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

РАБОТА ПРОВЕРЕНА

Рецензент
Доцент кафедры МиМОМ
ФГБАОУ ВО «ЮУрГГПУ»,
к.ф.-м.н.

___________ Е.О. Шумакова

«___»______________ 2024 г.

ДОПУСТИТЬ К ЗАЩИТЕ

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

__________ Л.Б. Соколинский

«___»___________ 2024 г.

Разработка программной системы для вычисления и анализа чисел, связанных с разбиениями

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.04.02.2024.308-1387.ВКР

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

Автор работы,
студент группы КЭ-220
__________ К.В. Бастрыкина

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 магистранта
студентке группы КЭ-220
Бастрыкиной Ксении Валерьевне,
обучающейся по направлению
02.04.02 «Фундаментальная информатика и информационные технологии»
(магистерская программа «Машинное обучение и анализ больших данных»)

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)

Разработка программной системы для анализа чисел, связанных с разбиениями.

2. **Срок сдачи студентом законченной работы:** 20.05.2024 г.

3. **Исходные данные к работе**

3.1. Эндрюс Г. Теория разбиений. – М.: Наука, 1982. – 256 с.

3.2. Фробениус Г. Теория характеров и представлений групп. – Харьков: Гос. науч.-техн. изд. Украины, 1937. – 211 с.

3.3. Flajolet P., Sedgewick R. Analytic Combinatorics. – Cambridge university press, 2009. – 810 с.

4. **Перечень подлежащих разработке вопросов**

4.1. Разработать алгоритм для вычисления вектора разбиения.

4.2. Разработать алгоритм для вычисления характеристики вектора разбиения.

4.3. Разработать алгоритм для вычисления ранга разбиения.

4.4. Разработать алгоритм для нахождения частного вектора разбиения.

4.5. Найти самоассоциированные разбиения.

4.6. Проанализировать, полученные значения.

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 К.В. Бастрыкина

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Изучение разбиений позволяет решать множество задач, связанных с перестановками, сочетаниями и другими комбинаторными задачами.

Теория разбиения актуальна в следующих направлениях.

1. Криптография. Например, в криптосистемах на основе решетки используются разбиения чисел для построения некоторых криптографических протоколов. Исследования в этой области направлены на разработку новых алгоритмов и методов, которые были бы устойчивы к квантовым вычислениям и улучшали бы общую безопасность криптосистем.
2. Комбинаторика. Исследования в этой области позволяют развивать новые методы и алгоритмы для решения комбинаторных задач, а также находить применение в других областях, таких как сетевой анализ или оптимизация.
3. Машинное обучение. Разбиение числа могут быть полезны в задачах машинного обучения, включая генерацию данных, обработку текста, оптимизацию алгоритмов и сегментацию изображений. Исследования в этой области помогают разрабатывать новые модели и методы, которые могут улучшить работу алгоритмов машинного обучения и расширить их применение в различных задачах.
4. Теория чисел. Исследования в области разбиений числа помогают расширить знания о структуре числовых систем и развивать новые методы анализа чисел.

Исследования в области разбиений числа являются неотъемлемой частью многих отраслей науки: комбинаторика, машинное обучение, криптография, квантовые вычисления и пр.

Постановка задачи

Целью выпускной квалификационной работы является разработка системы для вычисления и анализа чисел, связанных с разбиениями.

Для достижения поставленной цели необходимо решить следующие задачи:

1) изучить предметную область;
2) разработать и реализовать алгоритм нахождения характеристики вектора разбиения и ранга характеристики вектора разбиения;
3) разработать и реализовать алгоритм нахождения частного степени разбиения;
4) разработать и реализовать алгоритм нахождения самоассоциированного разбиения, нечетных разбиений и нечетных произведений для самоассоциированных разбиений;
5) провести анализ полученных результатов.

Структура и содержание работы

Работа состоит из введения, пяти глав, заключения, списка литературы и приложения. Объем работы составляет 47 страниц, объем списка литературы – 20 источников.

В первой главе приведен обзор научной литературы.

Во второй главе описываются основные теоретические сведения.

Третья глава посвящена разработке алгоритмов и проектированию приложения, описаны функциональные и нефункциональные требования.

