университет»

Кафедра высокопроизводительных вычислительных технологий и систем

**ЗАДАНИЕ**

на учебную практику

Студент: Сиротин Александр Евгеньевич Группа: МКН-315

Период практики: 06.06.2022 – 03.07.2022

Вид (тип) практики: учебная практика (научно-исследовательская работа (получение первичных навыков научно-исследовательской работы))

База практической подготовки: каф. ВВТиС, ФГБОУ ВО УГАТУ

Руководитель от базы практической подготовки: Ямилева А.М.

Руководитель от университета: Ямилева А.М.

Тема практики

Применение Python для решения математических задач.

Задачи исследования

1. Изучить язык программирования python, библиотеку numpy на примере онлайн-курсов [1, 2, 3]. Выполнить задания по python согласно варианту.

2. Изучить функции библиотек numpy и scipy для решения математических задач.

3. Изучить функции и объекты библиотеки pyqt. Написать оконное приложение с использованием pyqt, реализующее следующий функционал: визуализация фракталов Жюлиа, в зависимости от входных параметров. Использование файловой системы для сохранения сгенерированного изображения.

Список рекомендуемой литературы

1. Балакирев, С. Добрый, добрый Python - обучающий курс от Сергея Балакирева // Stepik : Образовательная платформа. — Режим доступа: <https://stepik.org/course/100707/syllabus>, для авториз. пользователей. — Загл. с экрана.

2. Балакирев, С. Добрый, добрый Python ООП - обучающий курс от Сергея Балакирева // Stepik : Образовательная платформа. — Режим доступа: <https://stepik.org/course/116336/syllabus>, для авториз. пользователей. — Загл. с экрана.

3. Задойный, А. Практикум по математике и Python : образовательный курс // Stepik : Образовательная платформа. — Режим доступа:

<https://stepik.org/course/3356/syllabus>, для авториз. пользователей. — Загл. с экрана.

4. Лучано, Р. Python. К вершинам мастерства / Р. Лучано ; перевод с английского А. А. Слинкин. — Москва : ДМК Пресс, 2016. — 768 с. — Режим доступа: <https://e.lanbook.com/book/93273>, для авториз. пользователей.

|  |  |
| --- | --- |
| Дата выдачи задания  06 июня 2022 г. | Дата окончания работы  02 июля 2022 г. |

Руководитель от университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ямилева А.М.

Принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сиротин А.Е.

**СОДЕРЖАНИЕ**

[Введение 5](#_Toc107992763)

[1 Теоретическая часть 6](#_Toc107992764)

[1.1 Язык программирования Python 6](#_Toc107992765)

[1.2 Научные библиотеки python 6](#_Toc107992766)

[1.3 Библиотека PyQt 7](#_Toc107992767)

[1.4 Множества Жюлиа 8](#_Toc107992768)

[2 Практическая часть 9](#_Toc107992769)

[2.1 Изучение Python 9](#_Toc107992770)

[2.2 Библиотеки python 9](#_Toc107992771)

[2.2.1 Matplotlib 9](#_Toc107992772)

[2.2.2 numpy 10](#_Toc107992773)

[2.2.3 scipy 11](#_Toc107992774)

[2.3 модельные задачи 11](#_Toc107992775)

[2.3.1 Краевая задача для оду 2-го порядка 11](#_Toc107992776)

[2.3.2 Краевая задача для оду 4-го порядка 14](#_Toc107992777)

[2.3.3 Решение нелинейного уравнения и систем нелинейных уравнений 14](#_Toc107992778)

[2.3.4 Метод наименьших квадратов 15](#_Toc107992779)

[2.3.5 Модель эпидемии SIR 17](#_Toc107992780)

[2.3.6 Модель Лотки-Вольтерра 18](#_Toc107992781)

[2.4 Приложение на PyQt5 19](#_Toc107992782)

[Заключение 22](#_Toc107992783)

[Список литературы 23](#_Toc107992784)

[Приложение А листинг программы 24](#_Toc107992785)

# Введение

Python представляет популярный высокоуровневый язык программирования, который предназначен для создания приложений различных типов приложений. Это и веб-приложения, и игры, и настольные программы, и работа с базами данных. Довольно большое распространение питон получил в области машинного обучения и исследований искусственного интеллекта.

Также с его помощью, а именно с помощью математических библиотек, можно решать прикладные задачи по математике: построение графиков, проведение математических расчетов, решение ОДУ и СЛАУ, построение математических моделей.

# 1 Теоретическая часть

## 1.1 Язык программирования Python

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++.

## 1.2 Научные библиотеки python

NumPy (сокращенно от Numerical Python)— библиотека с открытым исходным кодом для языка программирования Python.

Математические алгоритмы, реализованные на интерпретируемых языках (например, Python), часто работают гораздо медленнее тех же алгоритмов, реализованных на компилируемых языках (например, Фортран, Си, Java). Библиотека NumPy предоставляет реализации вычислительных алгоритмов (в виде функций и операторов), оптимизированные для работы с многомерными массивами. В результате любой алгоритм, который может быть выражен в виде последовательности операций над массивами (матрицами) и реализованный с использованием NumPy, работает так же быстро, как эквивалентный код, выполняемый в MATLAB.

