Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Уфимский государственный авиационный технический университет»

Факультет ИРТ
Кафедра ВВТиС

ОТЧЕТ
по учебной практике

Группа ПМ-353	Фамилия И.О.	Подпись	Дата	Оценка
Студент	Шамаев И.Р.			
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: Шамаев Ильдар Группа: ПМ-353

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: реализация уравнения теплопроводности для стационарного двумерного случая.

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.
3. 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.
4. Маккинни, У. Python и анализ данных / У. Маккинни ; перевод с английского А. А. Слинкина. — Москва : ДМК Пресс, 2020. — 540 с. — Режим доступа: https://e.lanbook.com/book/131721, для авториз. пользователей.
5. Хилл, К. Научное программирование на Python / К. Хилл ; перевод с английского А. В. Снастина. — Москва : ДМК Пресс, 2021. — 646 с. — Режим доступа: https://e.lanbook.com/book/241031, для авториз. пользователей.
6. Дауни, А. Б. Изучение сложных систем с помощью Python / А. Б. Дауни ; перевод с английского Д. А. Беликова. — Москва : ДМК Пресс, 2019. — 160 с. — Режим доступа: https://e.lanbook.com/book/131701, для авториз. пользователей.

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

Руководитель от университета _____ Ямилева А.М.

Принял к исполнению _____ Шамаев И.Р.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Учебная практика имеет важное значение в подготовке квалифицированного специалиста.

Цель работы − применить язык программирования Python для решения математических задач.

Для достижения данной цели необходимо решить следующие задачи.

1. Изучить язык программирования Python, в том числе библиотеки NumPy, Matplotlib, Scipy и PyQt.
2. Применить данные библиотеки для решения математических задач.

Практика проходила на кафедре высокопроизводительных вычислительных технологий и систем (ВВТиС).

Кафедра высокопроизводительных вычислительных технологий и систем (ВВТиС) является кафедрой факультета информатики и робототехники.

Основными направлениями деятельности кафедры ВВТиС являются:

1) организация обучения студентов, магистрантов, аспирантов, научных работников и преподавателей работе на высокопроизводительных вычислительных комплексах;
2) выполнение научно-исследовательских работ в области математического моделирования технических объектов и технологических процессов с применением высокопроизводительной вычислительной техники;
3) проведение исследований в области 3D-визуализации и виртуальной реальности;
4) разработка новых методов симметрийного анализа дифференциальных уравнений и их применение в задачах математического моделирования.

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 ЯЗЫК ПРОГРАММИРОВАНИЯ PYTHON

Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. В то же время стандартная библиотека включает большой объём полезных функций.

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++.

Python пользуется популярностью во многих областях: его можно использовать для анализа данных и машинного обучения, бэкенда, веб-разработки, системного администрирования и игр.

У Python много готовых библиотек для решения математических задач. Библиотеками в программировании называют инструменты для решения конкретных типов задач. Несколько примеров популярных библиотек для Python:

- **NumPy, Matplotlib, SciPy**. Библиотеки для работы с искусственным интеллектом и машинным обучением. Используются для сложных математических вычислений.
- **Pygame**. Библиотека для создания небольших игр и мультимедийных приложений.
- **Pandas**. Библиотека для работы с большими данными.
- **SQLAlchemy**. Библиотека для работы с базами данных.
- **Django**, **Flask**. Библиотеки для разработки серверной части приложений (Создание веб-приложения).

1.2 НАУЧНЫЕ БИБЛИОТЕКИ PYTHON

NumPy это open-source модуль для python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций. Они объединяются в высокоуровневые пакеты. Они обеспечивают функционал, который можно сравнить с функционалом MatLab. NumPy (Numeric Python) предоставляет базовые методы для манипуляции с большими массивами и матрицами. SciPy (Scientific Python) расширяет функционал NumPy огромной коллекцией полезных алгоритмов, таких как минимизация, преобразование Фурье, регрессия, и другие прикладные математические техники.

SciPy — это библиотека для языка Python, основанная на расширении NumPy, но для более глубоких и сложных научных вычислений, анализа данных и построения графиков. SciPy в основном написана на Python и частично на языках C, C++ и Fortran, поэтому отличается высокой производительностью и скоростью работы.

Отличия SciPy от NumPy. NumPy — это еще одна научная библиотека, на базе которой реализована SciPy. Ее особенность — использование специальных структур данных — многомерных массивов, в которых хранится информация. С этим типом данных также работает SciPy. NumPy содержит данные массивов и такие операции, как индексация, сортировка и т. д., а SciPy состоит из числового кода.

Основные различия между библиотеками:

- в SciPy гораздо больше функций и методов, чем в NumPy;
- NumPy ориентирована на базовые вычисления и простую работу с матрицами, SciPy предназначена для глубокого научного анализа;
- NumPy не имеет дополнительных зависимостей, вместе с библиотекой не нужно ничего устанавливать. SciPy требует установки NumPy для корректной работы.

SciPy расширяет возможности NumPy, а некоторые частые действия в ней

реализованы как отдельные функции, поэтому библиотека сильно упрощает работу со сложными задачами с использованием продвинутой математики. При этом для ряда более простых задач она избыточна. Будет достаточно NumPy или других библиотек.

1.3 БИБЛИОТЕКА PYQT

PyQt — это библиотека Python для создания приложений с графическим интерфейсом с помощью инструментария Qt. Созданная в Riverbank Computing, PyQt является свободным ПО (по лицензии GPL) и разрабатывается с 1999 года. Последняя версия PyQt6 — на основе Qt 6 — выпущена в 2021 году, и библиотека продолжает обновляться. Это руководство можно также использовать для PySide2, PySide6 и PyQt5.

Сегодня используются две основные версии: PyQt5 на основе Qt5 и PyQt6 на основе Qt6. Обе почти полностью совместимы, за исключением импорта и отсутствия поддержки некоторых продвинутых модулей из Qt6. В PyQt6 вносятся изменения в работу пространств имён и флагов, но ими легко управлять.

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

- существующий набор виджетов графического интерфейса;
- стили виджетов;
- доступ к базам данных с помощью SQL (ODBC, MySQL, PostgreSQL, Oracle);
- QScintilla, основанный на Scintilla виджет текстового редактора;
- поддержку интернационализации (i18n);
- парсер XML;
- поддержку SVG;
- интеграцию с WebKit, движком рендеринга HTML;
- поддержку воспроизведения видео и аудио.

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.

1.4 УРАВНЕНИЕ ТЕПЛОПРОВОДНОСТИ НА ДВУМЕРНОЙ ОБЛАСТИ

Запишем уравнение Лапласа $\frac{\partial^2 T}{\partial x^3}+\frac{\partial^2 T}{\partial y^2}=0$. Это двухмерное уравнение Лапласа, с помощью которого рассчитаем распределение температуры внутри конечного объема. В уравнении T – температура, x, y- координаты.