В четверной главе представлена программа реализации алгоритмов и приложения.

В пятой главе проводится оценка корректности работы алгоритма, тестирование приложения, анализ полученных результатов.

В приложении А содержатся спецификации вариантов использования приложения.

В приложении Б представлены графики смещения пиков рангов натурального числа.

В приложении В приведены первые 22 строки файла CSV.

В заключении сделаны выводы о проделанной работе и полученных результатах.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Обзор литературы

В книге «Теория разбиений» [1] автор Эндрюс Г. рассматривает вопросы, связанные с теорией разбиений. Теория разбиений является разделом комбинаторики, который изучает различные способы разбиения натуральных чисел на слагаемые.

В данной книге автор вводит основные понятия и определения теории разбиений, такие как функция разбиения, диаграмма Юнга и др. Подробно рассматривается методы решения задач, связанных с разбиениями, включая нахождения числа разбиений данного натурального числа, нахождения суммы элементов разбиения и т.д.

Книга «Теория характеров и представления групп» [2] – это научный труд по математике, изучающий группы и их представления. В книге рассматриваются различные методы и подходы к исследованию групп. Особое внимание уделяется теории характеров. Помимо этого, Г. Фробениус изучает связь между симметрией группы и свойствами ее элементов, он вводит понятия характера и представления группы, которые помогают изучать алгебраические свойства групп.

Характер группы – это функция, определенная на элементах группы, которая используется для анализа структуры группы. В частности, характеры используются для классификации неприводимых представлений группы.

Представление группы – это способ записи элементов в виде матриц, сохраняющих групповую структуру. Каждое представление имеет свой характер, который можно использовать для изучения свойств этого представления.

Симметрия групп может быть применена во многих отраслях науки: физике, химии, квантовых вычислениях. Например, как симметрия объекта влияет на его свойства, такие как проводимость или теплопроводность; создание материалов с определенной симметрией, которые будут обладать определенными физическими свойствами.

Задача факторизации – задача о разложении натурального числа на простые множители. Данная задача считается вычислительно сложной и на данный момент не существует не квантового алгоритма для ее решения. Решение этой задачи применяется в теории чисел, квантовых вычислениях и эллиптических кривых. Среди известных не квантовых алгоритмов: метод факторизации Ферма, перебор возможных делителей, алгоритм Ленстры, решето числового поля и многие другие. Большая вычислительная сложность лежит в задачи криптостойкости некоторых алгоритмов шифрования с открытым ключом, таких как RSA [3]. Для решения задачи факторизации больших натуральных чисел необходимо разрабатывать новые методы и алгоритмы.

В таблице 1 приведены числа RSA и их факторизация. Для чисел RSA-100 – RSA-155 работали разные команды, но неизменным участником оставался Арьен Ленстра. Для большинства уже факторизованных чисел применялся общий метод решета числового поля.

Таблица 1 – Достижения в области факторизации больших чисел

Число	**Количество десятичных цифр**	**Стоимость**	**Дата факторизации**
RSA-100	100	–	1 апреля 1991 г.
RSA-110	110	–	Апрель 1992 г.
RSA-120	120	–	Июнь 1993 г.
RSA-129	129	100$	Апрель 1994 г.
RSA-130	130	–	10 апеля 1996 г.
RSA-140	140	–	2 февраля 1999 г.
RSA-155	155	–	22 августа 1999 г.
RSA-160	160	–	1 апреля 2003 г.
RSA-170	170	–	29 декабря 2009 г.
RSA-576	174	10000$	3 декабря 2003 г.
RSA-180	180	–	8 мая 2010 г.
RSA-704	212	30000$	2 июня 2012 г.
RSA-220	220	–	Май 2016 г.
RSA-230	230	–	15 августа 2018 г.