SciPy — это библиотека Python с открытым исходным кодом, предназначенная для решения научных и математических проблем. Она построена на базе NumPy и позволяет управлять данными, а также визуализировать их с помощью разных высокоуровневых команд.

И NumPy, и SciPy являются библиотеками Python, которые используются для математического и числового анализов. NumPy содержит данные массивов и операции, такие как сортировка, индексация, а SciPy состоит из числового кода. И хотя в NumPy есть функции для работы с линейной алгеброй, преобразованиями Фурье и т. д., в SciPy они представлены в полном виде вместе с массой других. А для полноценного научного анализа в Python нужно устанавливать и NumPy, и SciPy, поскольку последняя построена на базе NumPy.

## 1.3 Библиотека PyQt

PyQt — набор расширений (биндингов) графического фреймворка Qt для языка программирования Python, выполненный в виде расширения Python.

PyQt разработан британской компанией Riverbank Computing. PyQt работает на всех платформах, поддерживаемых Qt: Linux и другие UNIX-подобные ОС, macOS и Windows. Существует 3 версии: PyQt6, PyQt5 и PyQt4, поддерживающие соответствующие версии Qt. PyQt распространяется под лицензиями GPL (2 и 3 версии) и коммерческой.

PyQt практически полностью реализует возможности Qt. Это более 600 классов, более 6000 функций и методов.

PyQt также включает в себя Qt Designer (Qt Creator) — дизайнер графического интерфейса пользователя. Программа pyuic генерирует Python код из файлов, созданных в Qt Designer. Это делает PyQt очень полезным инструментом для быстрого прототипирования. Кроме того, можно добавлять новые графические элементы управления, написанные на Python, в Qt Designer.

Раньше PyQt поставлялся вместе со средой разработки Eric, написанной на PyQt. Eric имеет встроенный отладчик и может быть использована для создания консольных программ. Теперь она доступна в качестве отдельного проекта.

## 1.4 Множества Жюлиа

В голоморфной динамике мно́жество Жюлиа́ — множество точек, динамика в окрестности которых в определённом смысле неустойчива по отношению к малым возмущениям начального положения. В случае, если f — полином, рассматривают также заполненное множество Жюлиа — множество точек, не стремящихся к бесконечности. Обычное множество Жюлиа при этом является его границей.

Множество Фату — дополнение к множеству Жюлиа. Иными словами, динамика итерирования f на регулярна, а на хаотична.

Дополняет большую теорему Пикара о «поведении аналитической функции в окрестности существенно особой точки».

Эти множества названы по именам французских математиков Гастона Жюлиа и Пьера Фату, положивших начало исследованию голоморфной динамики в начале XX века.

# 2 Практическая часть

<Сделать подразделы по каждому заданию, всего 3 подраздела. Выполнение задач с курсов описывается кратко, одним-двумя абзацами. Инд. задание по numpy (клеточные автоматы) и scipy расписывается более подробно (постановка/описание задачи и результат). Инд. задание для pyqt описывается подробно с полученными примерами и анализом результатов. Пункты (заголовки 3 уровня) вводятся при необходимости.>

## 2.1 Изучение Python

Изучение основ языка программирования Python на примере онлайн курсов включило в себя изучение следующих разделов:

* управляющие конструкции в Python (print, input, арифметические операции, условные конструкции (if, elif, else) и цикл while)
* cтроки и списки
* функции
* коллекции – генераторы с itertools
* ООП (Объектно*-*ОриентированноеПрограммирование)
* работа с файлами
* встроенные библиотеки

Данные разделы были изучены на платформе Stepik [1,2].

## 2.2 Библиотеки python

### 2.2.1 Matplotlib

Matplotlib является гибким, легко конфигурируемым пакетом, который вместе с NumPy, SciPy и IPython предоставляет возможности, подобные MATLAB.

В процессе изучения были использованы различные возможности библиотеки Matplotlib. Например, была построена параметризованная кривая , может быть построена непосредственно с помощью [plot](https://matplotlib.org/stable/api/_as_gen/matplotlib.axes.Axes.plot.html" \l "matplotlib.axes.Axes.plot" \o "matplotlib.axes.Axes.plot). (Рисунок 1).

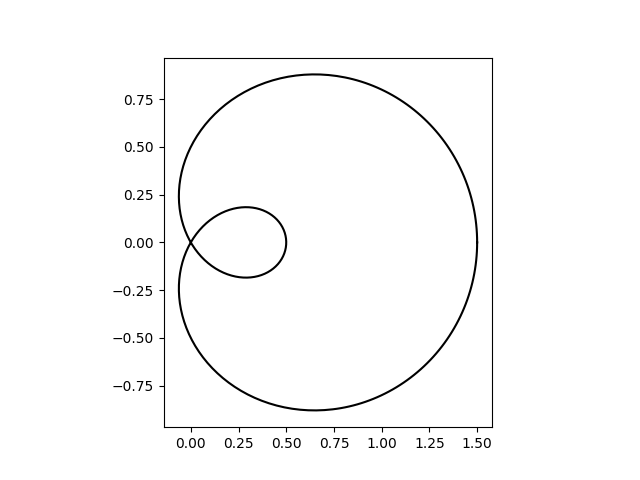


Рисунок 1 ­– График параметрической кривой

### 2.2.2 numpy

На платформе Stepik (Практикум по математике и Python) [3] были выполнены задания по следующим разделам:

* Основные операции создания векторов и матриц
* Манипуляции индексов этих векторов и матриц
* Основные математические операции как скалярное произведение векторов, умножение матриц, генерации случайных чисел, и т. д.