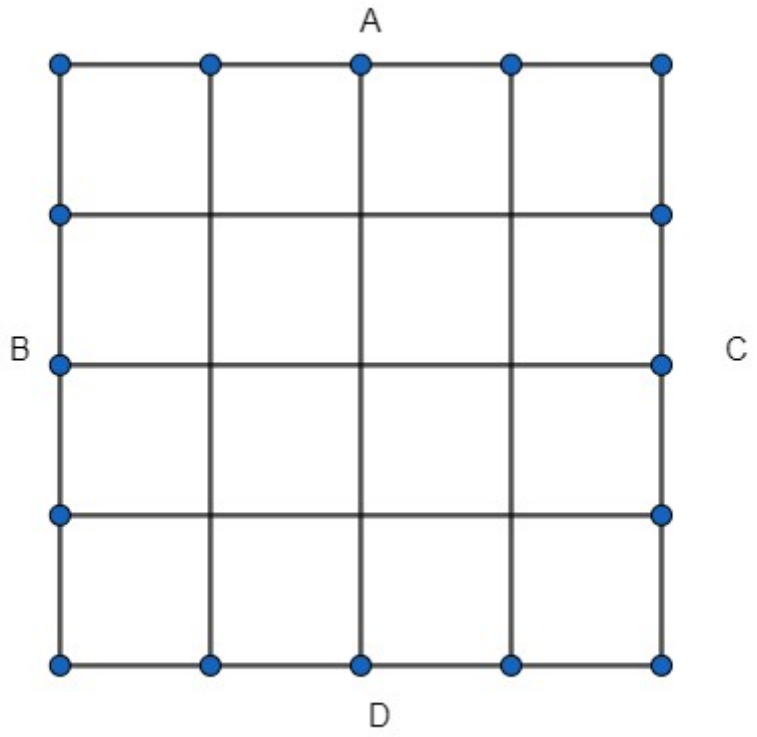

Рисунок 1 – Постановка задачи

На рисунке 1 изображена конечная секция, в которой мы должны найти

узлы сетки, внутри контура. Определим граничные условия на гранях A, B, C, D и перейдем к расчету температуры в каждом узле сетки. Примем за x грань D=10 см, за y, грань B=10 см. Разделим нашу область на равные ячейки сетки, где шаг по сетке будет равен Δx =Δy= 1 см. и мы должны решить уравнение в каждом узле как показано на рисунке 2

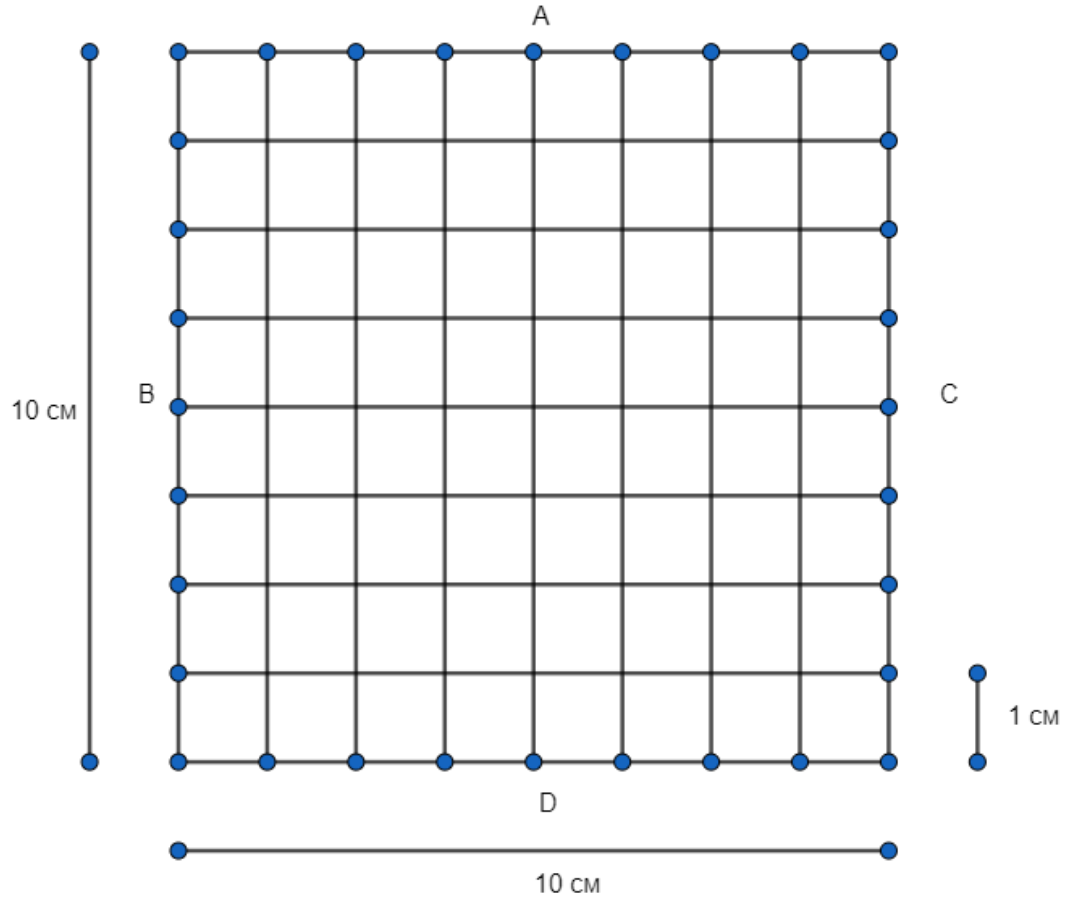

Рисунок 2 – Определение сетки внутри контура

Метод EDM интерполирует значение, которые между боковыми значениями. Тогда температура на каждом узле сетки будет рассчитываться как

$$\frac{T_{i+1,j}-2T_{i,j}+T_{i-1,j}}{\Delta x^2}+\frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{\Delta y^2}=0$$

Упростим уравнение расчёта температуры и оно будет выглядеть следующим образом: $T_{i,j}=\frac{1}{4}\left(T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1}\right)$. Теперь мы должны

реализовать это уравнение в коде python

2 ПРАКТИЧЕСКАЯ ЧАСТИ

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)
- строки и списки
- функции
- коллекции – генераторы с itertools
- ООП (Объектно-Ориентированное Программирование)
- работа с файлами
- встроенные библиотеки

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

2.2 БИБЛИОТЕКИ PYTHON

2.2.1 MATPLOTLIB

Matplotlib — библиотека на языке программирования Python для визуализации данных двумерной (2D) графикой (3D графика также поддерживается). Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях.

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

В процессе изучения были использованы различные возможности библиотеки Matplotlib. Например, была построена поверхность, заданная параметрически $x(u,v)$, $y(u,v)$, $z(u,v)$, с помощью plot. (Рисунок).

