МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка программной системы для вычисления и анализа дискретных логарифмов для начальных диапазонов простых чисел

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.04.2024.308-356.ВКР

Научный руководитель,
профессор кафедры СП, д.ф.-м.н.,
доцент
_____ Р.Ж. Алеев

Автор работы,
студент группы КЭ-433
_____ П.А. Бородин

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-433
Бородину Павлу Андреевичу,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)
Разработка программной системы для вычисления и анализа дискретных логарифмов для начальных диапазонов простых чисел.
2. **Срок сдачи студентом законченной работы:** 03.06.2024 г.
3. **Исходные данные к работе**
3.1. Ерусалимский Я.М. Дискретная математика: Теория, задачи, приложения. – М.: Вузовская книга, 2011. – 265 с.
3.2. Тайцлин М.А., Столбоушкин А.П. Математические основания информатики. – Тверь: ТвГУ, 1998. – 377 с.
3.3. Яблонский С.В. Введение в дискретную математику. – Изд. 4-е, стер. – М.: Высшая школа, 2003. – 384 с.
4. **Перечень подлежащих разработке вопросов**
4.1. Составление списка простых чисел из диапазона [2..P].
4.2. Разработка функции для нахождения наименьшего примитивного корня R по модулю простых чисел из диапазона [2..P].

4.3. Разработка функции для нахождения и анализа всех дискретных R-логарифмов по модулю простых чисел из диапазона [2..P].

4.4. Разработка графического приложения, выполняющего все указанные функции.

5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
профессор кафедры СП, д.ф.-м.н., доцент Р.Ж. Алеев

Задание принял к исполнению П.А. Бородин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Создание программы для вычисления дискретных логарифмов для начальных диапазонов простых чисел актуально в связи с необходимостью решения задач, требующих большого количества вычислений, которые очень сложно провести вручную.

Дискретные логарифмы применяются в таких областях как криптография, комбинаторика, теория чисел и для узких исследований в математике. В криптографии дискретные логарифмы используются в схемах асимметричного шифрования, таких как Diffie-Hellman, ElGamal, DSA, ECDSA. Они также играют ключевую роль в криптографических протоколах, таких как TLS (Transport Layer Security) и протоколы обмена ключами. В комбинаторике дискретные логарифмы могут быть использованы для решения различных задач, таких как анализ времени работы алгоритмов, исследование структуры графов и разработка эффективных алгоритмов.

Работа нацелена на изучение возможностей языка программирования Python в сочетании с его графической библиотекой Tkinter для создания интерактивных математических приложений.

Результаты данного исследования способствуют расширению применения Python и Tkinter в области математики и научных вычислений, а также помогают в создании более доступных и интуитивно понятных инструментов для работы с математическими концепциями.

Постановка задачи

Целью выпускной квалификационной работы является разработка программной системы для вычисления и анализа дискретных логарифмов для начальных диапазонов простых чисел. Для достижения поставленной цели необходимо решить следующие задачи:

1) составить список простых чисел из диапазона [2..P];
2) разработать функцию для нахождения наименьшего примитивного корня R по модулю простых чисел из диапазона [2..P];

3) разработать функцию для нахождения и анализа всех дискретных R-логарифмов по модулю простых чисел из диапазона [2..P];

4) разработать графическое приложение со всеми указанными функциями.

Структура и содержание работы

Работа состоит из введения, шести глав, заключения, списка литературы и приложений А, Б. Объем работы составляет 38 страниц, объем списка литературы – 15 источников.

В первой главе «Обзор литературы» осуществлен обзор ключевых работ, используемых в исследовании.

Во второй главе «Теоретическая часть» представлена вся необходимая теория с примерами вычислений.

В третьей главе «Вычисления» подробно представлен процесс вычисления для конкретного числа, для него составлена таблица и построены графики.

В четвертой главе «Разработка модели, проектирование» определены функциональные и нефункциональные требования, представлены диаграммы вариантов использования и деятельности. Спроектированы модели всех окон, задействованных при использовании приложения. Указаны предназначения всех компонентов для каждого окна.

В пятой главе «Программная реализация системы» указаны выбранные технологии и инструменты для разработки системы. Приводится реализация оконного приложения и всех его компонентов.

В шестой главе «Тестирование системы» производятся функциональное тестирование, тестирование приложения на корректность входных и выходных данных, а также анализируется производительность системы.

В приложении А представлены таблица и графики для результатов числа $P = 79$.

В приложении Б представлен листинг всего приложения.

1. ОБЗОР ЛИТЕРАТУРЫ

Для разработки программной системы для вычисления и анализа дискретных логарифмов важно рассмотреть широкий спектр литературных источников, охватывающих теоретические основы, алгоритмы и прикладные аспекты дискретной математики и информатики. В этом разделе представлен обзор ключевых работ, используемых в исследовании.

Теоретические основы и алгоритмы

Работы [8] и [15] являются основополагающими в области дискретной математики. В них подробно рассматриваются основные понятия и методы дискретной математики, предоставляя базу для понимания алгоритмов, используемых в исследовании. В публикации [3] акцентируется внимание на фундаментальных принципах данной области изучения и их применении, что способствует более глубокому пониманию теоретических аспектов задачи.

В книге [13] освещаются важные аспекты математической теории, лежащей в основе алгоритмов, применяемых в информатике. Их работа является ключевым источником для понимания методов обработки информации и алгоритмов, используемых в разработке программной системы.

Работа [5] предоставляет ценные сведения о теоретических аспектах простых чисел и их свойствах, что необходимо для разработки функций, связанных с нахождением простых чисел и их примитивных корней. Для изучения более углубленной информации в области теории чисел, способствующей пониманию числовых методов, используемых в проекте, был использован источник [11].

Алгебраические структуры и логические принципы рассматриваются в книгах [2, 6],что полезно при разработке алгоритмов для вычисления дискретных логарифмов и анализа их свойств. Комбинаторные методы, описанные в изданиях [4, 9], также являются важными инструментами для решения задач в дискретной математике.

Прикладные аспекты

Прикладные аспекты затрагиваются в книгах о компьютерной алгебре и комбинаторике для программистов [7, 10]. Они важны для понимания современных методов компьютерной алгебры, которые могут применяться при разработке программной системы. Акцентируется внимание на практических аспектах комбинаторных алгоритмов, что полезно при реализации и оптимизации разработанных функций.

GAP – свободно распространяемая на условиях лицензии GNU GPL система компьютерной алгебры для вычислительной дискретной алгебры с особым вниманием к вычислительной теории групп. Документация по системе GAP (Groups, Algorithms, and Programming) предоставляет полезные сведения о существующих алгоритмах и их реализации в современных программных системах [1]. Эта информация важна для понимания того, как эффективно реализовать алгоритмы для вычисления дискретных логарифмов.

Обзор литературы показывает, что существует богатая база теоретических и прикладных исследований, которые можно использовать для разработки эффективной программной системы для вычисления и анализа дискретных логарифмов. Источники предоставляют необходимые теоретические знания, алгоритмические подходы и практические методы, которые обеспечат успешное выполнение поставленных задач.

Вывод по первой главе

В первой главе был осуществлен обзор ключевых работ, используемых в исследовании, включающих в себя теоретические основы, алгоритмы и прикладные аспекты дискретной математики и информатики.

2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Наименьший примитивный корень

Примитивный корень по модулю p – это число, у которого при возведении его в степени от 1 до p в остатке получаются числа от 1 до $p-1$. Соответственно, наименьший примитивный корень – первый из примитивных корней.

Например: проверка числа 3 по модулю 7 на примитивный корень представлена на рисунке 1.

$$
\begin{array}{rcrcrcrcl}
3^1 & = & 3 & = & 3^0 \times 3 & \equiv & 1 \times 3 & = & 3 \equiv 3 \pmod 7 \\
3^2 & = & 9 & = & 3^1 \times 3 & \equiv & 3 \times 3 & = & 9 \equiv 2 \pmod 7 \\
3^3 & = & 27 & = & 3^2 \times 3 & \equiv & 2 \times 3 & = & 6 \equiv 6 \pmod 7 \\
3^4 & = & 81 & = & 3^3 \times 3 & \equiv & 6 \times 3 & = & 18 \equiv 4 \pmod 7 \\
3^5 & = & 243 & = & 3^4 \times 3 & \equiv & 4 \times 3 & = & 12 \equiv 5 \pmod 7 \\
3^6 & = & 729 & = & 3^5 \times 3 & \equiv & 5 \times 3 & = & 15 \equiv 1 \pmod 7 \\
3^7 & = & 2187 & = & 3^6 \times 3 & \equiv & 1 \times 3 & = & 3 \equiv 3 \pmod 7
\end{array}
$$

Рисунок 1 – Пример проверки числа 3 по модулю 7
на примитивный корень

Здесь видно, что остатки в периоде, равные 3, 2, 6, 4, 5, 1, образуют перестановку всех ненулевых остатков по модулю 7, подразумевая, что 3 действительно является примитивным корнем по модулю 7. Подробнее о свойствах примитивных корней говорится в работе [12].

Дискретный R-логарифм

Дискретный R-логарифм – это такие числа x и y, которые являются решениями равенства вида $R^x \equiv y(mod\ p)$.

Решение задачи дискретного логарифмирования состоит в нахождении некоторого целого неотрицательного числа x, удовлетворяющего уравнению $g^x = a$. Если оно разрешимо, у него должно быть, хотя бы одно натуральное решение, не превышающее порядок группы. Это сразу дает

грубую оценку сложности алгоритма поиска решений сверху – алгоритм полного перебора нашел бы решение за число шагов не выше порядка данной группы.