Также была реализована игра «Жизнь», которая является одной из разновидностей клеточных автоматов. Пример работы программы изображен на рисунке 2.

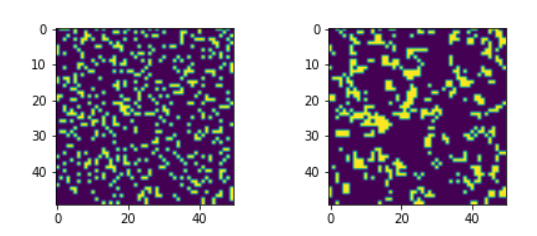


Рисунок 2 – пример работы игры «Жизнь»

### 2.2.3 scipy

SciPy – библиотека для языка программирования Python с открытым исходным кодом, предназначенная для выполнения научных и инженерных расчётов.

Данная библиотека использовалась для решения модельных задач, а именно: построение решения краевой задачи для ОДУ 2-го и 4-го порядков, решение нелинейных уравнений и минимизация функций с помощью модуля scipy.optimize, интегрирование ОДУ с помощью модуля scipy.integrate.

## 2.3 модельные задачи

### 2.3.1 Краевая задача для оду 2-го порядка

Необходимо решить краевую задачу для ОДУ 2-го порядка методом конечных разностей:

(1)

Конечно–разностная схема имеет вид:

Данная схема имеет второй порядок аппроксимации по h.

Аппроксимация краевых условий второго порядка точности по h имеет вид:

Таким образом, мы получили 3-х диагональную матрицу системы размера (N+1) x (N+1).

Используя полноразмерный вариант матрицы, была написана программа для решения полученной системы линейных уравнений с помощью решателя linalg.solve. Также матрица была преобразована в ленточный вид для использования решателя linalg.solve\_banded.

В качестве примера возьмем:

(2)

График точного решения этой краевой задачи можно увидеть на рисунке 3.

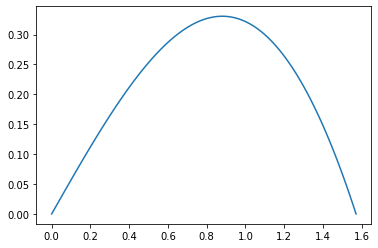


Рисунок 3 – график точного решения краевой задачи (2)

График численного решения задачи можно увидеть на рисунке 4.

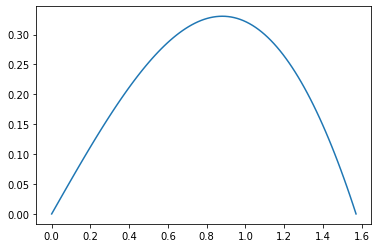


Рисунок 4 – график численного решения краевой задачи (2)

График погрешности краевой задачи можно увидеть на рисунке 5.

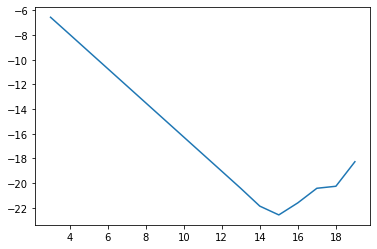


Рисунок 5 – погрешности краевой задачи (2) в логарифмической норме

По таблице 1 видно, что решатель linalg.solve\_banded требует значительно меньше времени.

Таблица 1 – время затраченное на решение СЛАУ разными решателями

|  |  |  |  |
| --- | --- | --- | --- |
| N | 100 | 1000 | 10000 |
| t, c – linalg.solve | 0.171875 | 0.15625 | 35.53125 |
| t,c – linalg.solve\_banded | 0.09375 | 0.109375 | 1.90625 |

### 2.3.2 Краевая задача для оду 4-го порядка

Необходимо решить краевую задачу для ОДУ 4-го порядка методом конечных разностей:

(3)

Конечно–разностная схема имеет вид:

Аппроксимация краевых условий первого порядка точности по h имеет вид:

В результате мы имеем 5-ти диагональную матрицу.

Аналогично пункту 2.3.1, написана программа, строящая решение с использованием linalg.solve\_banded и linalg.solve.

### 2.3.3 Решение нелинейного уравнения и систем нелинейных уравнений

Необходимо решить нелинейное уравнение и систему нелинейных уравнений, предварительно отделив корни аналитически или графически.