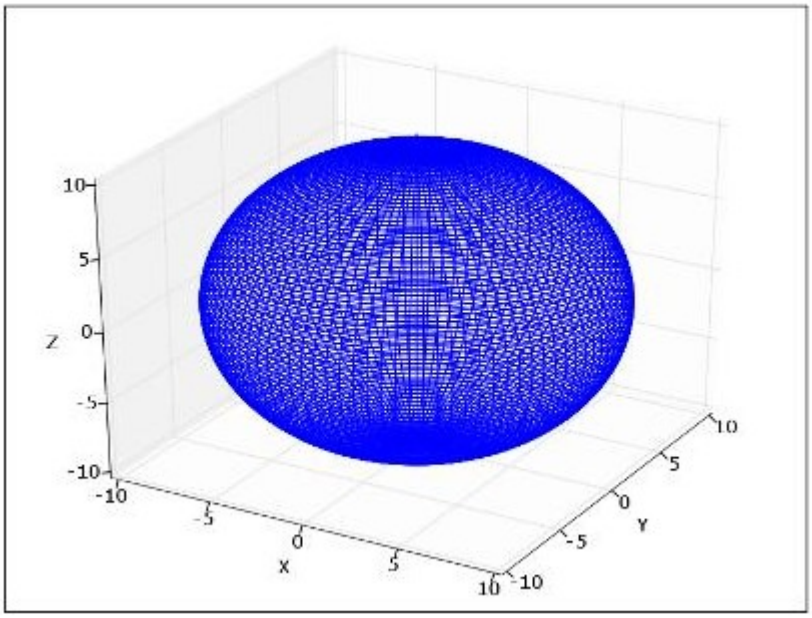

Рисунок 3 – График поверхности, заданной параметрически

2.2.2 NUMPY

NumPy — библиотека с открытым исходным кодом для Python, реализующая множество математических операций для работы с векторами, матрицами и массивами. Важное преимущество Numpy перед собственной реализацией массивов (например, на списках) – это векторные операции, которые происходят гораздо быстрее, последовательных.

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 был реализован клеточный автомат «Коралл». Пример работы программы изображен на рисунке 2.

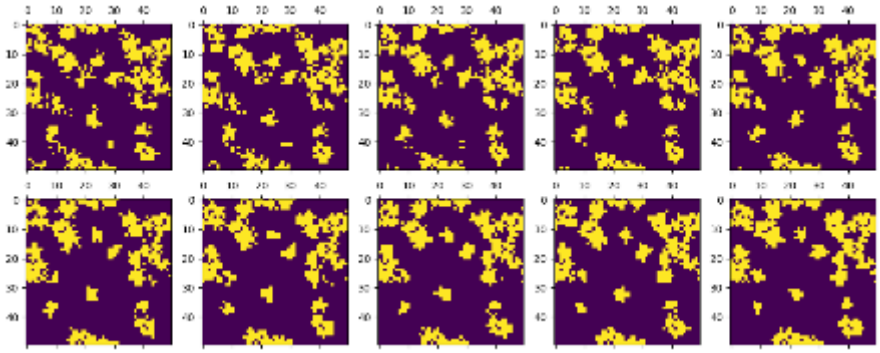

Рисунок 4 – Пример работы клеточного автомата «Коралл»

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:

$$\begin{cases} u'' + au' + bu = f(x), x \in (0, L) \\ \gamma u' + \beta u = \alpha, x = (0, L) \end{cases} \quad (1)$$

Конечно–разностная схема имеет вид:

$$u_{i-1}\left(\frac{1}{h^2} - \frac{a}{2h}\right) + u_i\left(b - \frac{2}{h^2}\right) + u_{i+1}\left(\frac{1}{h^2} + \frac{a}{2h}\right) = f_i, i = 1..N-1$$

Данная схема имеет второй порядок аппроксимации по h.

Аппроксимация краевых условий второго порядка точности по h имеет вид:

$$u_0\left(\beta_1+\frac{\gamma_1}{1-\frac{ah^2}{2}}\left(\frac{bh^2}{2}-\frac{1}{h}\right)\right)+u_1\left(\frac{\frac{\gamma_1}{h}}{1-\frac{ah^2}{2}}\right)=\alpha_1+\frac{\frac{f_i\gamma_1h^2}{2}}{1-\frac{ah^2}{2}}$$

$$u_{N-1}\left(\frac{-\gamma_2}{h}\left(\frac{1}{1+\frac{ah^2}{2}}\right)\right)+u_N\left(\beta_2+\frac{\gamma_2\left(\frac{1}{h}-\frac{bh^2}{2}\right)}{1+\frac{ah^2}{2}}\right)=\alpha_2-\frac{\frac{f_i\gamma_2h^2}{2}}{1+\frac{ah^2}{2}}$$

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:

$$\begin{cases} u''+u=-x, x\in\left(0,\frac{\pi}{2}\right) \\ u=0, x=0 \\ u=0, x=\frac{\pi}{2} \end{cases} \quad (2)$$

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

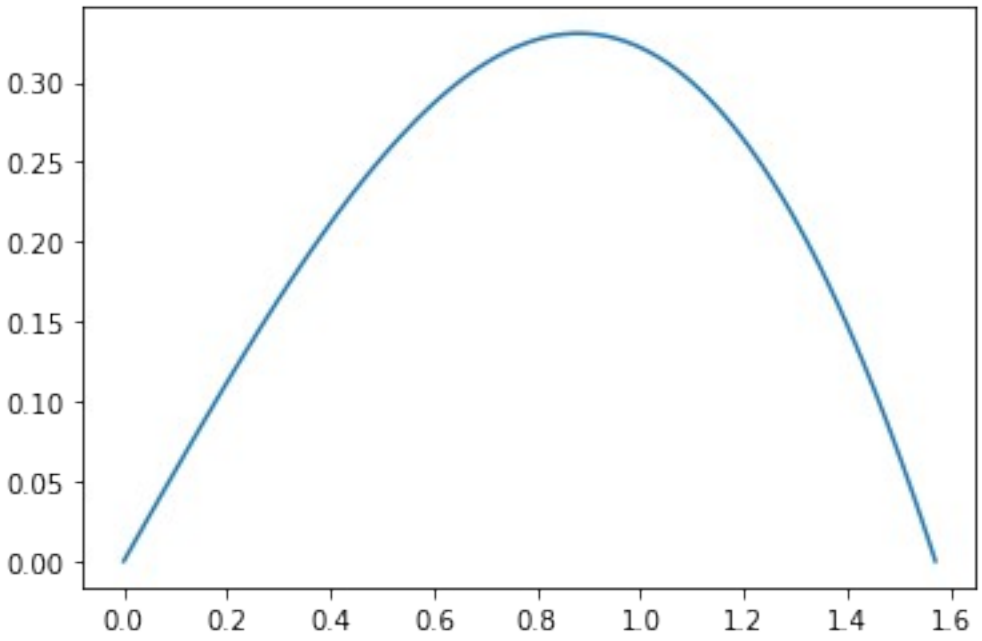

Рисунок 5 – График точного решения краевой задачи (2)

График численного решения задачи можно увидеть на рисунке 4.