Окончание таблицы 1

Число	Количество десятичных цифр	Стоимость	Дата факторизации
RSA-232	232	–	17 февраля 2020 г.
RSA-768	232	–	12 декабря 2009 г.
RSA-270	270	–	–
RSA-896	270	75000$	–
RSA-1024	309	100000$	–
RSA-1536	463	150000$	–
RSA-2048	617	20000$	–

В статье «Сведение задачи факторизации натурального числа к задаче разбиения числа на части» [4] Е.А. Ваулин и М.С. Назаров рассматривают методы решения задачи факторизации натуральных чисел, то есть разложения числа на его простые множители. Факторизация натурального числа является важной и актуальной задачей в области теории чисел и криптографии.

Авторы статьи предлагают новый подход к решению этой задачи, основанный на сведении ее к задаче разбиения данного числа на несколько меньших чисел (частей). Для этого они вводят понятие «эффективной ширины» числа, которая равна количеству его простых делителей.

Статья начинается с введения основных понятий и определений, а также формулировки задачи факторизации и задачи разбиения числа. Далее авторы предлагают алгоритм сведения одной задачи к другой, основанный на разбиении числа на эффективные широты.

Затем авторы приводят результаты численных экспериментов, которые подтверждают эффективность предложенного метода. В частности, они показывают, что на некоторых наборах данных предложенный метод работает быстрее, чем известные алгоритмы факторизации.

Однако статья не лишена недостатков. Например, авторы не приводят детального анализа сложности предложенного алгоритма, что могло бы помочь оценить его эффективность в сравнении с другими методами факторизации чисел.

«Analytic Combinatorics» [5] – это книга, написанная Ф. Флайоле и Р. Седжвиком, опубликованная издательством Кембриджского университета в 2009 году. Книга посвящена комбинаторному перечислению и использует производящие функции и комплексный анализ для понимания темпов роста числа комбинаторных объектов.

Книга состоит из трех частей. Первая часть охватывает символический метод в комбинаторике, где классы комбинаторных объектов ассоциируются с формулами, описывающими их структуры, и затем эти формулы переосмысливаются для получения производящих функций или экспоненциальных производящих функций классов.

Вторая часть касается применения инструментов комплексного анализа к производящей функции для анализа асимптотики количества объектов в комбинаторном классе. Здесь используются интегральная формула Коши и знания особенностей функции для оценки результирующих интегралов.

Третья часть исследует поведение случайных комбинаторных структур с использованием того же набора инструментов. В книге рассматриваются такие темы, как последовательности, формальные языки, целочисленные разделы, композиции, перестановки, графы и пути в графах, решетчаты пути.

В книге «Integer Partitions» [6] рассмотрены методы и алгоритмы для вычисления количества разбиений, такие как включения-исключения, метод производящих функций и метод рекуррентных соотношений. Так же рассматривается связь между разбиениями целых чисел и другими областями математики: комбинаторика, теория групп и теория представлений, теория чисел, физика.

2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Под разбиением натурального числа представляют его разложение в виде суммы натуральных слагаемых. Разбиения одного числа считается равными, если эти разбиения отличаются порядком слагаемых.

В [1] приведено выражение Харди-Рамануджана для $p(n)$, где $p(n)$ обозначает количество всех разбиений натурального числа n.

Асимптотическая формула (1) полученная из теоремы Харди-Рамануджана-Радемахера дает количество разбиений натурального числа с точностью 0,004:

$$p(n) \sim \frac{1}{4n\sqrt{3}} exp\left\{\pi\left(\frac{2n}{3}\right)^{\frac{1}{2}}\right\}, \tag{1}$$

при $n \to \infty$.

Вектор разбиения

Каждое разбиение натурального числа n будет характеризоваться вектором $\vec{\varkappa} = (\varkappa_1, \dots, \varkappa_n)$, где:

$$0 \le \varkappa_1 \le \varkappa_2 \le \dots \le \varkappa_n \text{ и } \sum_{i=1}^{n} \varkappa_i = \frac{1}{2}n(n+1). \tag{2}$$

Определение 1. Итак, каждое разбиение натурального числа n будет характеризоваться вектором $\vec{\varkappa} = (\varkappa_1, \dots, \varkappa_n)$, с условиями (3 и 4). Этот вектор будем называть векторным разбиением. Любое разбиение будет полностью определять вектор разбиения, а задание такого вектора с условиями (2) будет полностью определять разбиение.