Нелинейное уравнение:

(4)

Для решения уравнения необходимо использовать fsolve из модуля scipy.optimize. Входными параметрами является функция вида , и некоторая точка, в окрестности которой будет искаться корень.

Результат работы программы изображен на рисунке 6.



Рисунок 6 – корни уравнения (4)

Аналогичным образом решается система уравнений (5):

(5)

Результат работы программы изображен на рисунке 7.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 – решение системы уравнений (5)

### 2.3.4 Метод наименьших квадратов

#### 2.3.4.1 Задача 1

Требуется восстановить линейную функциональную зависимость методом наименьших квадратов, вычислить полную квадратичную невязку полученной функции и вычислить значение функции при температуре 21°С. Привести график полученной функциональной зависимости вместе с экспериментальными точками.

Таблица 2 – экспериментально измеренные значения сопротивления при трех различных значениях температуры

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | T1 | T2 | T3 | T4 | T5 | R1 | R2 | R3 | R4 | R5 |
| 3 | 20 | 22 | 24 | 26 | 28 | 40 | 29.0 | 21.1 | 15.3 | 11.1 |

Результат работы программы изображен на рисунке 8.

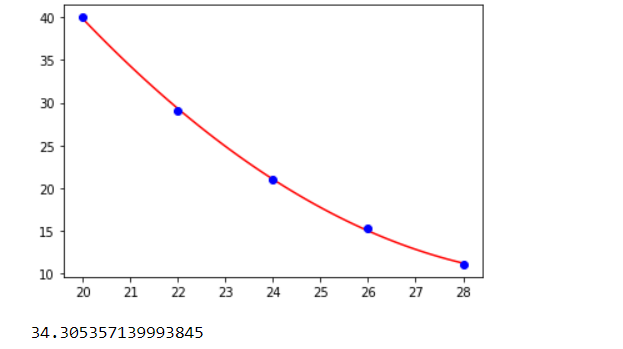


Рисунок 8 – график полученной функциональной зависимости вместе с экспериментальными точками, и значение функции при температуре 21°С

#### 2.3.4.2 Задача 2

Используя метод наименьших квадратов, необходимо определить константу распада, период полураспада и значение суммы квадратов невязок. Привести график полученной функциональной зависимости вместе с экспериментальными точками. Закон радиоактивного распада: . Из определения периода полураспада следует его связь с постоянной распада:

Таблица 3 – экспериментально измеренные значения

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№** | **T1** | **T2** | **T3** | **T4** | **T5** | **T6** | **N1** | **N2** | **N3** | **N4** | **N5** | **N6** |
| 3 | 0 | 0.249024 | 0.498048 | 0.747072 | 0.996096 | 1.24512 | 1000872 | 942719 | 891818 | 840464 | 795312 | 748568 |

Результат работы программы изображен на рисунке 9.

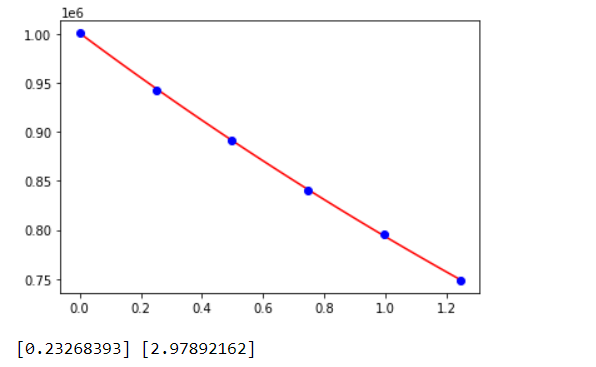


Рисунок 9 – график полученной функциональной зависимости вместе с экспериментальными точками, и значение константы распада с периодом полураспада соответственно.

### 2.3.5 Модель эпидемии SIR

При изучении развития эпидемии некоторого заболевания обычно выделяют 3 группы людей:

S – группа людей, восприимчивых к данному заболеванию, но еще не заразившихся (subjected);

I – группа уже инфицированных людей, которые являются разносчиками данного заболевания (infected);

R – группа людей, невосприимчивых к данному заболеванию, получивших иммунитет или изолированных (removed).

Один из вариантов математической модели может быть записан в виде:

(6)

где 𝛽 - скорость распространения инфекции, 𝛾 - скорость выздоровления.

В силу линейной зависимости системы (сумма всех людей постоянна) достаточно рассматривать только два уравнения из трех, а третье восстановить алгебраически.

Необходимо рассмотреть динамику системы (т.е. решить систему уравнений и получить траектории) при некоторых произвольных начальных данных и значениях параметров. Привести графики зависимостей.