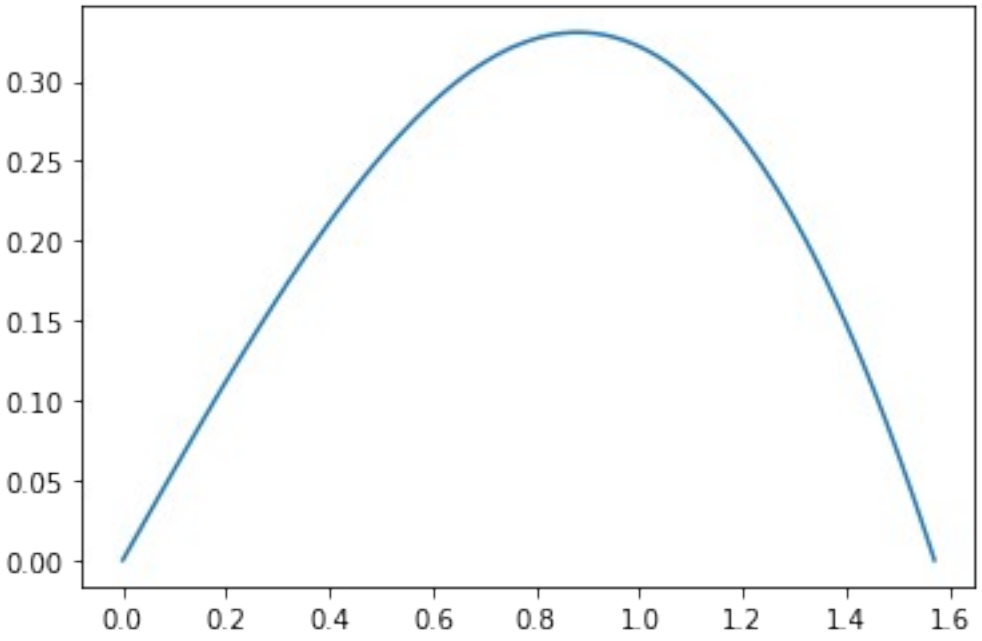

Рисунок 6 – График численного решения краевой задачи (2)

График погрешности краевой задачи можно увидеть на рисунке 5.

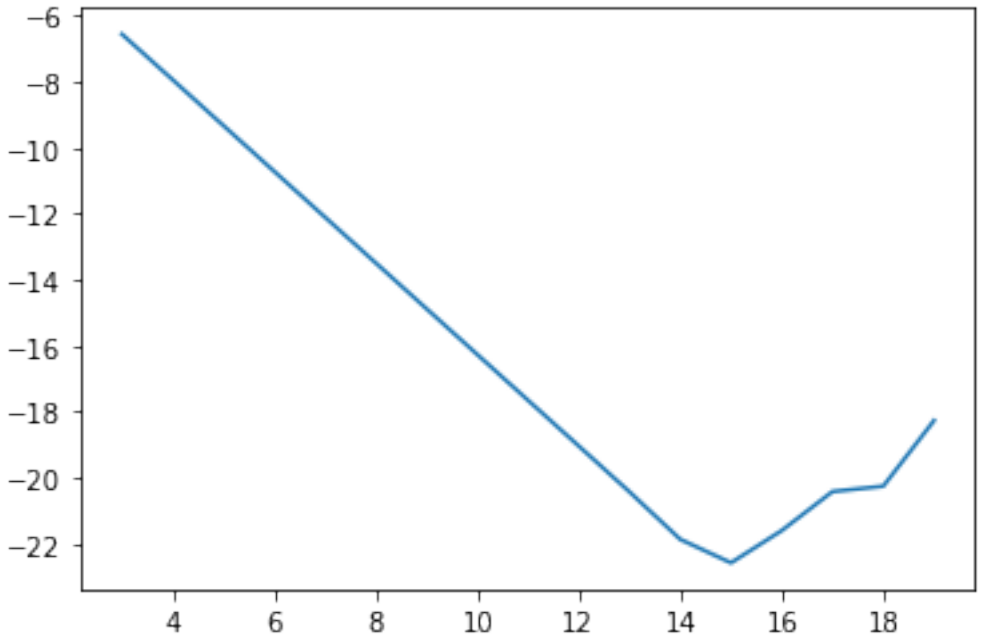

Рисунок 7 – Погрешности краевой задачи (2) в логарифмической норме

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:

$$\begin{cases} a_4 u'''' + a_3 u''' + a_2 u'' + a_1 u' + a_0 u = f(x) \\ u|_{x=0} = \phi_0,\ u|_{x=L} = \phi_1 \\ u'|_{x=0} = \psi_0,\ u'|_{x=L} = \psi_1 \end{cases} \quad (3)$$

Конечно–разностная схема имеет вид:

$$u_{i-2}\left(\frac{a_4}{h^4} - \frac{a_3}{2h^3}\right) + u_{i-1}\left(\frac{-4a_4}{h^4} + \frac{a_3}{h^3} + \frac{a_2}{h^2} - \frac{a_1}{2h}\right) + u_i\left(\frac{6a_4}{h^4} - \frac{2a_2}{h^2} + a_0\right) + u_{i+1}\left(\frac{-4a_4}{h^4} - \frac{a_3}{h^3} + \frac{a_2}{h^2} + \frac{a_1}{2h}\right) + u_{i+2}\left(\frac{a_4}{h^4} + \frac{}{2}\right.$$

Аппроксимация краевых условий первого порядка точности по h имеет вид:

$$u_0 = \phi_0, u_N = \phi_1$$

$$\frac{u_1 - u_0}{h} = \psi_0, \frac{u_N - u_{N-1}}{h} = \psi_1$$

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.

2.4 ПРИЛОЖЕНИЕ НА PYQT5

Приложение разработано с помощью наследования классов PyQt5. Главное окно наследует класс QMainWindow, а область отображения множества – класс SubWindow. Обработка событий реализована в отдельных функциях, что повышает читаемость кода. Таким образом, код приложения структурирован и может быть просто изменен при доработках.