От вектора к разбиению

Переход от вектора $\vec{\varkappa} = (\varkappa_1, \dots, \varkappa_n)$ к разбиению происходит следующим образом.

Рассмотрим числа $0 \le \varkappa_1 \le \varkappa_2 - 1 \le \varkappa_3 - 2 \le \dots \le \varkappa_n - (n-1)$. Пусть m из них отличны от 0, тогда получаем числа $\lambda_1 = \varkappa_n - (n-1)$, $\lambda_2 = \varkappa_{n-1} - (n-2), \dots, \lambda_m = \varkappa_{n-(m-1)} - \big(n - (n-m)\big) \le 1$, которые образуют разбиение числа n.

От разбиения к вектору

Если наоборот нам задано разбиение числа n, где $\lambda_1 \geq \cdots \geq \lambda_m \geq 1$ и $n \ = \lambda_1 + \cdots + \lambda_n$, тогда для любого $i \in \{1, \dots, n-m\}$:

$$\varkappa_i = i - 1, \tag{3}$$

а для любого $i \in \{\ (n-m) + 1, \dots, n\}$:

$$\varkappa_i = \lambda_{i-(n-m)} + i - 1. \tag{4}$$

2.1. Характеристика и ранг характеристики вектора разбиения

Фробениус вводит понятие характеристики вектора $\vec{\varkappa}$ [2].

Каждое из n чисел вектора $\vec{\varkappa}$ не превосходит $2n-1$, так как $\vec{\varkappa}_n$ достигает максимального значения в $2n-1$, когда остальные $n-1$ чисел имеют минимальные значения $\vec{\varkappa}_1 = 0, \vec{\varkappa}_2 = 1, \dots, \vec{\varkappa}_{n-1} = n-2$.

Пусть $a_1, a_2, \dots, a_n$ – числа $0, 1, \dots, n-1$ в какой-либо последовательности. В таком случае $(n-1) - a_1, (n-1) - a_2, \dots, (n-1) - a_n$ – те же самые числа, но может быть в другой последовательности.

Пусть $n-r$ из них встречается среди чисел $\vec{\varkappa}$, например $(n-1) - a_{r+1}, \dots, (n-1) - a_n$.

Итак, имеем:

$$\{\varkappa_1, \dots, \varkappa_n\} \cap \{0, \dots, n-1\} = \{(n-1) - a_{r+1}, \dots, (n-1) - a_n\}. \tag{5}$$

Далее преобразуем это соотношение (5), а именно умножим его на -1, а затем прибавим к полученному $(n-1)$:

$$\{(n-1) - \varkappa_1, \dots, (n-1) - \varkappa_n\} \cap \{0, \dots, n-1\} = \{a_{r+1}, \dots, a_n\}. \tag{6}$$

Далее $\{a_1, \dots, a_r\} = \{0, \dots, n-1\} \backslash \{(n-1) - a_{r+1}, \dots, (n-1) - a_n\}$.

Снова умножим (6) на -1, а затем прибавим к полученному $(n-1)$, тогда:

$$\{a_1, \dots, a_r\} = \{0, \dots, n-1\} \backslash \{a_{r+1}, \dots, a_n\}. \tag{7}$$

В (7) получен первый вектор из определения характеристики вектора разбиения (8).

Остальные r чисел вектора $\vec{\varkappa}$ не менее чем n. Обозначим их через $n+b_1,\dots,n+b_r$.

Определение 2. Фробениус называет:

$$\begin{pmatrix} a_1 & a_2 & \cdots & a_r \\ b_1 & b_2 & \dots & b_r \end{pmatrix}, \tag{8}$$

характеристикой вектора $\vec{\varkappa}$ и для определенности полагает $a_1 < a_2 < \cdots <$ $< a_r$, $b_1 < b_2 < \cdots < b_r$.

Далее будем обозначать характеристику (8) так:

$$\begin{pmatrix} a_1 & a_2 & \cdots & a_r \\ b_1 & b_2 & \dots & b_r \end{pmatrix} = \begin{pmatrix} \vec{a} \\ \vec{b} \end{pmatrix}. \tag{9}$$

Замечание 1. Для всех $i \in \{1,\dots,n-r\}$ из (9):

$$\varkappa_i = (n-1) - a_{r+i}. \tag{10}$$

А также для всех $i \in \{(n-r)+1,\dots,n\}$ из (9):

$$\varkappa_i = n + b_{i-(n-r)}. \tag{11}$$

Ранг характеристики

Определение 3. Число r будем называть рангом характеристики (8).