Подробнее задачи дискретного логарифмирования представлены в работе [14].

Например: нахождение дискретных логарифмов для $3^x \equiv y(mod\ 17)$ представлено на рисунке 2.

$3^1 \equiv 3$	$3^5 \equiv 5$	$3^9 \equiv 14$	$3^{13} \equiv 12$
$3^2 \equiv 9$	$3^6 \equiv 15$	$3^{10} \equiv 8$	$3^{14} \equiv 2$
$3^3 \equiv 10$	$3^7 \equiv 11$	$3^{11} \equiv 7$	$3^{15} \equiv 6$
$3^4 \equiv 13$	$3^8 \equiv 16$	$3^{12} \equiv 4$	$3^{16} \equiv 1$

Рисунок 2 – Пример нахождения всех дискретных логарифмов для $3^x \equiv y(mod\ 17)$

Зная примитивный корень можно получить все дискретные логарифмы. Для этого примитивный корень возводится в степени от 1 до $p - 1$. Также можно убедиться, что примитивный корень был определен верно (в остатке получаются числа от 1 до $p - 1$).

Вывод по второй главе

Во второй главе была рассмотрена теоретическая часть основных математических определений. Был рассмотрен и описан принцип нахождения примитивных корней и дискретных логарифмов.

3. ВЫЧИСЛЕНИЯ

Всю изложенную выше теорию можно преобразовать в программный код и разобрать пример для конкретного числа.

Пример вычисления для $P = 23$.

Фиксируется простое число $P = 23$. Найти наименьший примитивный корень по модулю 23 можно, используя функцию нахождения наименьшего примитивного корня (листинг 1). Для числа 23 наименьший примитивный корень равен 5.

Листинг 1 – Функция нахождения наименьшего примитивного корня

```
def min_r(p, exam):
    for i in range(2, p, +1):
        new_exam = exam.copy()
        for j in range(1, p+1, +1):
            ost = i**j % p
            try:
                new_exam.remove(ost)
            except ValueError:
                pass
        if new_exam == []:
            r = i
            return r
```

Найти все дискретные 5-логарифмы по модулю 23, $d(k)$ и $|d(k)|$ можно, используя функцию нахождения всех дискретных логарифмов (листинг 2). Где $d(k)$ – разность k и степени числа, а $|d(k)|$ – модуль данной разности. Для числа 23 результаты приведены на рисунке 3.

Листинг 2 – Функция нахождения всех дискретных логарифмов

```
def log(p, r):
    dk_min = 1000
    dk_max = -1000
    dk_mod_min = 1000
    dk_mod_max = -1000
    for i in range(0, p-1, +1):
        ost = r**i % p
        dk = ost - i
        if dk < dk_min:
            dk_min = dk
        elif dk > dk_max:
            dk_max = dk
        dk_mod = abs(ost - i)
        if dk_mod < dk_mod_min:
            dk_mod_min = dk_mod
        elif dk_mod > dk_mod_max:
            dk_mod_max = dk_mod
```

[k, степень, d(k), |d(k)|]:

[1, 0, 1, 1], [5, 1, 4, 4], [2, 2, 0, 0], [10, 3, 7, 7], [4, 4, 0, 0], [20, 5, 15, 15],

[8, 6, 2, 2], [17, 7, 10, 10], [16, 8, 8, 8], [11, 9, 2, 2], [9, 10, -1, 1], [22, 11, 11, 11],

[18, 12, 6, 6], [21, 13, 8, 8], [13, 14, -1, 1], [19, 15, 4, 4], [3, 16, -13, 13],

[15, 17, -2, 2], [6, 18, -12, 12], [7, 19, -12, 12], [12, 20, -8, 8], [14, 21, -7, 7].

[d(k)_min, d(k)_max, |d(k)_min|, |d(k)_max |]:

[-13, 15, 0, 15].

Рисунок 3 – Все дискретные логарифмы для числа 23

Составляется окончательная таблица. Результаты вычислений для числа 23 представлены в таблице 1.

Таблица 1 – Результаты для $P = 23$

По возрастанию k				**По возрастанию степени**			
k	**Степень**	**d(k)**	**|d(k)|**	**Степень**	**k**	**d(k)**	**|d(k)|**
1	0	1	1	0	1	1	1
2	2	0	0	1	5	4	4
3	16	-13	13	2	2	0	0
4	4	0	0	3	10	7	7
5	1	4	4	4	4	0	0
6	18	-12	12	5	20	15	15
7	19	-12	12	6	8	2	2
8	6	2	2	7	17	10	10
9	10	-1	1	8	16	8	8
10	3	7	7	9	11	2	2
11	9	2	2	10	9	-1	1
12	20	-8	8	11	22	11	11
13	14	-1	1	12	18	6	6
14	21	-7	7	13	21	8	8
15	17	-2	2	14	13	-1	1
16	8	8	8	15	19	4	4
17	7	10	10	16	3	-13	13
18	12	6	6	17	15	-2	2
19	15	4	4	18	6	-12	12
20	5	15	15	19	7	-12	12
21	13	8	8	20	12	-8	8
22	11	11	11	21	14	-7	7
Минимум		-13	0	**Минимум**		-13	0
Максимум		15	15	**Максимум**		15	15

Строится график зависимости степени от k. Результат вычислений для числа 23 представлен на рисунке 4.

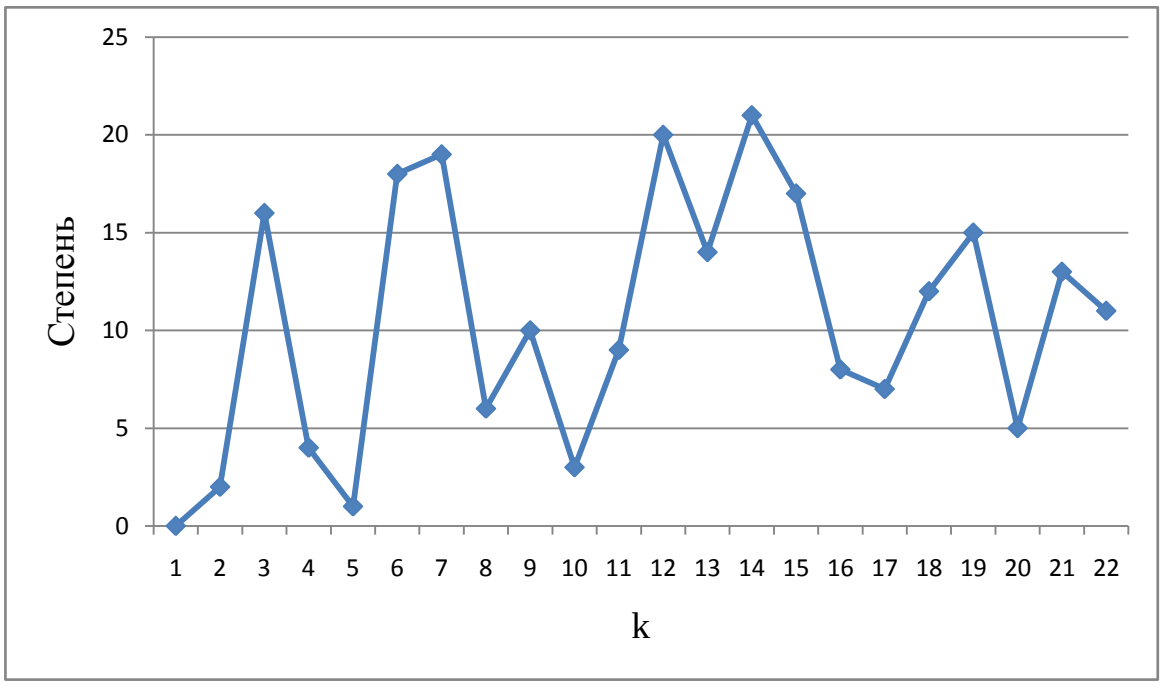

Рисунок 4 – Зависимость степени от k для $P = 23$

Строится график зависимости $d(k)$ от k. Результат вычислений для числа 23 представлен на рисунке 5.

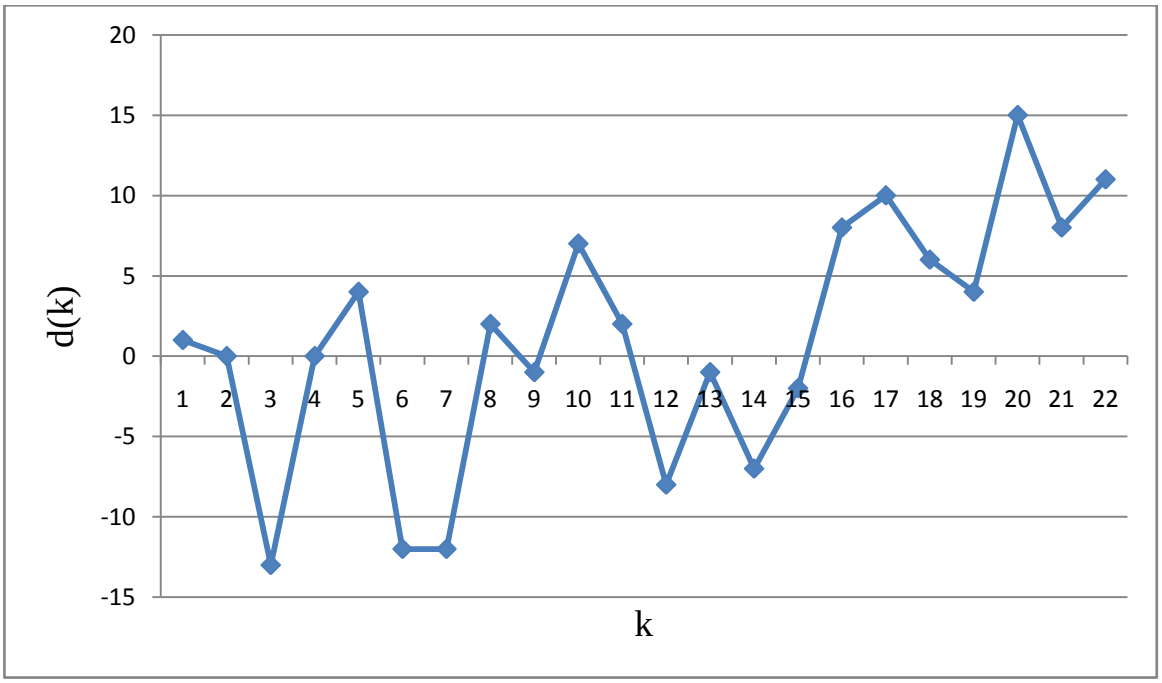

Рисунок 5 – Зависимость $d(k)$ от k для $P = 23$

Строится график зависимости $|d(k)|$ от k. Результат вычислений для числа 23 представлен на рисунке 6.