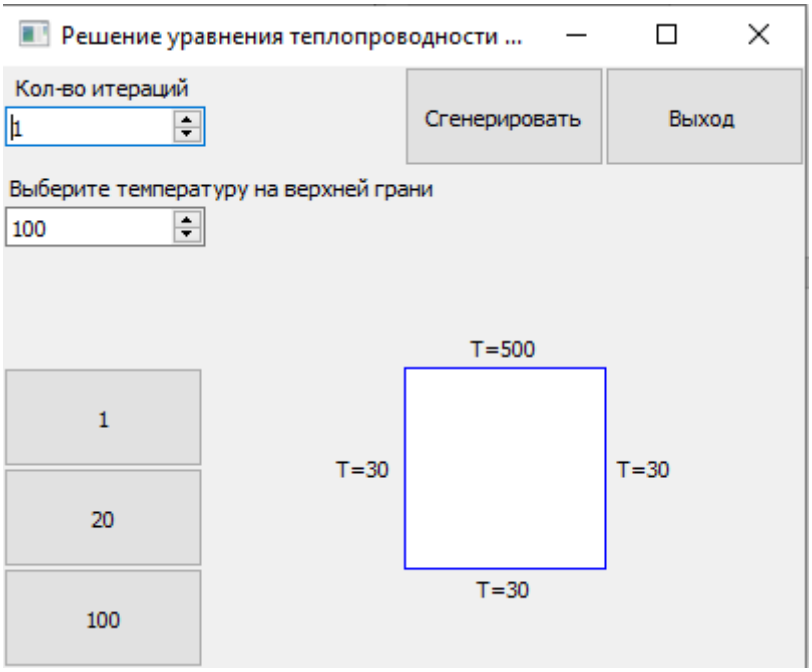

Рисунок 8 – Внешний вид приложения

Изначально температура на верхней грани 500 градусов, по бокам и внизу по 30 градусов. Пользователь может изменить температуру на верхней грани и количество итераций в специальном спинбоксе. После выбора

пользователь может сгенерировать модель по выставленным условиям, нажав на кнопку “Сгенерировать”. Допустим, выберем количество итераций 11 и температуру на верхней грани T=100 градусам и сгенерируем данный случай. На рисунке 9 показана реализация нашего случая.

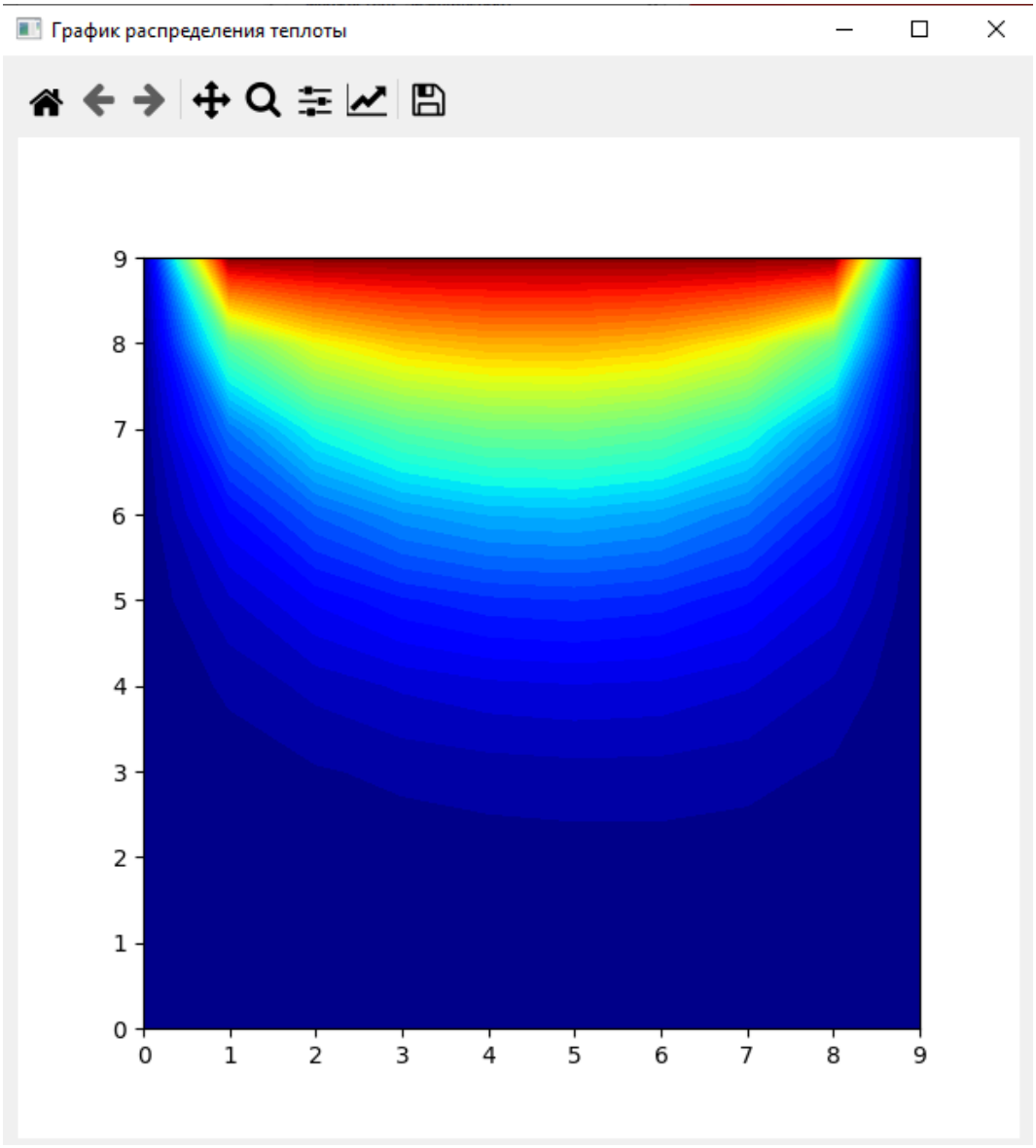

Рисунок 9 – Распределение температуры при 10 итерациях и 100 градусам на верхней грани

Рассмотрим случай при том же количестве итераций, но в этот раз увеличим температуру на верхней грани до 2000 градусов. Реализация данного случая представлена на рисунке 10.