Отметим, что $\sum_{i=1}^{n} \varkappa_i = \frac{1}{2}n(n+1)$ и, складывая это с (10) и (11) получим:

$$\sum_{i=1}^{r} a_i + \sum_{i=1}^{r} b_i = n - r. \tag{12}$$

Из (12) имеем:

$$r \le \sqrt{n}. \tag{13}$$

В (13) получено свойства ранга разбиения.

Степень разбиения

Обозначение 1. Пусть определитель Вандермонда (14) со второй строчкой $|x_1, x_2, \dots, x_t|$:

$$\Delta(x_1, x_2, \dots, x_t) = \prod_{1 \le i \le j \le t} \left(x_j - x_i\right). \tag{14}$$

Обозначение 2. Если вектор $\vec{x} = (x_1, x_2, \dots, x_t)$ имеет неотрицательные целые компоненты, то обозначим $\vec{x}! = x_1! \ \dots \ x_t!$.

Определение 4. Частным степени разбиения будем называть число:

$$z(\vec{\varkappa}) = \frac{\vec{\varkappa}!}{\Delta(\vec{\varkappa})} = \frac{\vec{a}!\vec{b}!\prod_{i,j=1}^{r}(a_i+b_j+1)}{\Delta(\vec{a})\Delta(\vec{b})}. \quad (16)$$

2.2. Самоассоциированные разбиения

Определение 5. Разбиения с характеристикой $\binom{\vec{a}}{\vec{a}}$ называются самоассоциированными.

Ранги самоассоциированных разбиений

Для самоассоциированного разбиения согласно соотношению (12) имеем:

$$2\sum_{i=1}^{r} a_i = n - r. \quad (17)$$

Поэтому $r \equiv n\ (mod\ 2)$, то есть n и r либо оба нечетны, либо оба четны.

Нечетные разбиения для самоассоциированных разбиений

С каждым самоассоциированным разбиением с характеристикой $\binom{\vec{a}}{\vec{a}}$ можно связать набор $(2a_1 + 1, 2a_2 + 1, \dots, 2a_r + 1) = \vec{O}(\vec{a})$, который состоит из различных нечетных чисел.

Кроме того, из соотношения (16) получаем $\sum_{i=1}^{r}(2a_i + 1) = n$, т.е. $\vec{O}(\vec{a})$ – разбиение n.

Нечетные произведения для самоассоциированных разбиений

Определение 6. Назовем нечетным произведением для заданного самоассоциированного разбиения число:

$$P(\vec{a}) = (-1)^{\frac{n-r}{2}} \prod_{i=1}^{r}(2a_i + 1). \quad (18)$$

Знак произведения определяется числом:

$$(-1)^{\frac{n-r}{2}} = \begin{cases} 1, \text{при } r \equiv n\ (mod\ 4) \\ -1, \text{при } r \equiv n - 2\ (mod\ 4) \end{cases}, \quad (19)$$

поэтому назовем нечетное произведение положительным, если $r \equiv n \ (mod\ 4)$, отрицательным, если это не выполняется.

Степени для самоассоциированного разбиения в случае положительных нечетных произведений

Пусть вектор разбиения $\vec{\varkappa}$ задает самоассоциированное разбиение с положительным нечетным произведением. Тогда частное разбиение по формуле (16) равно:

$$z(\vec{\varkappa}) = \frac{\vec{a}!\,\vec{a}!\prod_{i,j=1}^{r}\left(a_i + a_j + 1\right)}{\Delta(\vec{a})\Delta(\vec{a})} = \left(\frac{\vec{a}!}{\Delta(\vec{a})}\right)^2 \prod_{i,j=1}^{r}\left(a_i + a_j + 1\right). \tag{20}$$