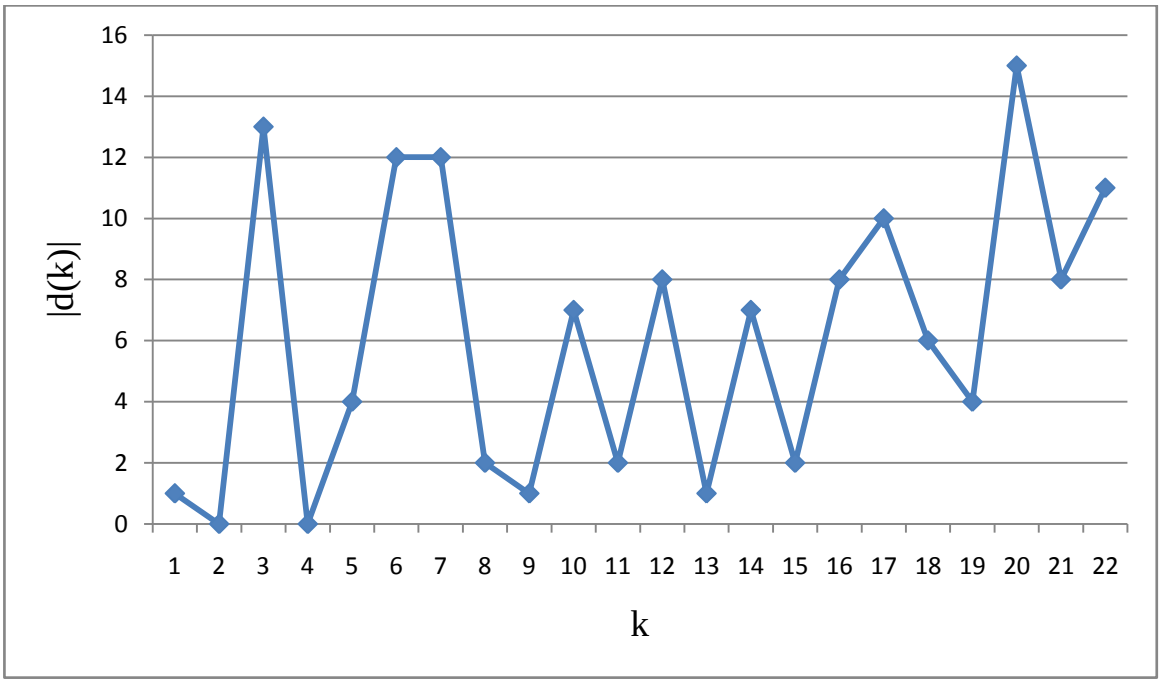

Рисунок 6 – Зависимость $|d(k)|$ от k для $P = 23$

Таким образом, для каждого числа можно составить таблицу со всеми интересующими значениями. Построить графики зависимостей этих значений от k.

Аналогичным образом осуществляются вычисления для другого числа, находятся результаты и строятся графиками. Результаты для $P = 79$ приведены в приложении А.

Вывод по третьей главе

В третьей главе подробно представлен процесс вычисления для числа 23, для него составлена таблица и построены графики. Подобные вычисления можно провести с любым простым числом.

4. РАЗРАБОТКА МОДЕЛИ, ПРОЕКТИРОВАНИЕ

Функциональные требования

Функциональные требования системы описывают функционал программного обеспечения и поведение, которое должна предоставлять данная программа. Для реализации приложения были выдвинуты следующие требования.

1. Пользователь должен иметь возможность вводить данные для вычислений.
2. Пользователь должен иметь возможность видеть результаты вычислений.
3. Система должна производить проверку корректности вводимых данных.
4. Пользователь должен иметь возможность просматривать список простых чисел.

Нефункциональные требования

Нефункциональные требования определяют свойства и ограничения, которые накладываются на систему, не относящиеся к ее поведению. Описывают, как должен работать программный продукт. Разрабатываемое приложение должно удовлетворять следующим нефункциональным требованиям.

1. Система должна корректно функционировать на компьютерах под управлением операционной системы семейства Windows версии не ниже Windows 7.
2. Система должна иметь удобный и понятный пользовательский интерфейс.
3. Система должна быть завершенной без нужды в установке каких-либо дополнений.

4.1. Диаграммы вариантов использования и деятельности

В ходе проектирования программной системы была построена модель взаимодействия пользователя с приложением в виде диаграммы вариантов использования, что помогает описать систему на концептуальном уровне.

Приложение используется одним актером: пользователем. Пользователь может совершить следующие действия.

1. Получить список простых чисел.
2. Получить ответ по заданному числу или ошибку о введении некорректных данных.
3. Выйти из приложения.

На рисунке 7 приведена диаграмма вариантов использования.

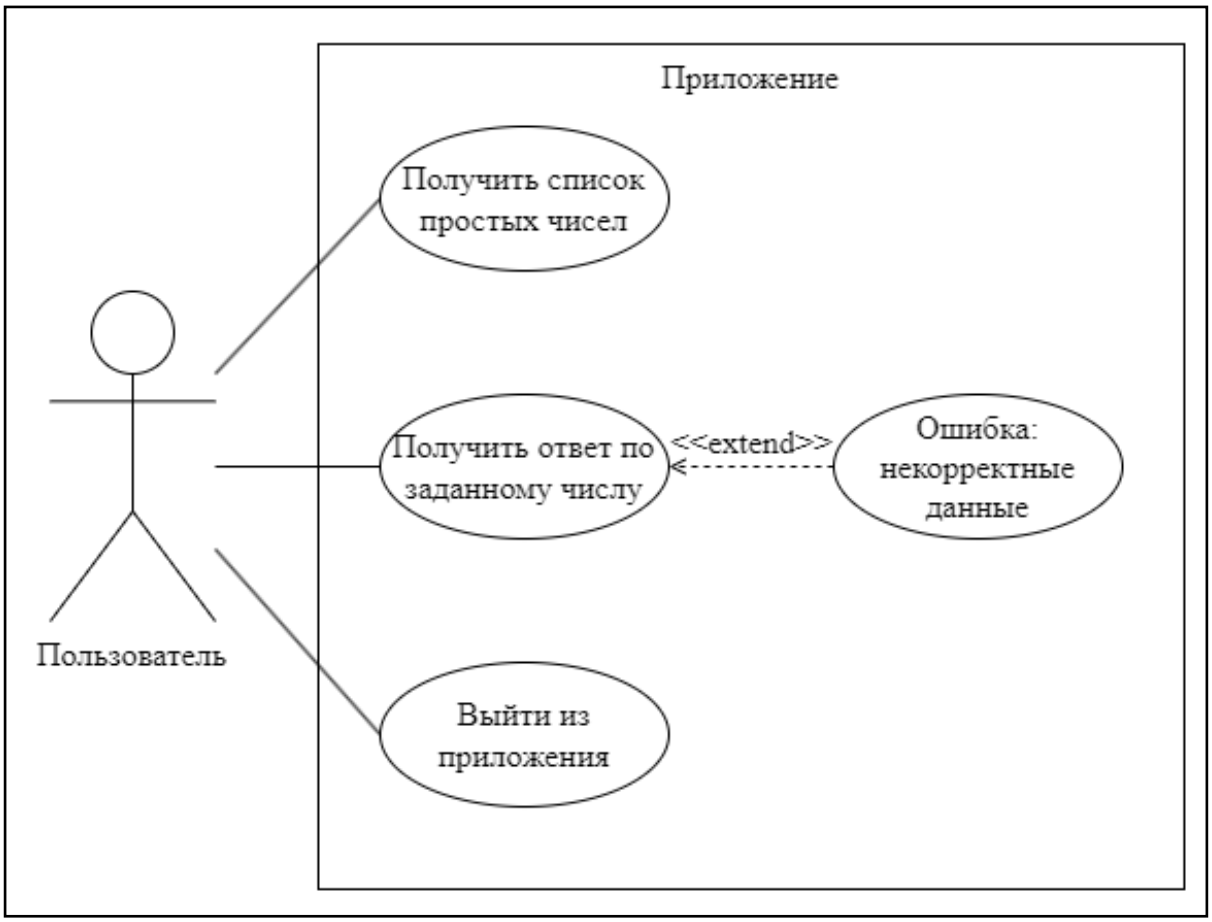

Рисунок 7 – Диаграмма вариантов использования

Для отображения взаимодействия пользователя с программной системой была разработана диаграмма деятельности для варианта использования «Получить ответ по заданному числу». Диаграмма деятельности (рисунок 8) описывает сценарий вычисления всех дискретных R-логарифмов по модулю.