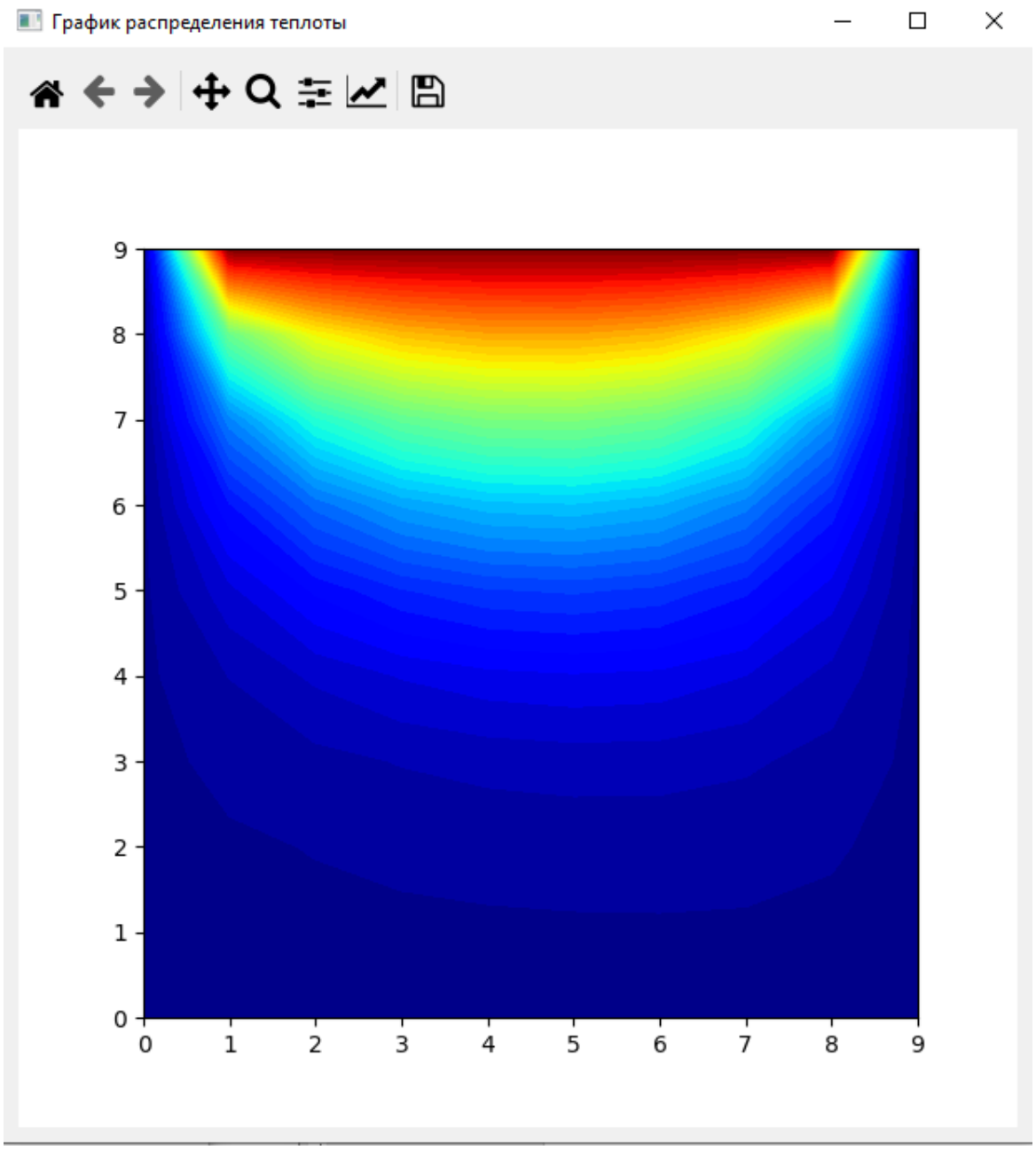

Рисунок 10 – Распределение температуры при 10 итерациях и 2000 градусам на верхней грани

Очевидно, что при более высокой температуре распределение теплоты внутри контура будет интенсивней.

Так же для наглядности, при изменении температуры будет изменяться модель на начальном окне. Это показано на рисунке 11.

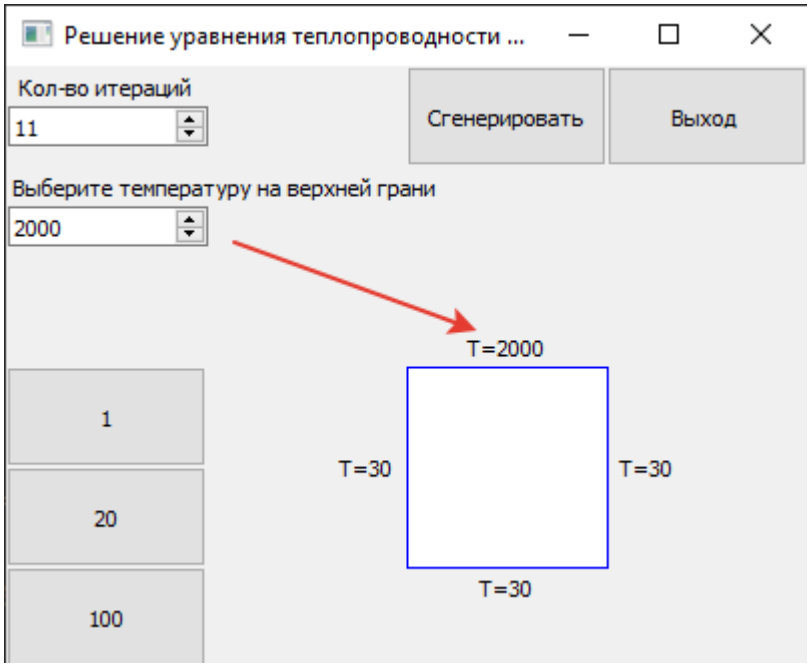

Рисунок 11 – Изменение температуры на мини-модели на начальном окне

В левом нижнем углу реализованы три кнопки со значениями 1, 20, 100 (представляющие собой количество итераций), как показано на рисунке 12. Например, при нажатии на кнопку “1”, будет сгенерирована модель распределения температуры, при текущей установленной температуре в спинбоксе.

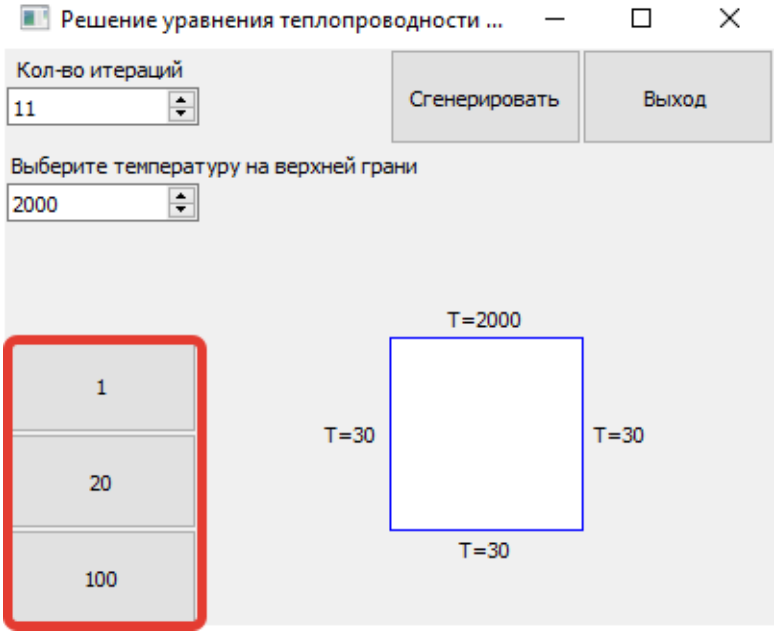

Рисунок 12 – Кнопки, генерирующие модель, с указанными на них количеством итераций