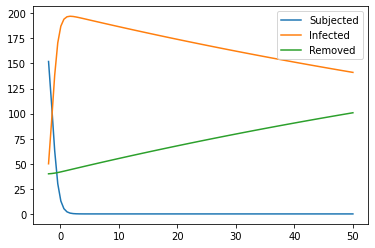


Рисунок 10 – Модель эпидемии SIR при , , , и

### 2.3.6 Модель Лотки-Вольтерра

Модель Лотки-Вольтерры, также известную как уравнения хищник-жертва, которая представляет собой пару нелинейных дифференциальных уравнений первого порядка, часто используемых для описания динамики биологических систем, в которых взаимодействуют два вида, один хищник, а другой его добыча. Модель была предложена независимо Альфредом Лоткой в 1925 году и Вито Вольтеррой в 1926 году и может быть описана как:

(7)

Модуль scipy.integrate предлагает метод под названием odeint, который очень прост в использовании для интегрирования ОДУ. Так, при , получаем следующее решение, изображенное на рисунке 11.

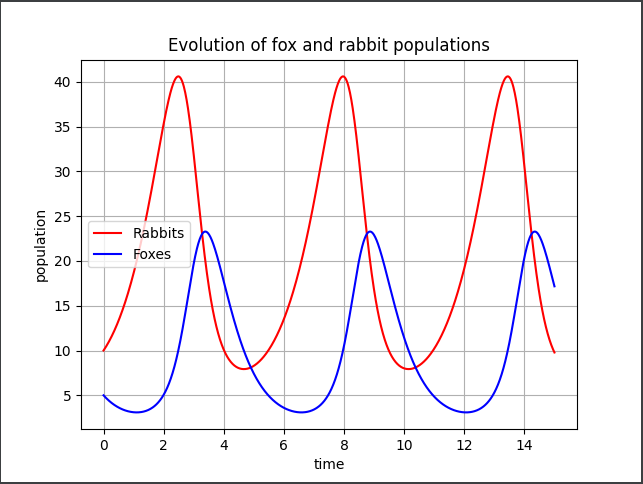


Рисунок 11 – решение системы уравнений (7)

Модифицированная модель Лотки-Вольтерра:

(8)

Получаем следующее решение, изображенное на рисунке 12.

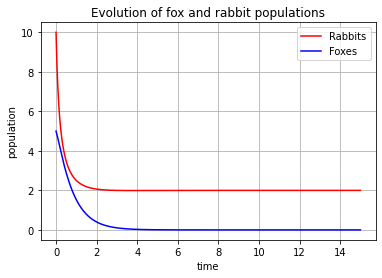


Рисунок 12 – решение системы уравнений (8) при , , , и

## 2.4 Приложение на PyQt5

Приложение считывает входные параметры и на их основе строит визуализация множества Жюлиа. Для создания приложение были использованы библиотеки PyQt5 и NumPy. После визуализации изображение можно сохранить на компьютер. У фракталов можно изменить точность, варьируя кол-во итераций, четкость, изменяя разрешение. Также можно изменить параметр c множества Жюлиа.

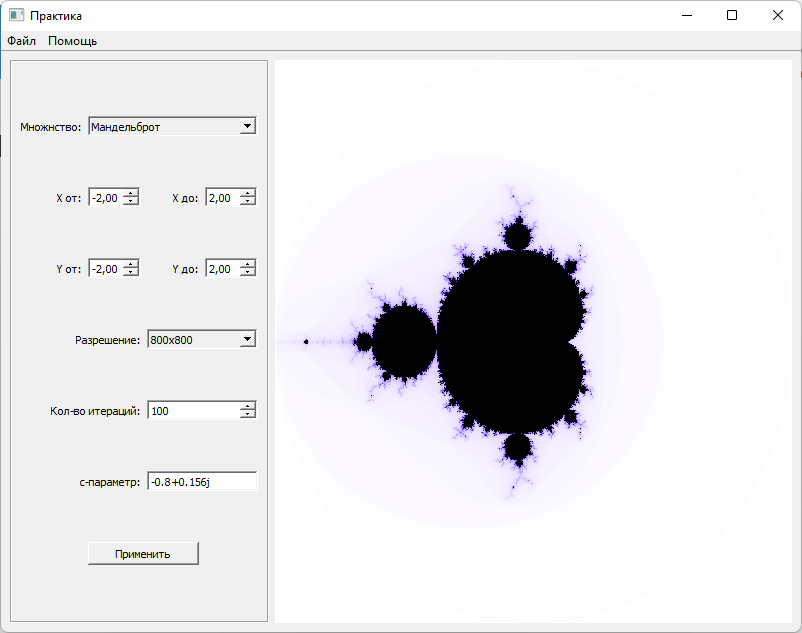


Рисунок 13 – интерфейс программы

Изображение выглядит как керамические изделия, фарфор

Автоматически созданное описание

Рисунок 14 – множество Жюлиа

Изображение выглядит как посуда, фарфор, тарелка, керамические изделия

Автоматически созданное описание

Рисунок 15 – множество Жюлиа

# Заключение

В ходе учебной практики были изучены синтаксис и возможности языка программирования Python, изучены специализированные математические библиотеки такие как Matplotlib, NumPy, SciPy, а также графическая библиотека PyQt. Были выполнены практические задачи с использованием этих инструментов: создание многооконных приложений, построение графиков, проведение математических расчетов, решение СЛАУ и ДУ.