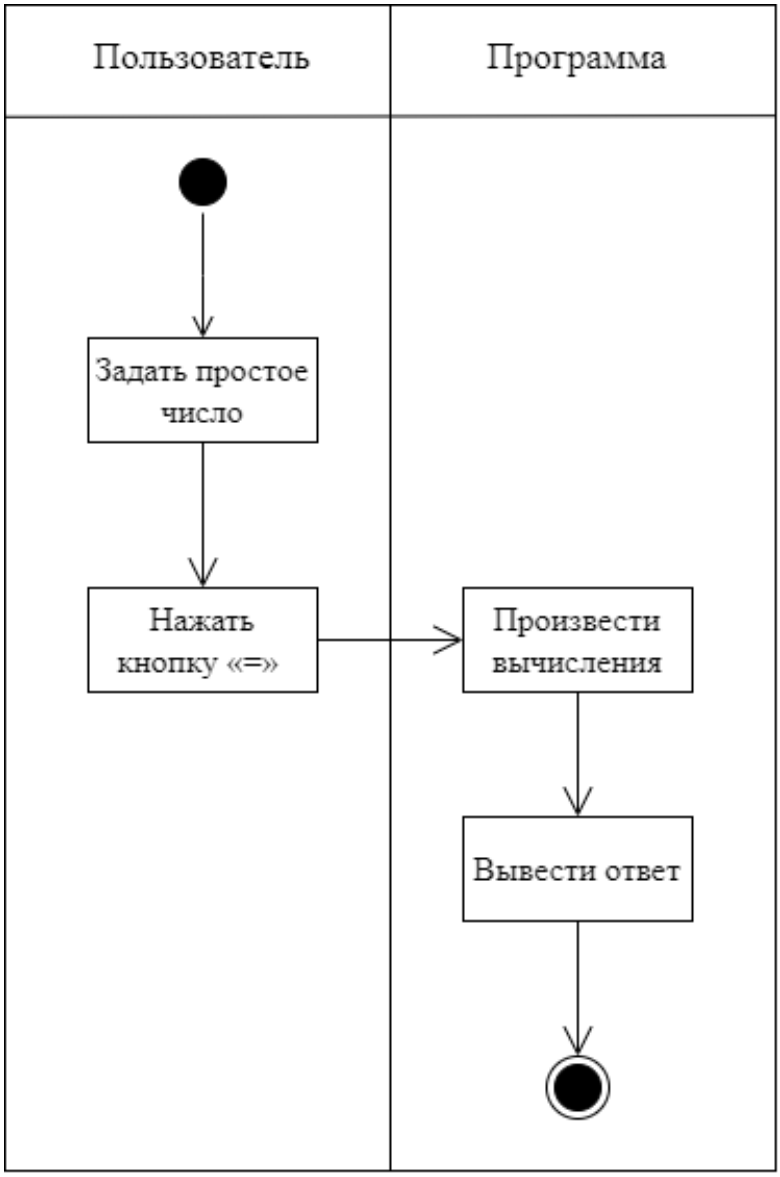

Рисунок 8 – Диаграмма деятельности

На данной диаграмме показан процесс получения результата вычислений всех дискретных R-логарифмов по модулю. Пользователю необходимо задать простое число для вычисления, запустить выполнение про-

граммы и дождаться ее завершения. В результате будет выведено соответствующее окно.

Диаграммы деятельности для вариантов использования «Получить список простых чисел» и «Выйти из приложения» имеют примитивную структуру, вследствие чего их построение не будет проведено. Для их использования необходимо нажать на соответствующую кнопку в выпадающем меню «Справка» главного окна приложения. В результате чего будет выведено окно со списком простых чисел или осуществлен выход из программы соответственно.

4.2. Проектирование интерфейса

Одним из важнейших этапов проектирования является разработка дизайна приложения. Для этого были созданы макеты экранов системы, обозначены все компоненты каждого окна приложения. Рассказано о предназначении всех компонентов программы.

Главное окно приложения имеет следующие компоненты.

1. Поле для ввода числа. Ввод осуществляется за счет набора цифр в приложении или при использовании клавиатуры.
2. Кнопки от 0 до 9, отвечающие за ввод соответствующих цифр в поле ввода.
3. Кнопка «←» для удаления последней цифры из поля ввода.
4. Кнопка «C» для полной очистки поля ввода.
5. Кнопка «=» для получения результата от числа, введенного в поле ввода.
6. Выпадающее меню «Справка», содержащее прочий функционал, описанный далее в тексте.

Модель главного окна приложения представлена на рисунке 9.

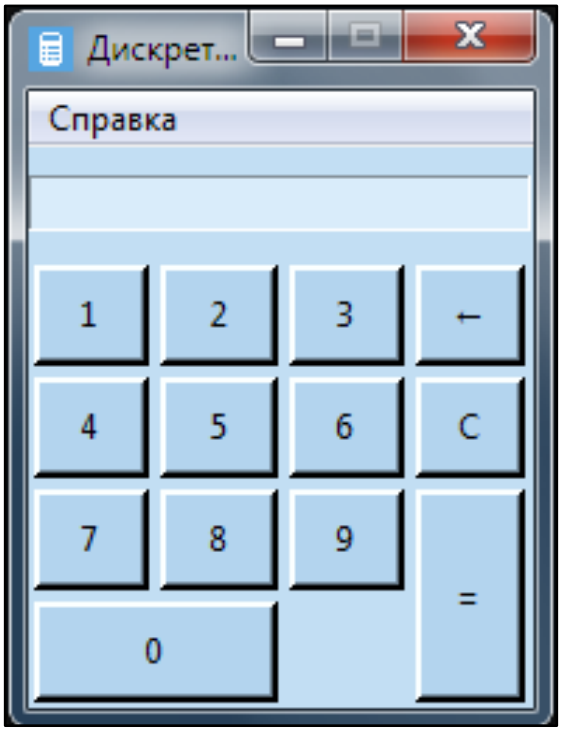

Рисунок 9 – Модель главного окна приложения

Выпадающее меню «Справка» имеет следующие компоненты.

1. Кнопка «Список простых чисел» для вывода списка простых чисел до 1000.
2. Кнопка «Выход» для закрытия приложения.

Компоненты выпадающего меню «Справка» представлены на рисунке 10.

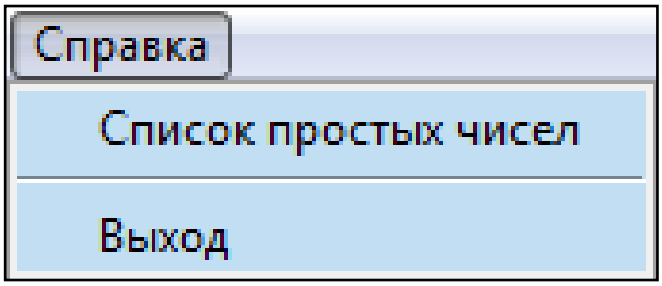

Рисунок 10 – Компоненты выпадающего меню «Справка»

Окно «Список простых чисел» представляет список всех простых чисел до 1000.

Модель окна с выводом списка простых чисел представлена на рисунке 11.

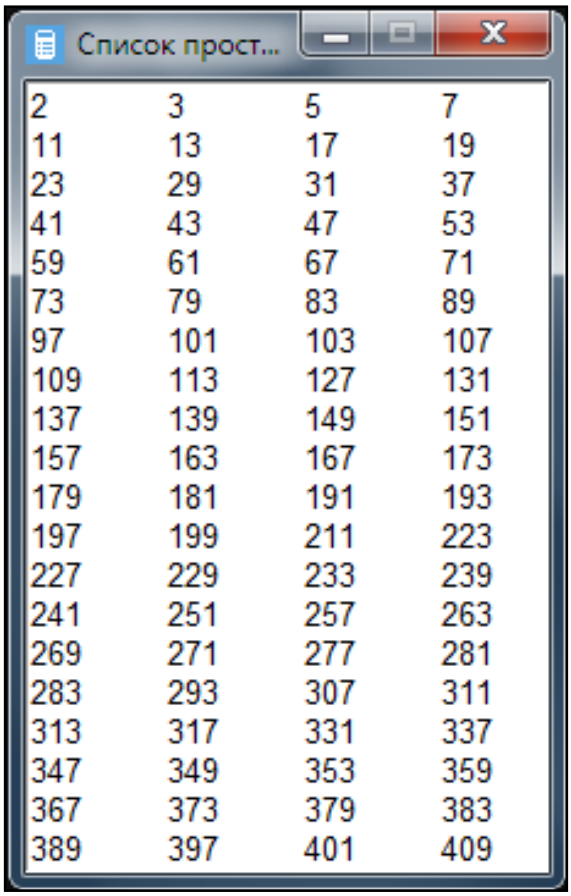

Рисунок 11 – Модель окна с выводом списка простых чисел

Окно с выводом результата имеет следующие компоненты.

1. Число p – введенное простое число.
2. Число r – наименьший примитивный корень по модулю p.
3. Столбец i – степень, в которую возводится число r.
4. Столбец ost – остаток от деления числа r^i по модулю p.
5. Столбец dk – разность ost и i.
6. Столбец dk_mod – число dk, взятое по модулю.
7. Числа dk_min, dk_max, dk_mod_min и dk_mod_max – минимальные и максимальные значения соответствующих столбцов.
8. Кнопка «Копировать» для копирования данных и помещения их в удобный формат.

Модель окна с выводом результата представлена на рисунке 12.