Далее преобразуем (20), используя (18):

$$\prod_{i,j=1}^{r}\left(a_i + a_j + 1\right) = \left(\prod_{i=1}^{r}(2a_i + 1)\right)\left(\prod_{1 \le i \le j \le r}^{r}\left(a_i + a_j + 1\right)\right)^2 = \tag{21}$$

$$= P(\vec{a})\left(\prod_{1 \le i \le j \le r}^{r}\left(a_i + a_j + 1\right)\right)^2.$$

3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОЙ СИСТЕМЫ

3.1. Анализ требований

Разрабатываемая система для изучения разбиений – это настольное приложение, которое позволяет, исходя из полученного от пользователя диапазона натуральных чисел, выявлять закономерности в полученных данных. В связи с этим, система должна удовлетворять ряду функциональных и нефункциональных требований.

Функциональные требования

Функциональные требования – это спецификации функций или сервисов, которые должны быть реализованы в программном продукте. Они определяют поведение системы в ответ на определенные входные данные или действия пользователя. Функциональные требования описывают, что система должна делать, какие функции и возможности она должна предоставлять.

Ниже представлен список функциональных требований.

1. Пользователю доступен ввод начала и конца диапазона натуральных чисел.
2. Система должна выводить графики, таблицы, исходя из введенного натурального числа.
3. Система должна предоставлять сохранение результатов исследования на диск.
4. Система должна позволять работать с разбиениями одного числа, с самоассоциированными разбиениями.
5. Пользователь может выбирать с каким показателем разбиения он будет работать (частое степени разбиения, ранг характеристики)
6. Система должна предупреждать пользователя о неверно введенном числе.
7. При работе с разбиениями одного числа система отображает следующие колонки таблицы: число, разбиение, вектор, ранг, характеристика, частное степени разбиения, самоассоциированные разбиения.

8. При работе с самоассоциированными разбиениями система отображает следующие колонки таблицы: число, разбиение, вектор, ранг, характеристика, частное степени разбиения, самоассоциированные разбиения.

9. Система должна предупреждать о неверно введенном диапазоне.

Нефункциональные требования

Нефункциональные требования определяют качественные атрибуты системы, такие как производительность, надежность, безопасность, масштабируемость и другие аспекты, которые не связаны прямо с функциональностью программного продукта.

Нефункциональные требования описывают «как» система должна работать, и обычно не имеют конкретных ответов «да» или «нет», а представляют собой оценку качества или характеристики системы.

Ниже представлен список нефункциональных требований.

1. Система должна быть реализована на языке Python.
2. Система должна поддерживаться ОС Windows.
3. В системе должна быть предусмотрена возможность сохранения в форматах .xlsx для табличного представления.
4. Разбиения хранятся в файле .csv.

3.2. Проектирование приложения

На рисунке 1 представлена диаграмма вариантов использования.

Основной актер, взаимодействующий с системой – пользователь. Пользователь имеет возможность ввести диапазон натуральных чисел, проанализировать введенный данные и получить по ним результаты. Пользователь имеет возможность сохранять табличные данные. Пользователь может проанализировать не только все описанные аспекты разбиения, но и каждый по отдельности, если это необходимо [7].

Данная диаграмма позволяет понять, как система взаимодействует с пользователем и какие функции поддерживает система.

Это позволяет создать эффективное и удобное для пользователей решение.

Рисунок 1 – Диаграмма вариантов использования

Спецификация вариантов использования – это план взаимодействия главного актера с приложением, в котором описаны шаги для достижения результата, включая обработку исключительных сценариев или сценариев, ведущих к ошибке, называемые альтернативными.

В таблице 2 приведен пользовательский сценарий работы с системой, на примере работы с показателем частного степени разбиения. У системы есть один основной поток и два альтернативных. Альтернативные сценарии нужны для обработки исключающих ситуаций, таких как: неверно введенное значение в поле ввода; не выбранная категория. Остальные спецификации приведены в приложении А.