# Список литературы

1. Балакирев, С. Добрый, добрый Python - обучающий курс от Сергея Балакирева // Stepik : Образовательная платформа. — Режим доступа: <https://stepik.org/course/100707/syllabus>, для авториз. пользователей. — Загл. с экрана.
2. Балакирев, С. Добрый, добрый Python ООП - обучающий курс от Сергея Балакирева // Stepik : Образовательная платформа. — Режим доступа: <https://stepik.org/course/116336/syllabus>, для авториз. пользователей. — Загл. с экрана.
3. Задойный, А. Практикум по математике и Python : образовательный курс // Stepik : Образовательная платформа. — Режим доступа: <https://stepik.org/course/3356/syllabus>, для авториз. пользователей. — Загл. с экрана.
4. Numpy v1.21 Manual // NumPy – Режим доступа: <https://numpy.org/doc/stable>
5. Matplotlib : lotka-volterra tutorial // Scipy CookBook – Режим доступа: <https://scipy-cookbook.readthedocs.io/items/LoktaVolterraTutorial.html>
6. Scipy.org // Scipy – Режим доступа: https://www.scipy.org

# Приложение А листинг программы

import numpy as np  
from PyQt5 import QtGui  
from PyQt5.QtCore import Qt  
from PyQt5.QtGui import QPainter, QColor, QPixmap  
from PyQt5.QtWidgets import QMainWindow, QApplication, QWidget, QPushButton, QGridLayout, \  
 QGroupBox, QComboBox, QLabel, QDoubleSpinBox, QSpinBox, QLineEdit, QAction, QFileDialog, \  
 QMessageBox  
  
  
class MainWindow(QMainWindow):  
 size\_index = 2  
  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
  
 self.create\_dictionary()  
 self.create\_actions()  
 self.create\_menu()  
 self.create\_central\_widget()  
  
 self.setCentralWidget(self.central\_widget)  
  
 self.setWindowTitle("Практика")  
 self.setMinimumSize(160, 160)  
 self.resize(800, 600)  
  
 def create\_actions(self):  
 self.save\_as\_act = QAction("Сохранить как", self)  
 self.save\_as\_act.setShortcut("Ctrl+S")  
 self.save\_as\_act.triggered.connect(self.save\_as)  
  
 self.exit\_act = QAction("Выход", self)  
 self.exit\_act.setShortcut("Ctrl+Q")  
 self.exit\_act.triggered.connect(self.close)  
  
 self.about\_act = QAction("О приложении", self)  
 self.about\_act.triggered.connect(self.about)  
  
  
 def create\_dictionary(self):  
 self.about\_text = """  
 В голоморфной динамике мно́жество Жюлиа́ J(f) рационального отображения —   
 множество точек, динамика в окрестности которых в определённом смысле неустойчива по отношению к малым возмущениям   
 начального положения. В случае, если f — полином, рассматривают также заполненное множество Жюлиа — множество точек,   
 не стремящихся к бесконечности. Обычное множество Жюлиа при этом является его границей.  
   
 Чтобы визуализировать множество с заданными параметрами нажмите "применить"  
 Чтобы сохранить изображение используйте Ctrl + S  
 Чтобы выйти Ctrl + Q  
 """  
  
 def save\_as(self):  
 filename, \_ = QFileDialog.getSaveFileName(self)  
 if filename:  
 return self.save\_file(filename)  
  
 def save\_file(self, filename):  
 return self.mj\_sets\_widget.q\_image.save(f"{filename}.png", "PNG", -1)  
  
 def about(self):  
 QMessageBox.about(self, "О приложении", self.about\_text)  
  
 def create\_menu(self):  
 self.file\_menu = self.menuBar().addMenu("Файл")  
 self.file\_menu.addAction(self.save\_as\_act)  
 self.file\_menu.addAction(self.exit\_act)  
  
 self.help\_menu = self.menuBar().addMenu("Помощь")  
 self.help\_menu.addAction(self.about\_act)  
  
 def create\_central\_widget(self):  
 self.central\_widget = QWidget()  
  
 self.create\_mj\_sets\_widget()  
 self.create\_grid\_group\_box()  
  
 main\_layout = QGridLayout()  
  
 main\_layout.setColumnStretch(0, 1)  
 main\_layout.setColumnStretch(1, 2)  
  
 main\_layout.addWidget(self.group\_box, 0, 0)  
 main\_layout.addWidget(self.mj\_sets\_widget, 0, 1)  
  
 self.central\_widget.setLayout(main\_layout)  
  
 def create\_grid\_group\_box(self):  
 self.group\_box = QGroupBox()  
  
 self.set\_combo\_box = QComboBox()  
 self.set\_combo\_box.addItem("Жюлиа", 0)  
 self.set\_combo\_box.addItem("Мандельброт", 1)  
 self.set\_combo\_box.activated.connect(self.changed\_set)  
  
 self.c\_line\_edit = QLineEdit()  
 self.c\_line\_edit.setText(str(self.mj\_sets\_widget.c).strip('()'))  
 self.c\_line\_edit.setReadOnly(False)  
  