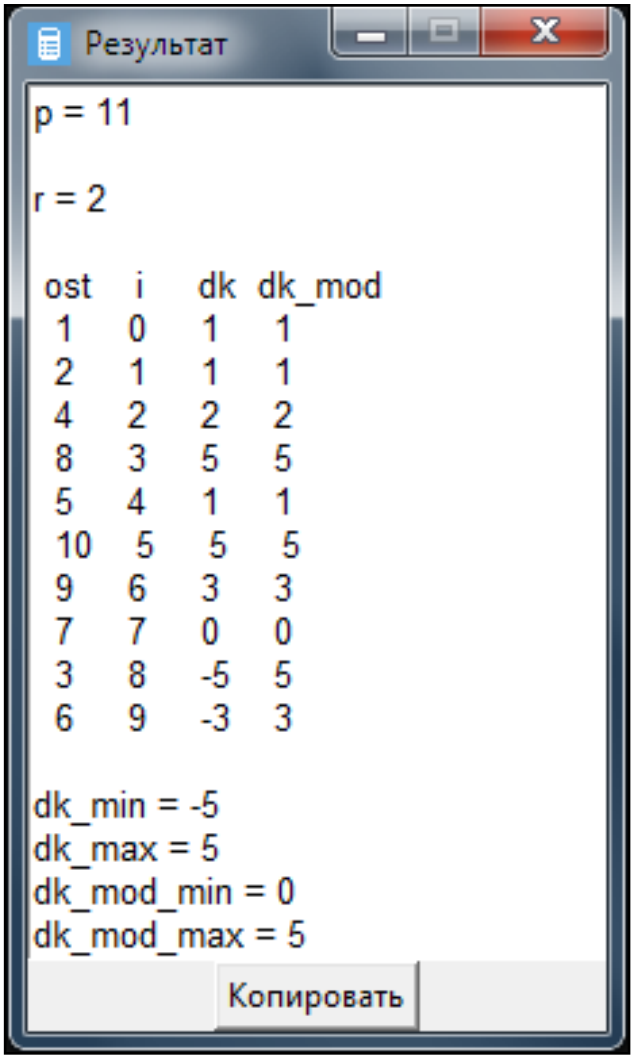

Рисунок 12 – Модель окна с выводом результата

Окно с выводом результата для чисел без примитивного корня имеет следующие компоненты.

1. Число p – введенное число.
2. Информация о том, что у введенного числа нет примитивного корня.

Модель окна с выводом результата для чисел без примитивного корня, представлена на рисунке 13.

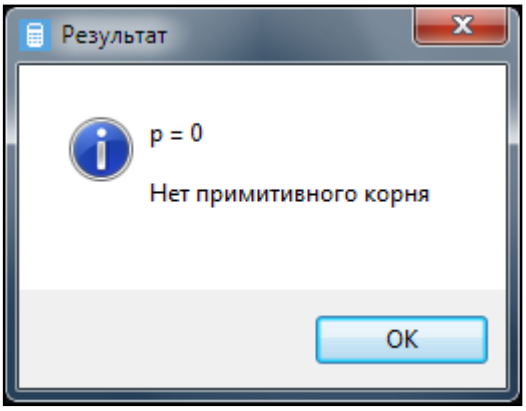

Рисунок 13 – Модель окна с выводом результата для чисел без примитивного корня

Окно с выводом ошибки для случаев, когда было введено не числовое значение, содержит информацию о том, что необходимо ввести число.

Модель окна с выводом ошибки для случаев, когда было введено не числовое значение, представлена на рисунке 14.

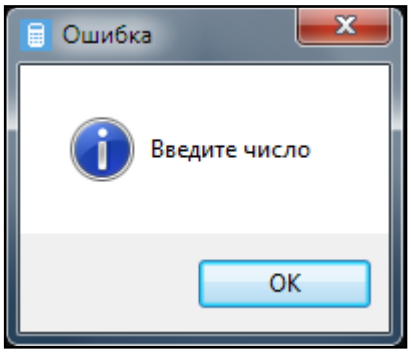

Рисунок 14 – Модель окна с выводом ошибки для случаев, когда было введено не числовое значение

Вывод по четвертой главе

В четвертой главе были определены функциональные и нефункциональные требования, приведены диаграммы вариантов использования и деятельности. Спроектированы модели всех окон, задействованных при использовании приложения. Указаны предназначения всех компонентов для каждого окна.

5. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ

Выбранные технологии и инструменты

При реализации системы были выбраны следующие технологии и инструменты.

1. Язык программирования Python. Выбор языка обусловлен его простотой, легкостью освоения, а также широкими возможностями для быстрого прототипирования и разработки. Python поддерживает большое количество библиотек и инструментов, что делает его универсальным решением для различных типов приложений, включая настольные. Используемая версия: Python 3.12.2.

2. Среда разработки PyCharm. Мощная интегрированная среда разработки, предоставляющая все необходимые инструменты для разработки, отладки и тестирования приложений на Python. PyCharm поддерживает автодополнение кода, рефакторинг, работу с системами контроля версий и многие другие функции, облегчающие процесс разработки. Используемая версия: PyCharm Community Edition 2023.2.6.

3. Интеграция пользовательского интерфейса Tkinter. Встроенная библиотека для создания графических интерфейсов в Python. Tkinter позволяет создавать удобные и интуитивно понятные пользовательские интерфейсы с использованием стандартных элементов управления, таких как кнопки, метки, текстовые поля и другие. Tkinter легко изучить и использовать, что делает его популярным выбором для разработки настольных приложений на Python. Используемая версия: Tkinter 8.6.13.

Применение вышеуказанных технологий и инструментов обеспечивает создание надежной и функциональной системы с интуитивно понятным интерфейсом, позволяя эффективно реализовать все поставленные цели и задачи.

Разработка оконного приложения и его компонентов

Создание главного окна приложения и поля для ввода представлено в листинге 3.

Листинг 3 – Создание главного окна приложения и поля для ввода

```
win = tk.Tk()
win.geometry(f"180x200")
win.resizable(False, False)
win['bg'] = '#C2DEF3'
win.iconbitmap(default="calc.ico")
win.title("Дискретные логарифмы")
entry=Entry(win, justify=RIGHT, font=('Arial', 10), width=25, bg='#D9ECFA')
entry.grid(row=0, column=0, columnspan=4, sticky=W)
```

Реализация работы кнопок от 0 до 9 для ввода соответствующих цифр в поле ввода представлена в листинге 4.

Листинг 4 – Реализация работы кнопок от 0 до 9

```
tk.Button(text='1', bd=3, bg='#B3D4EE', command=lambda : add_digit(1)).grid
(row=1, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='2', bd=3, bg='#B3D4EE', command=lambda : add_digit(2)).grid
(row=1, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='3', bd=3, bg='#B3D4EE', command=lambda : add_digit(3)).grid
(row=1, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='4', bd=3, bg='#B3D4EE', command=lambda : add_digit(4)).grid
(row=2, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='5', bd=3, bg='#B3D4EE', command=lambda : add_digit(5)).grid
(row=2, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='6', bd=3, bg='#B3D4EE', command=lambda : add_digit(6)).grid
(row=2, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='7', bd=3, bg='#B3D4EE', command=lambda : add_digit(7)).grid
(row=3, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='8', bd=3, bg='#B3D4EE', command=lambda : add_digit(8)).grid
(row=3, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='9', bd=3, bg='#B3D4EE', command=lambda : add_digit(9)).grid
(row=3, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='0', bd=3, bg='#B3D4EE', command=lambda : add_digit(0)).grid
(row=4, column=0, columnspan=2, stick='wens', padx=2, pady=2)
```

Реализация работы кнопки «←» для удаления последней цифры из поля ввода представлена в листинге 5.

Листинг 5 – Реализация работы кнопки «←»

```
def del_digit():
    value = entry.get()[:-1]
    entry.delete(0, END)
    entry.insert(0, value)
tk.Button(text='←', bd=3, bg='#B3D4EE', command=lambda : del_digit()).grid(
row=1, column=3, stick='wens', padx=2, pady=2)
```

Реализация работы кнопки «C» для полной очистки поля ввода представлена в листинге 6.

Листинг 6 – Реализация работы кнопки «C»

```
def clear():
    entry.delete(0, END)
    entry.insert(0, '')
tk.Button(text='C', bd=3, bg='#B3D4EE', command=lambda : clear()).grid(row=
2, column=3, stick='wens', padx=2, pady=2)
```

Реализация работы кнопки «=» для получения результата представлена в листинге 7.

Листинг 7 – Реализация работы кнопки «=»

```
def dlog():

    try:
        p = int(entry.get())
    except ValueError:
        result_text = "Введите число"
        messagebox.showinfo(title='Ошибка', message=result_text)
        exit()

    exam = list(range(1, p))

    result_text = f"p = {p}\n\n"

    # Нахождение наименьшего примитивного корня r
    def min_r(p, exam):…

    r = min_r(p, exam)

    try:
        result_text += f"r = {int(r)}\n\n"
    except TypeError:
        result_text += "Нет примитивного корня\n\n"
        messagebox.showinfo(title='Результат', message=result_text)
        exit()

    # Нахождение всех дискретных r-логарифмов
    def log(p, r):…

    log = log(p, r)
    result_text += f"{log}\n"

    result_window = tk.Toplevel()
    result_window.resizable(False, False)
    result_window.title('Результат')

    text_area = tk.Text(result_window, height=20, width=30)
    text_area.insert(tk.END, result_text)
    text_area.pack()

tk.Button(text='=', bd=3, bg='#B3D4EE', command=dlog).grid(row=3, column=3,
 rowspan=2, stick='wens', padx=2, pady=2)
```

Создание выпадающего меню «Справка» и реализация кнопок «Список простых чисел» и «Выход» представлено в листинге 8.