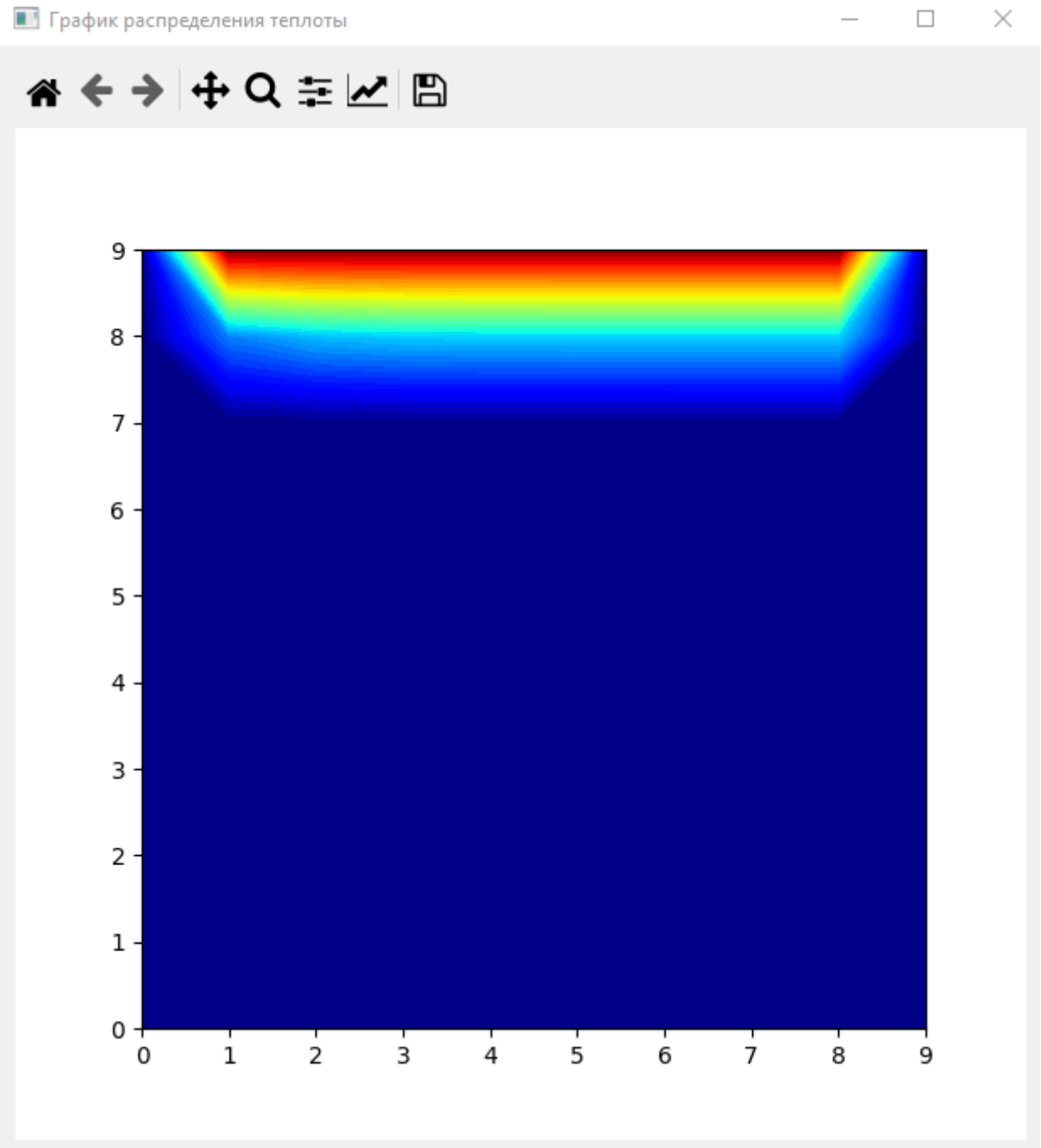

Рисунок 13 – Реализация кнопки, при 1 итерации

Так же во втором окне реализован toolbox, с помощью которого можно:

1) Изменять вид графика
2) Приближать график
3) Растянуть график по оси x или y
4) Вернуть график в исходное состояние
5) Сохранить сгенерированный график в виде фото на компьютере

ЗАКЛЮЧЕНИЕ

В ходе учебной практики были выполнены все поставленные задачи:

Был изучен язык программирования Python, в том числе библиотеки NumPy и Matplotlib на примере онлайн-курсов [1,2,3].

Полученные знания были применены для решения математических задач, а именно, с помощью библиотек NumPy, Matplotlib и SciPy:

- Построено решение для краевых задач для ОДУ 2-го и 4-го порядков
- Реализован клеточный автомат «Коралл»

Была изучена библиотека PyQt. С ее помощью реализовано приложение, которое рассчитывает и визуализирует распределение теплоты в конечном объеме стационарного двумерного случая