 self.image\_size\_combo\_box = QComboBox()  
 self.image\_size\_combo\_box.addItem("800x800", (800, 800))  
 self.image\_size\_combo\_box.addItem("400x400", (400, 400))  
 self.image\_size\_combo\_box.addItem("200x200", (200, 200))  
 self.image\_size\_combo\_box.addItem("100x100", (100, 100))  
 self.image\_size\_combo\_box.setCurrentIndex(3)  
  
 self.diverge\_number\_spin\_box = QSpinBox()  
 self.diverge\_number\_spin\_box.setRange(1, 200)  
 self.diverge\_number\_spin\_box.setValue(self.mj\_sets\_widget.num)  
  
 self.x\_min\_spin\_box = QDoubleSpinBox()  
 self.x\_max\_spin\_box = QDoubleSpinBox()  
 self.y\_min\_spin\_box = QDoubleSpinBox()  
 self.y\_max\_spin\_box = QDoubleSpinBox()  
  
 self.x\_min\_spin\_box.setSingleStep(0.01)  
 self.x\_max\_spin\_box.setSingleStep(0.01)  
 self.y\_min\_spin\_box.setSingleStep(0.01)  
 self.y\_max\_spin\_box.setSingleStep(0.01)  
  
 self.x\_min\_spin\_box.setRange(-5, 5)  
 self.x\_max\_spin\_box.setRange(-5, 5)  
 self.y\_min\_spin\_box.setRange(-5, 5)  
 self.y\_max\_spin\_box.setRange(-5, 5)  
  
 self.x\_min\_spin\_box.setValue(self.mj\_sets\_widget.x\_min)  
 self.x\_max\_spin\_box.setValue(self.mj\_sets\_widget.x\_max)  
 self.y\_min\_spin\_box.setValue(self.mj\_sets\_widget.y\_min)  
 self.y\_max\_spin\_box.setValue(self.mj\_sets\_widget.y\_max)  
  
 self.x\_min\_label = QLabel("X от:")  
 self.x\_max\_label = QLabel("X до:")  
 self.y\_min\_label = QLabel("Y от:")  
 self.y\_max\_label = QLabel("Y до:")  
  
 self.set\_label = QLabel("Множнство:")  
  
 self.c\_line\_edit\_label = QLabel("c-параметр:")  
  
 self.image\_size\_label = QLabel("Разрешение:")  
  
 self.diverge\_number\_label = QLabel("Кол-во итераций:")  
  
 self.apply\_button = QPushButton("Применить")  
 self.apply\_button.clicked.connect(self.apply\_pressed)  
  
 layout = QGridLayout()  
  
 layout.addWidget(self.set\_label, 0, 0, Qt.AlignRight)  
 layout.addWidget(self.set\_combo\_box, 0, 1, 1, 3)  
 layout.addWidget(self.x\_min\_label, 1, 0, Qt.AlignRight)  
 layout.addWidget(self.x\_min\_spin\_box, 1, 1)  
 layout.addWidget(self.x\_max\_label, 1, 2, Qt.AlignRight)  
 layout.addWidget(self.x\_max\_spin\_box, 1, 3)  
 layout.addWidget(self.y\_min\_label, 2, 0, Qt.AlignRight)  
 layout.addWidget(self.y\_min\_spin\_box, 2, 1)  
 layout.addWidget(self.y\_max\_label, 2, 2, Qt.AlignRight)  
 layout.addWidget(self.y\_max\_spin\_box, 2, 3)  
 layout.addWidget(self.image\_size\_label, 3, 0, 1, 2, Qt.AlignRight)  
 layout.addWidget(self.image\_size\_combo\_box, 3, 2, 1, 2)  
 layout.addWidget(self.diverge\_number\_label, 4, 0, 1, 2, Qt.AlignRight)  
 layout.addWidget(self.diverge\_number\_spin\_box, 4, 2, 1, 2)  
 layout.addWidget(self.c\_line\_edit\_label, 5, 0, 1, 2, Qt.AlignRight)  
 layout.addWidget(self.c\_line\_edit, 5, 2, 1, 2)  
 layout.addWidget(self.apply\_button, 6, 1, 1, 2)  
  
 self.group\_box.setLayout(layout)  
  
 def create\_mj\_sets\_widget(self):  
 self.mj\_sets\_widget = MandelbrotJuliaWidget()  
 self.mj\_sets\_widget.setContentsMargins(0, 0, 0, 0)  
  
 def changed\_set(self):  
 if self.set\_combo\_box.currentIndex() == 1:  
 self.c\_line\_edit.setReadOnly(True)  
 else:  
 self.c\_line\_edit.setReadOnly(False)  
  
 def apply\_pressed(self):  
 self.mj\_sets\_widget.set\_parameters(self.x\_min\_spin\_box.value(), self.x\_max\_spin\_box.value(),  
 self.y\_min\_spin\_box.value(), self.y\_max\_spin\_box.value(),  
 self.image\_size\_combo\_box.currentData()[0],  
 self.image\_size\_combo\_box.currentData()[1],  
 self.diverge\_number\_spin\_box.value(),  
 complex(self.c\_line\_edit.text().replace(" ", "")),  
 self.set\_combo\_box.currentIndex())  
  
  
class MandelbrotJuliaWidget(QWidget):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.set\_parameters()  
  
 def paintEvent(self, event):  
 super().paintEvent(event)  
  
 painter = QPainter(self)  
  
 painter.drawPixmap(self.rect(), QPixmap(self.q\_image))  
  
 def set\_parameters(self, x\_min=-2, x\_max=2, y\_min=-2, y\_max=2,  
 width=400, height=400, diverge\_number=100, c=-0.8 + 0.156j,  
 index=0):  
 # Range of x-values  
 self.x\_min = x\_min  
 self.x\_max = x\_max  
  