Листинг 8 – Создание выпадающего меню «Справка» и реализация кнопок «Список простых чисел» и «Выход»

```
def prime_list():
    prime_list_window = tk.Toplevel()
    prime_list_window.resizable(False, False)
    prime_list_window.title('Список простых чисел')

    text_area = tk.Text(prime_list_window, height=20, width=30, font=('Aria
l', 10))
    text_area.insert(tk.END, prime_list_text)
    text_area.pack()

mainmenu = Menu(win)
win.config(menu=mainmenu)
filemenu = Menu(mainmenu, tearoff=0, bg='#C2DEF3')
filemenu.add_command(label="Список простых чисел", command=prime_list)
filemenu.add_separator()
filemenu.add_command(label="Выход", command=exit)
mainmenu.add_cascade(label="Справка", menu=filemenu)
```

Реализация работы кнопки «Копировать» для копирования данных и помещения их в удобный формат представлена в листинге 9.

Листинг 9 – Реализация кнопки «Копировать»

```
def prime_list():
    prime_list_window = tk.Toplevel()
    prime_list_window.resizable(False, False)
    prime_list_window.title('Список простых чисел')

    text_area = tk.Text(prime_list_window, height=20, width=30, font=('Aria
l', 10))
    text_area.insert(tk.END, prime_list_text)
    text_area.pack()

mainmenu = Menu(win)
win.config(menu=mainmenu)
filemenu = Menu(mainmenu, tearoff=0, bg='#C2DEF3')
filemenu.add_command(label="Список простых чисел", command=prime_list)
filemenu.add_separator()
filemenu.add_command(label="Выход", command=exit)
mainmenu.add_cascade(label="Справка", menu=filemenu)
```

Вывод по пятой главе

В пятой главе «Программная реализация системы» были выбраны технологии и инструменты для разработки системы. Разработана реализация оконного приложения и всех его компонентов, упомянутых в предыдущей главе.

6. ТЕСТИРОВАНИЕ СИСТЕМЫ

Было проведено функциональное тестирования для проверки соответствия требованиям программы.

Были проведены тесты на корректность входных и выходных данных. В таблице 2 приведены результаты данного тестирования.

Таблица 2 – Тестирование на корректность входных и выходных данных

№	Работа программы	Ожидаемый результат	Тест
1	Ввод: 7 Вывод: r = 3 dk_min = 0 dk_max = 3 dk_mod_min = 0 dk_mod_max = 3	Ввод: 7 Вывод: r = 3 dk_min = 0 dk_max = 3 dk_mod_min = 0 dk_mod_max = 3	Пройден
2	Ввод: 23 Вывод: r = 5 dk_min = -13 dk_max = 15 dk_mod_min = 0 dk_mod_max = 15	Ввод: 23 Вывод: r = 5 dk_min = -13 dk_max = 15 dk_mod_min = 0 dk_mod_max = 15	Пройден
3	Ввод: 50 Вывод: Нет примитивного корня	Ввод: 50 Вывод: Нет примитивного корня	Пройден
4	Ввод: -23 Вывод: Нет примитивного корня	Ввод: -23 Вывод: Нет примитивного корня	Пройден
5	Ввод: +23 Вывод: r = 5 dk_min = -13 dk_max = 15 dk_mod_min = 0 dk_mod_max = 15	Ввод: +23 Вывод: r = 5 dk_min = -13 dk_max = 15 dk_mod_min = 0 dk_mod_max = 15	Пройден
6	Ввод: тест Вывод: Введите число	Ввод: тест Вывод: Введите число	Пройден

Замерено время выполнения вычислений для разных значений p. Результаты тестирования производительности приложения представлены на рисунке 15.

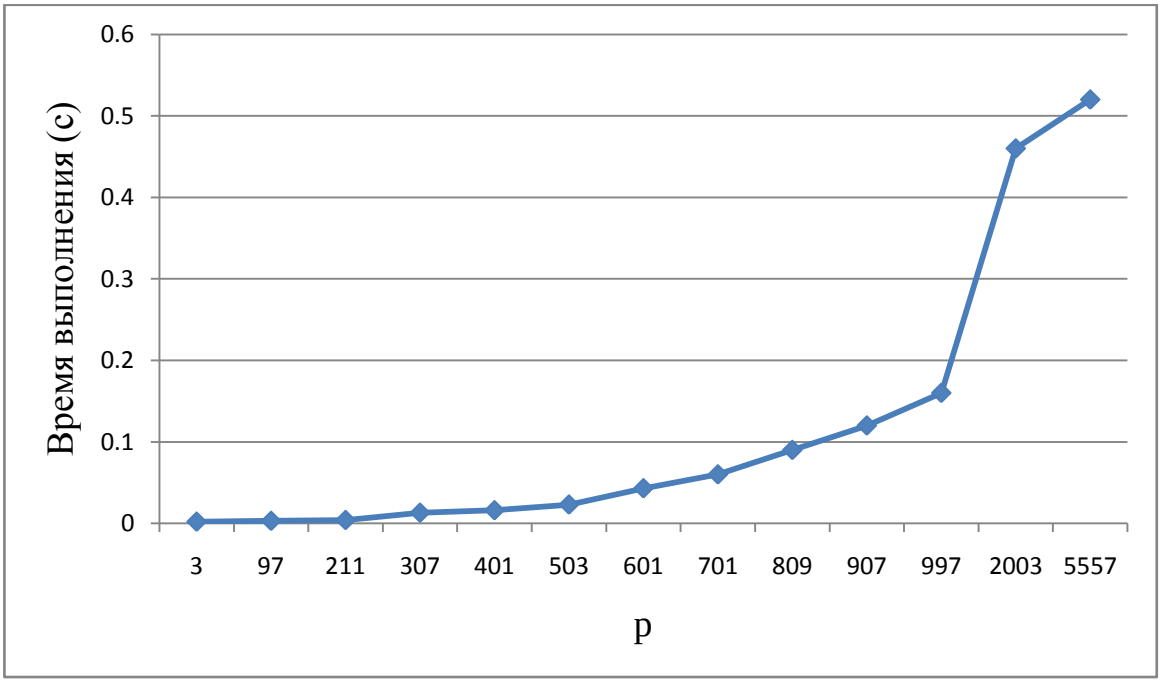

Рисунок 15 – Результаты тестирования производительности приложения

Исходный код программы, осуществляющей поиск дискретных логарифмов, приведен в приложении Б.

Вывод по шестой главе

В шестой главе «Тестирование системы» были проведены функциональное тестирование, тестирование приложения на корректность входных и выходных данных, а также проанализирована производительность системы.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы была разработана программная система для вычисления и анализа дискретных логарифмов для начальных диапазонов простых чисел. Были выполнены следующие задачи.

1. Составлен список простых чисел из диапазона [2..P].
2. Разработана функция для нахождения наименьшего примитивного корня R по модулю простых чисел из диапазона [2..P].
3. Разработана функция для нахождения и анализа всех дискретных R-логарифмов по модулю простых чисел из диапазона [2..P].
4. Разработано графическое приложение со всеми указанными функциями.

Реализованная программная система успешно прошла все тесты, полностью соответствует определенным функциональным и нефункциональным требованиям.

Планируется дальнейшая работа по добавлению функциональных возможностей, таких как построение графиков на основе введенных данных. Вследствие чего можно добавить процесс отслеживания корреляции между параметрами одного простого числа или нескольких разных простых чисел. Это поможет упростить восприятие получаемых данных, а также установить связь между различными параметрами, если таковая имеется.

ЛИТЕРАТУРА

1. The GAP Group, GAP. // Groups, Algorithms and Programming Version 4.12.2, 2022. [Электронный ресурс] URL: https://www.gap-system.org/Doc/manuals (дата обращения: 21.02.2024 г.).

2. Агафонов В.Н. Математические основы обработки информации. // Новосибирск: НГУ, 1982. – 92 с.

3. Верещагин Н.К., Шень А. Начала теории множеств. // М.: МЦНМО, 2000. – 128 с.

4. Виленкин Н.Я. Популярная комбинаторика. // М.: Наука, 1975. – 208 с.

5. Виноградов И.М. Основы теории чисел. // М.: – Л., Гостехиздат, 1952. – 180 с.

6. Гиндикин С.Г. Алгебра и логика в задачах. // М.: Наука, 1972. – 288 с.

7. Грицук Д.В. Компьютерная алгебра. // Брестский государственный университет А.С. Пушкина. Брест, 2018. – 270 с.

8. Ерусалимский Я.М. Дискретная математика: Теория, задачи, приложения. // М.: Вузовская книга, 2011. – 265 с.

9. Карпов Ю.Г. Теория автоматов. // М.: и др.: Питер, 2003. – 206 с.

10. Липский В. Комбинаторика для программистов. // М.: Мир, 1988. – 213 с.

11. Нестеренко Ю.В. Теория чисел. // М.: Академия, 2008. – 292 с.

12. Новиков Ф.А. Дискретная математика: для бакалавров и магистров. // М.: [и др.]: Питер, 2013. – 399 с.

13. Тайцлин М.А., Столбоушкин А.П. Математические основания информатики. // Тверь: ТвГУ, 1998. – 377 с.

14. Трост Э.В. Простые числа. // Москва, 1959. – 135 с.

15. Яблонский С.В. Введение в дискретную математику. // М.: Высшая школа, 2003. – 384 с.

ПРИЛОЖЕНИЯ

Приложение А. Результаты для P = 79

Результаты для $P = 79$ приведены в таблице 1.