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

(обязательное)

```
import sys
import numpy as np
from PyQt5 import QtWidgets, QtCore
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as
FigureCanvas, NavigationToolbar2QT as NavigationToolbar
from matplotlib.figure import Figure
import matplotlib.pyplot as plt
from tkinter import Tk, Canvas, Frame, BOTH
from PyQt5.QtWidgets import QWidget, QApplication
from PyQt5.QtGui import QPainter, QColor, QBrush


maxInter = 1000
Ttop = 500

class MplCanvas(FigureCanvas):

    def __init__(self, parent=None, width=5, height=4, dpi=100):
        fig = Figure(figsize=(width, height), dpi=dpi)
        self.axes = fig.add_subplot(111)
        super(MplCanvas, self).__init__(fig)


class MainWindow(QtWidgets.QWidget, Frame):
     def __init__(self, parent=None):
         QtWidgets.QWidget.__init__(self, parent)
         super(MainWindow, self).__init__()
         self.resize(400, 300)
         self.setWindowTitle("Решение уравнения теплопроводности на двумерной
области")

         #maxInter = 100
         # self.is_animate=False
         self.initUI()



     def initUI(self):
         global maxInter
         global Ttop
         self.inter=1
         self.spin=QtWidgets.QSpinBox(self)
         self.spin.setGeometry(1, 20, 100, 20)
         self.spin.setRange(1, 100)
         self.spin.setValue(self.inter)
         self.spin.setSingleStep(10)
        #maxInter=self.spin.value()

         self.Ttop=1
         self.spin1 = QtWidgets.QSpinBox(self)
         self.spin1.setGeometry(1, 70, 100, 20)
         self.spin1.setRange(100, 2000)
         self.spin1.setValue(self.Ttop)
         self.spin1.setSingleStep(100)

         # Button1
         self.button1 = QtWidgets.QPushButton(self)
```

```
        self.button1.setGeometry(200, 0, 100, 50)
        self.button1.setText('Сгенерировать')

        self.label = QtWidgets.QLabel(self)
        self.label.setGeometry(0, 0, 100, 20)
        self.label.setText('Кол-во итераций')
        self.label.setAlignment(QtCore.Qt.AlignCenter)

        self.button2 = QtWidgets.QPushButton(self)
        self.button2.setGeometry(300, 0, 100, 50)
        self.button2.setText('Выход')

        self.button3 = QtWidgets.QPushButton(self)
        self.button3.setGeometry(0, 150, 100, 50)
        self.button3.setText('1')

        self.button4 = QtWidgets.QPushButton(self)
        self.button4.setGeometry(0, 200, 100, 50)
        self.button4.setText('20')

        self.button5 = QtWidgets.QPushButton(self)
        self.button5.setGeometry(0, 250, 100, 50)
        self.button5.setText('100')

        self.labelT1 = QtWidgets.QLabel(self)
        self.labelT1.setGeometry(200, 130, 100, 20)
        self.labelT1.setText('T=500')
        self.labelT1.setAlignment(QtCore.Qt.AlignCenter)

        self.labelT2 = QtWidgets.QLabel(self)
        self.labelT2.setGeometry(200, 250, 100, 20)
        self.labelT2.setText('T=30')
        self.labelT2.setAlignment(QtCore.Qt.AlignCenter)

        self.labelT3 = QtWidgets.QLabel(self)
        self.labelT3.setGeometry(130, 190, 100, 20)
        self.labelT3.setText('T=30')
        self.labelT3.setAlignment(QtCore.Qt.AlignCenter)

        self.labelT4 = QtWidgets.QLabel(self)
        self.labelT4.setGeometry(270, 190, 100, 20)
        self.labelT4.setText('T=30')
        self.labelT4.setAlignment(QtCore.Qt.AlignCenter)

        self.labelT = QtWidgets.QLabel(self)
        self.labelT.setGeometry(0, 50, 220, 20)
        self.labelT.setText('Выберите температуру на верхней грани')
        self.labelT.setAlignment(QtCore.Qt.AlignCenter)


        # Sub Window
        self.sub_window = SubWindow()

        # Button Event
        self.button3.clicked.connect(self.clicked1)
        self.button4.clicked.connect(self.clicked4)
        self.button5.clicked.connect(self.clicked2)
        #self.button1.clicked.connect(self.sub_window.show)
        self.button1.clicked.connect(self.clicked3)
        self.button1.clicked.connect(self.clicked5)
        self.button2.clicked.connect(QtWidgets.qApp.quit)
        #self.button3.clicked.connect(self.clicked1)
```

```
        #self.button4.clicked.connect(self.clicked2)

    def paintEvent(self, e):
        qp = QPainter()
        qp.begin(self)
        self.drawRectangles(qp)
        qp.end()


    def drawRectangles(self, qp):
        col = QColor(0, 0, 0)
        col.setNamedColor('blue')
        qp.setPen(col)

        qp.setBrush(QColor(255, 255, 255))
        qp.drawRect(200, 150, 100, 100)

    def clicked5(self):

        global Ttop
        Ttop = self.spin1.value()
        self.sub_window.__init__()
        self.labelT1.setText(f'T={Ttop}')
        print("Ttop: ", Ttop)
        return Ttop


    def clicked3(self):

        global maxInter
        maxInter = self.spin.value()
        self.sub_window.__init__()
        print(maxInter)
        return maxInter

    def clicked1(self):

        global maxInter
        maxInter=1
        self.sub_window.__init__()
        print(maxInter)
        return maxInter

    def clicked4(self):

        global maxInter
        maxInter=20
        self.sub_window.__init__()
        print(maxInter)
        return maxInter

    def clicked2(self):

        global maxInter
        maxInter=1000
        self.sub_window.__init__()
        print(maxInter)
        return maxInter

def main():
    root = Tk()
    ex = MainWindow()
```

```
    root.geometry("330x220+300+300")
    root.mainloop()

class SubWindow(QtWidgets.QMainWindow):
   #  def __init__(self):
    #     super(SubWindow, self).__init__()
     #    self.resize(400, 300)


     def __init__(self, *args, **kwargs):
         super(SubWindow, self).__init__(*args, **kwargs)
         self.setWindowTitle("График распределения теплоты")

         sc=self.canvas = MplCanvas(self, width=6, height=6, dpi=100)
         self.setCentralWidget(self.canvas)
         # Create toolbar, passing canvas as first parament, parent (self, the
MainWindow) as second.
         toolbar = NavigationToolbar(sc, self)

         layout = QtWidgets.QVBoxLayout()
         layout.addWidget(toolbar)
         layout.addWidget(sc)

         # Create a placeholder widget to hold our toolbar and canvas.
         widget = QtWidgets.QWidget()
         widget.setLayout(layout)
         self.setCentralWidget(widget)

         fout = open('output.txt', 'w')

         #maxIter = 1
         global maxInter
         global Ttop
         print("Количество итераций: ", maxInter)
         lenX = 10
         lenY = 10

         delta = 1


         Tbottom = 30
         Tleft = 30
         Tright = 30

         Tintial = 30
         colorinterpolation = 50
         colorMap = plt.cm.jet

         X, Y = np.meshgrid(np.arange(0, lenX), np.arange(0, lenY))

         T = np.empty((lenX, lenY))
         T.fill(Tintial)

         T[(lenY - 1):, :] = Ttop
         T[:1, :] = Tbottom
         T[:, (lenX - 1):] = Tright
         T[:, :1] = Tleft

         for i in range(0, maxInter):
             for j in range(1, lenX - 1, delta):
                 for t in range(1, lenX - 1, delta):
                     T[j, t] = 0.25 * (T[j + 1][t] + T[j - 1][t] + T[j][t + 1] +
```

```
T[j][t - 1])
                    # self.canvas.axes.grid.axis.plot(X, Y, T)
                     print( T[j, t], file=fout)
        fout.close()
        self.xdata = X
        self.ydata = Y
        #self.canvas.axes.plot(T)
        #self.canvas.axes.plot(self.ydata,self.xdata, 'b')
        #self.title("Распределение температуры")
        self.canvas.axes.contourf(X, Y, T, colorinterpolation, cmap=colorMap)
        #self.canvas.axes.colorbar()
        self.show()
        #self.close()


if __name__ == '__main__':
    app = QtWidgets.QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())
```

ФГБОУ ВО «Уфимский государственный авиационный технический университет»

Кафедра ВВТиС

Студент Шамаев И.Р.

Руководитель базы практической подготовки: Ямилева А.М.

Дневник практики

№	Дата	Содержание выполненной работы	Подпись руководителя
1.	06.06.22 – 09.06.22	Изучение основных конструкций и коллекций Python. Выполнение заданий из курса [1]	
2.	10.06.22 – 13.06.22	Изучение функций в Python. Выполнение заданий из курса [1]	
3.	14.06.22 – 16.06.22	ООП и продвинутые возможности Python. Выполнение заданий из курса [2]	
4.	17.06.22 – 18.06.22	Изучение массивов библиотеки NumPy. Выполнение заданий из курса [3]	
5.	20.06.22 – 22.06.22	Решение математических задач с использованием библиотек NumPy и SciPy	
6.	23.06.22 – 25.06.22	Изучение библиотеки PyQt	
7.	25.06.22 – 30.06.22	Разработка оконного приложения на Python и PyQt по индивидуальному заданию	
8.	01.07.22 – 02.07.22	Подготовка и защита отчета	