 # Range of y-values  
 self.y\_min = y\_min  
 self.y\_max = y\_max  
  
 # Calculate the range  
 self.x\_range = x\_max - x\_min  
 self.y\_range = y\_max - y\_min  
  
 # Image size  
 self.width, self.height = width, height  
  
 # Image scale values  
 self.x\_scale = float(self.x\_range) / self.width  
 self.y\_scale = float(self.y\_range) / self.height  
  
 # Diverge parameter  
 self.num = diverge\_number  
  
 # Julia parameter  
 self.c = c  
  
 self.function\_for\_set = self.func\_array[index]  
  
 data = np.fromfunction(np.vectorize(self.get\_color\_from\_pixel), (self.width, self.height))  
 self.q\_image = QtGui.QImage(data, data.shape[0], data.shape[1], QtGui.QImage.Format\_RGB32)  
  
 self.update()  
  
 @staticmethod  
 def rgba\_to\_hex(r, g, b, a):  
 lst = [r, g, b, a]  
 lst = [255 if x > 255 else x for x in lst]  
 lst = [0 if x < 0 else x for x in lst]  
 r, g, b, a = lst  
 return int('0x{:02x}{:02x}{:02x}{:02x}'.format(a, r, g, b), 16)  
  
 def get\_color\_from\_pixel(self, y, x):  
 z = (self.x\_min + x \* self.x\_scale) + (self.y\_max - y \* self.y\_scale) \* 1j  
  
 # Put it into the mandelbrot function  
 count = self.function\_for\_set(self, z)  
  
 set\_color = self.rgba\_to\_hex(0, 0, 0, 0)  
  
 HSV = (255, 3 \* count, 255)  
 hue, sat, value = HSV  
 hsv = QColor.fromHsv(hue, sat, value, 255)  
 r, g, b = hsv.red(), hsv.green(), hsv.blue()  
 diverge\_color = self.rgba\_to\_hex(r, g, b, 0)  
  
 return set\_color if count == self.num else diverge\_color  
  
 def mandelbrot(self, z):  
 *""" Runs the process num times  
 and returns the 'diverge' count """* count = 0  
  
 # Define z1 as z  
 z\_n = z  
  
 # Iterate num times  
 while count <= self.num:  
 # Check for divergence  
 if abs(z\_n) > 2.0:  
 # Return the step it diverged on  
 return count  
  
 # Iterate z  
 z\_n = z\_n \* z\_n + z  
 count += 1  
  
 # If z hasn't diverged by the end  
 return self.num  
  
 def julia(self, z):  
 *""" Runs the process num times  
 and returns the 'diverge' count """* count = 0  
  
 # Define z1 as z  
 z\_n = z  
  
 # Iterate num times  
 while count <= self.num:  
 # Check for divergence  
 if abs(z\_n) > 2.0:  
 # Return the step it diverged on  
 return count  
  
 # Iterate z  
 z\_n = z\_n \* z\_n + self.c  
 count += 1  
  
 # If z hasn't diverged by the end  
 return self.num  
  
 func\_array = (julia, mandelbrot)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 import sys  
  
 app = QApplication(sys.argv)  
 window = MainWindow()  
 window.show()  
 sys.exit(app.exec\_())

**Кафедра ВВТиС**

**Студент** Сиротин А.Е.

**Руководитель базы практической подготовки:** Ямилева А.М.

**Дневник практики**

| № | Дата | Содержание выполненной работы | Подпись руководителя |
| --- | --- | --- | --- |
|  | 06.06.22 – 09.06.22 | Изучение основных конструкций и коллекций Python. Выполнение заданий из курса [1] |  |
|  | 10.06.22 – 13.06.22 | Изучение функций в Python. Выполнение заданий из курса [1] |  |
|  | 14.06.22 – 16.06.22 | ООП и продвинутые возможности Python. Выполнение заданий из курса [2] |  |
|  | 17.06.22 – 18.06.22 | Изучение массивов библиотеки NumPy. Выполнение заданий из курса [3] |  |
|  | 20.06.22 – 22.06.22 | Решение математических задач с использованием библиотек NumPy и SciPy |  |
|  | 23.06.22 – 25.06.22 | Изучение библиотеки PyQt |  |
|  | 25.06.22 – 30.06.22 | Разработка оконного приложения на Python и PyQt по индивидуальному заданию |  |
|  | 01.07.22 – 02.07.22 | Подготовка и защита отчета |  |