Таблица 1 – Результаты для $P = 79$

По возрастанию k				По возрастанию степени			
k	**Степень**	**d(k)**	**|d(k)|**	**Степень**	**k**	**d(k)**	**|d(k)|**
1	0	1	1	0	1	1	1
2	4	-2	2	1	3	2	2
3	1	2	2	2	9	7	7
4	8	-4	4	3	27	24	24
5	62	-57	57	4	2	-2	2
6	5	1	1	5	6	1	1
7	53	-46	46	6	18	12	12
8	12	-4	4	7	54	47	47
9	2	7	7	8	4	-4	4
10	66	-56	56	9	12	3	3
11	68	-57	57	10	36	26	26
12	9	3	3	11	29	18	18
13	34	-21	21	12	8	-4	4
14	57	-43	43	13	24	11	11
15	63	-48	48	14	72	58	58
16	16	0	0	15	58	43	43
17	21	-4	4	16	16	0	0
18	6	12	12	17	48	31	31
19	32	-13	13	18	65	47	47
20	70	-50	50	19	37	18	18
21	54	-33	33	20	32	12	12
22	72	-50	50	21	17	-4	4
23	26	-3	3	22	51	29	29
24	13	11	11	23	74	51	51
25	46	-21	21	24	64	40	40
26	38	-12	12	25	34	9	9
27	3	24	24	26	23	-3	3
28	61	-33	33	27	69	42	42
29	11	18	18	28	49	21	21
30	67	-37	37	29	68	39	39
31	56	-25	25	30	46	16	16
32	20	12	12	31	59	28	28
33	69	-36	36	32	19	-13	13
34	25	9	9	33	57	24	24
35	37	-2	2	34	13	-21	21
36	10	26	26	35	39	4	4
37	19	18	18	36	38	2	2
38	36	2	2	37	35	-2	2
39	35	4	4	38	26	-12	12
40	74	-34	34	39	78	39	39

Окончание таблицы 1 приложения А

По возрастанию k				По возрастанию степени			
k	**Степень**	**d(k)**	**|d(k)|**	**k**	**Степень**	**d(k)**	**|d(k)|**
41	75	-34	34	40	76	36	36
42	58	-16	16	41	70	29	29
43	49	-6	6	42	52	10	10
44	76	-32	32	43	77	34	34
45	64	-19	19	44	73	29	29
46	30	16	16	45	61	16	16
47	59	-12	12	46	25	-21	21
48	17	31	31	47	75	28	28
49	28	21	21	48	67	19	19
50	50	0	0	49	43	-6	6
51	22	29	29	50	50	0	0
52	42	10	10	51	71	20	20
53	77	-24	24	52	55	3	3
54	7	47	47	53	7	-46	46
55	52	3	3	54	21	-33	33
56	65	-9	9	55	63	8	8
57	33	24	24	56	31	-25	25
58	15	43	43	57	14	-43	43
59	31	28	28	58	42	-16	16
60	71	-11	11	59	47	-12	12
61	45	16	16	60	62	2	2
62	60	2	2	61	28	-33	33
63	55	8	8	62	5	-57	57
64	24	40	40	63	15	-48	48
65	18	47	47	64	45	-19	19
66	73	-7	7	65	56	-9	9
67	48	19	19	66	10	-56	56
68	29	39	39	67	30	-37	37
69	27	42	42	68	11	-57	57
70	41	29	29	69	33	-36	36
71	51	20	20	70	20	-50	50
72	14	58	58	71	60	-11	11
73	44	29	29	72	22	-50	50
74	23	51	51	73	66	-7	7
75	47	28	28	74	40	-34	34
76	40	36	36	75	41	-34	34
77	43	34	34	76	44	-32	32
78	39	39	39	77	53	-24	24
Минимум		-57	0	**Минимум**		-57	0
Максимум		58	58	**Максимум**		58	58

График зависимости степени числа от k для $P = 79$ представлен на рисунке 1.

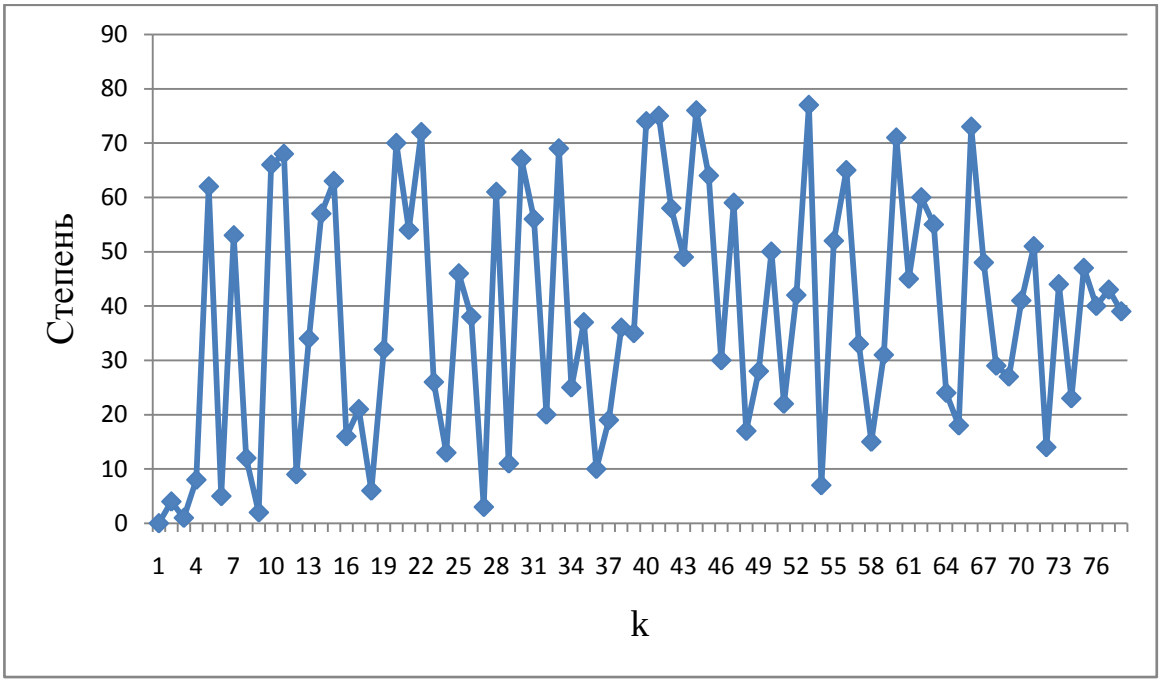

Рисунок 1 – Зависимость степени от k для $P = 79$

График зависимости $d(k)$ от k для $P = 79$ представлен на рисунке 2.

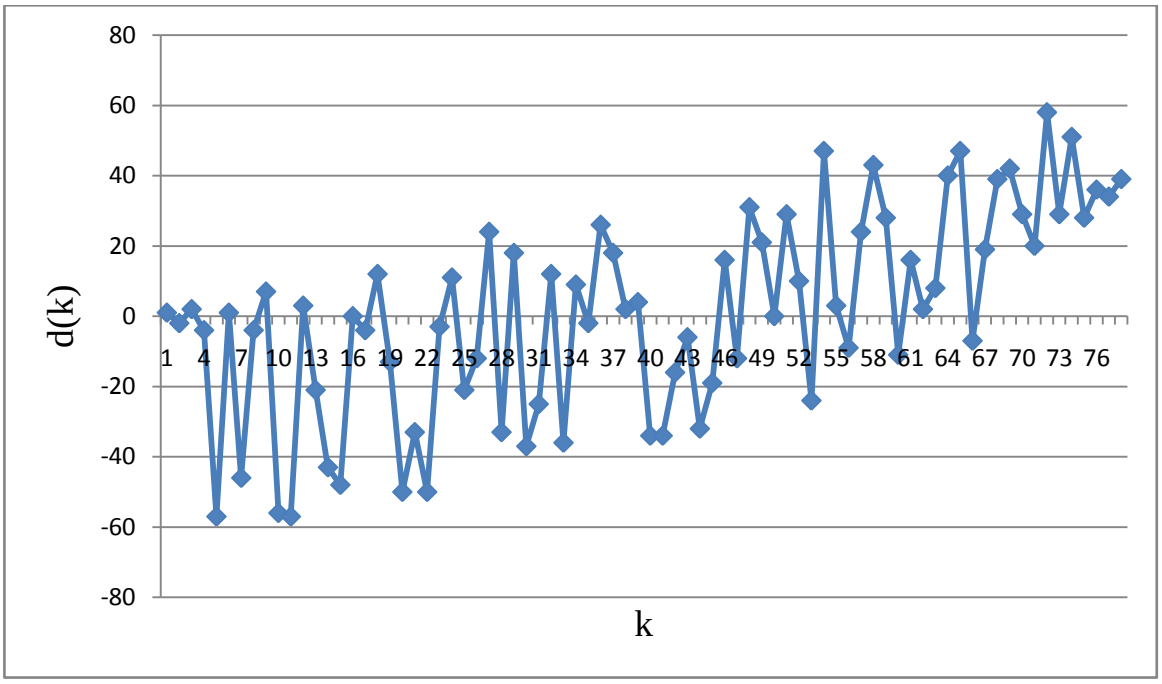

Рисунок 2 – Зависимость $d(k)$ от k для $P = 79$

График зависимости $|d(k)|$ от k для $P = 79$ представлен на рисунке 3.

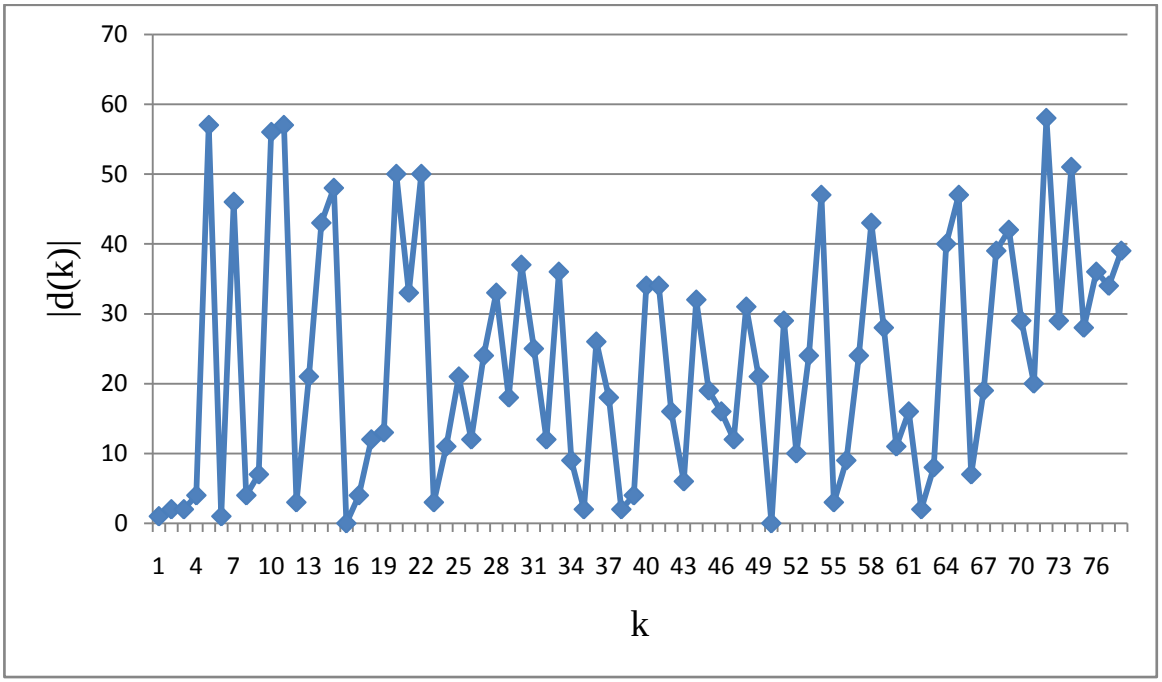

Рисунок 3 – Зависимость $|d(k)|$ от k для $P = 79$

Приложение Б. Исходный код программы

Исходный код программы, осуществляющей поиск дискретных логарифмов, представлен в листинге 1.

Листинг 1 – Исходный код программы для дискретных логарифмов

```
from tkinter import *
from tkinter import messagebox
import tkinter as tk
from tkinter import ttk

def dlog():

    try:
        p = int(entry.get())
    except ValueError:
        result_text = "Введите число"
        messagebox.showinfo(title='Ошибка', message=result_text)
        return

    exam = list(range(1, p))

    result_text = f"p = {p}\n\n"

    # Нахождение наименьшего примитивного корня r
    def min_r(p, exam):
        for i in range(2, p, +1):
            new_exam = exam.copy()
            for j in range(1, p+1, +1):
                ost = i**j % p
                try:
                    new_exam.remove(ost)
                except ValueError:
                    pass
            if new_exam == []:
                r = i
                return r

    r = min_r(p, exam)

    try:
        result_text += f"r = {int(r)}\n\n"
    except TypeError:
        result_text += "Нет примитивного корня\n\n"
        messagebox.showinfo(title='Результат', message=result_text)
        return

    # Нахождение всех дискретных r-логарифмов
    def log(p, r):
        dk_min = 1000
        dk_max = -1000
        dk_mod_min = 1000
        dk_mod_max = -1000
        log_result_text = f"{'ost'.center(5)} {'i'.center(5)} {'dk'.center(
5)} {'dk_mod'.center(5)}\n"

        for i in range(0, p-1, +1):
            ost = r**i % p

            dk = ost - i
```

```
            if dk < dk_min:
                dk_min = dk
            elif dk > dk_max:
                dk_max = dk

            dk_mod = abs(ost - i)
            if dk_mod < dk_mod_min:
                dk_mod_min = dk_mod
            elif dk_mod > dk_mod_max:
                dk_mod_max = dk_mod

            log_result_text += f"{str(ost).center(5)} {str(i).center(5)} {s
tr(dk).center(5)} {str(dk_mod).center(5)}\n"

        log_result_text += f"\ndk_min = {dk_min}\ndk_max = {dk_max}\ndk_mod
_min = {dk_mod_min}\ndk_mod_max = {dk_mod_max}"
        return log_result_text

    log = log(p, r)
    result_text += f"{log}"

    result_window = tk.Toplevel()
    result_window.resizable(False, False)
    result_window.title('Результат')

    text_area = tk.Text(result_window, height=20, width=30, font=('Arial',
10))
    text_area.insert(tk.END, result_text)
    text_area.pack()

    def copy_text():
        result_window.clipboard_clear()
        result_window.clipboard_append(text_area.get("1.0", tk.END))

    copy_button = tk.Button(result_window, text="Копировать", command=copy_
text)
    copy_button.pack()


win = tk.Tk()
win.geometry(f"180x200")
win.resizable(False, False)
win['bg'] = '#C2DEF3'
win.iconbitmap(default="calc.ico")
win.title("Дискретные логарифмы")

prime_list_text = '''2  3   5   7   11  13  17  19  23  29  31  37
41  43  47  53  59  61  67  71  73  79  83  89
97  101 103 107 109 113 127 131 137 139 149 151
157 163 167 173 179 181 191 193 197 199 211 223
227 229 233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349 353 359
367 373 379 383 389 397 401 409 419 421 431 433
439 443 449 457 461 463 467 479 487 491 499 503
509 521 523 541 547 557 563 569 571 577 587 593
599 601 607 613 617 619 631 641 643 647 653 659
661 673 677 683 691 701 709 719 727 733 739 743
751 757 761 769 773 787 797 809 811 821 823 827
829 839 853 857 859 863 877 881 883 887 907 911
919 929 937 941 947 953 967 971 977 983 991 997'''
```

```
def prime_list():
    prime_list_window = tk.Toplevel()
    prime_list_window.resizable(False, False)
    prime_list_window.title('Список простых чисел')

    text_area = tk.Text(prime_list_window, height=20, width=30, font=('Aria
l', 10))
    text_area.insert(tk.END, prime_list_text)
    text_area.pack()


mainmenu = Menu(win)
win.config(menu=mainmenu)
filemenu = Menu(mainmenu, tearoff=0, bg='#C2DEF3')
filemenu.add_command(label="Список простых чисел", command=prime_list)
filemenu.add_separator()
filemenu.add_command(label="Выход", command=exit)
mainmenu.add_cascade(label="Справка", menu=filemenu)

en-
try = Entry(win, justify=RIGHT, font=('Arial', 10), width=25, bg='#D9ECFA')
entry.grid(row=0, column=0, columnspan=4, sticky=W)

def add_digit(digit):
    value = entry.get() + str(digit)
    entry.delete(0, END)
    entry.insert(0, value)

def del_digit():
    value = entry.get()[:-1]
    entry.delete(0, END)
    entry.insert(0, value)

def clear():
    entry.delete(0, END)
    entry.insert(0, '')

tk.Button(text='1', bd=3, bg='#B3D4EE', command=lambda : add_digit(1)).grid
(row=1, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='2', bd=3, bg='#B3D4EE', command=lambda : add_digit(2)).grid
(row=1, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='3', bd=3, bg='#B3D4EE', command=lambda : add_digit(3)).grid
(row=1, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='4', bd=3, bg='#B3D4EE', command=lambda : add_digit(4)).grid
(row=2, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='5', bd=3, bg='#B3D4EE', command=lambda : add_digit(5)).grid
(row=2, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='6', bd=3, bg='#B3D4EE', command=lambda : add_digit(6)).grid
(row=2, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='7', bd=3, bg='#B3D4EE', command=lambda : add_digit(7)).grid
(row=3, column=0, stick='wens', padx=2, pady=2)
tk.Button(text='8', bd=3, bg='#B3D4EE', command=lambda : add_digit(8)).grid
(row=3, column=1, stick='wens', padx=2, pady=2)
tk.Button(text='9', bd=3, bg='#B3D4EE', command=lambda : add_digit(9)).grid
(row=3, column=2, stick='wens', padx=2, pady=2)
tk.Button(text='0', bd=3, bg='#B3D4EE', command=lambda : add_digit(0)).grid
(row=4, column=0, columnspan=2, stick='wens', padx=2, pady=2)
tk.Button(text='←', bd=3, bg='#B3D4EE', command=lambda : del_digit()).grid(
row=1, column=3, stick='wens', padx=2, pady=2)
tk.Button(text='C', bd=3, bg='#B3D4EE', command=lambda : clear()).grid(row=
2, column=3, stick='wens', padx=2, pady=2)
```

```
tk.Button(text='=', bd=3, bg='#B3D4EE', command=dlog).grid(row=3, column=3,
 rowspan=2, stick='wens', padx=2, pady=2)

win.grid_columnconfigure(0, minsize=30)
win.grid_columnconfigure(1, minsize=30)
win.grid_columnconfigure(2, minsize=30)

win.grid_rowconfigure(0, minsize=40)
win.grid_rowconfigure(1, minsize=40)
win.grid_rowconfigure(2, minsize=40)
win.grid_rowconfigure(3, minsize=40)
win.grid_rowconfigure(4, minsize=40)
win.grid_rowconfigure(5, minsize=40)

win.mainloop()
```