Таблица 2 – Спецификация варианта использования «Частное степени разбиения»

Прецедент: получить частное степени разбиения
ID: 1
Краткое описание Получение результатов по частному степени разбиения
Основной актер Пользователь
Второстепенные актеры: Нет
Предусловие Запущено приложение, и пользователь находится на «Разбиения одного числа»
Основной поток 1. Пользователь вводит натуральное число 2. Пользователь выбирает в поле выбора «Частное степени разбиения» 3. Пользователь нажимает кнопку «Показать результат» 4. Система отображает таблицу и график
Постусловие: показана таблица и график
Альтернативные потоки I. Ошибка ввода 1. Приложение очищает поле ввода 2. Приложение выводит сообщение об ошибке ввода II. Ошибка выбора категории 1. Приложение выводит сообщение об отсутствии выбора категории

Для представления пользовательского интерфейса на рисунках 2 и 3 приведен макет приложения на вкладках «Разбиение одного числа» и «Самоассоциированные разбиения» соответственно. Основные компоненты макета – это поле ввода, меню, кнопка, поле выбора, таблица и рисунок графика. В поле ввода пользователь должен ввести натуральное число, выбрать с каким показателем он хочет работать. Для работы с самоассоциированными разбиениями уже необходимо ввести два натуральных числа – диапазон и нажать на кнопку.

Макеты имеют одинаковую структуру и отличаются только одним виджетом, так как с самоассоциированными разбиениями лучше работать на диапазоне.

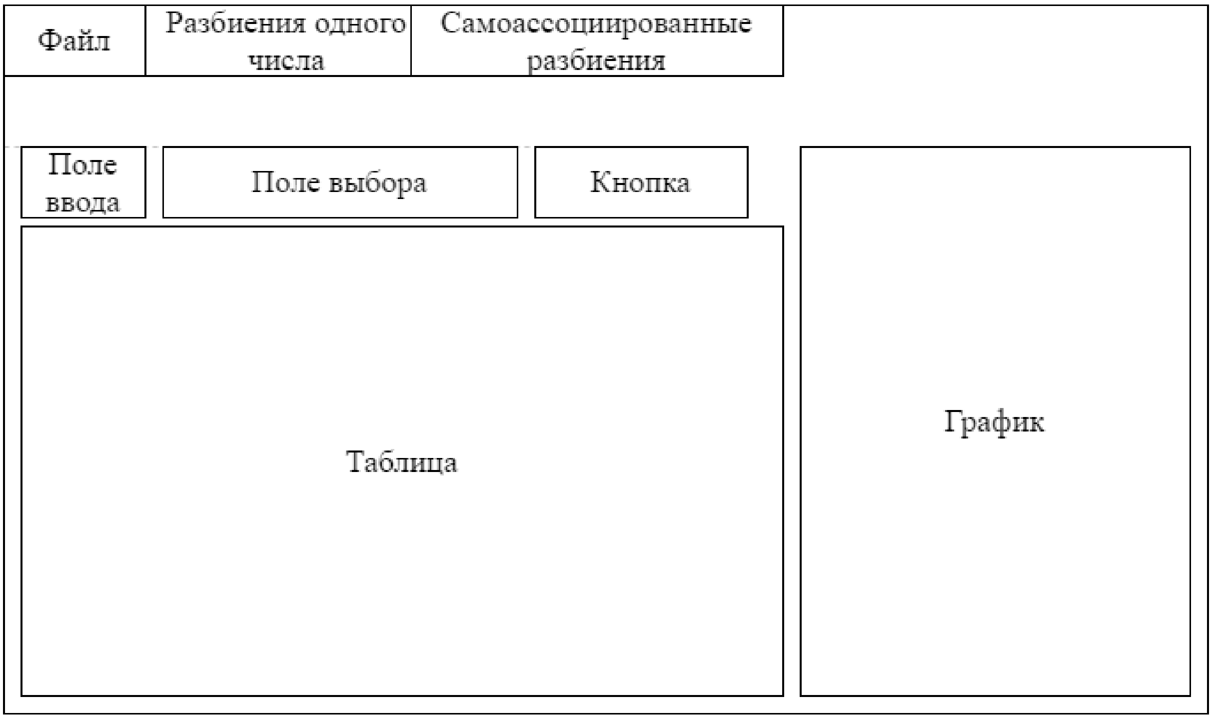

Рисунок 2 – Макет приложения (разбиения одного числа)

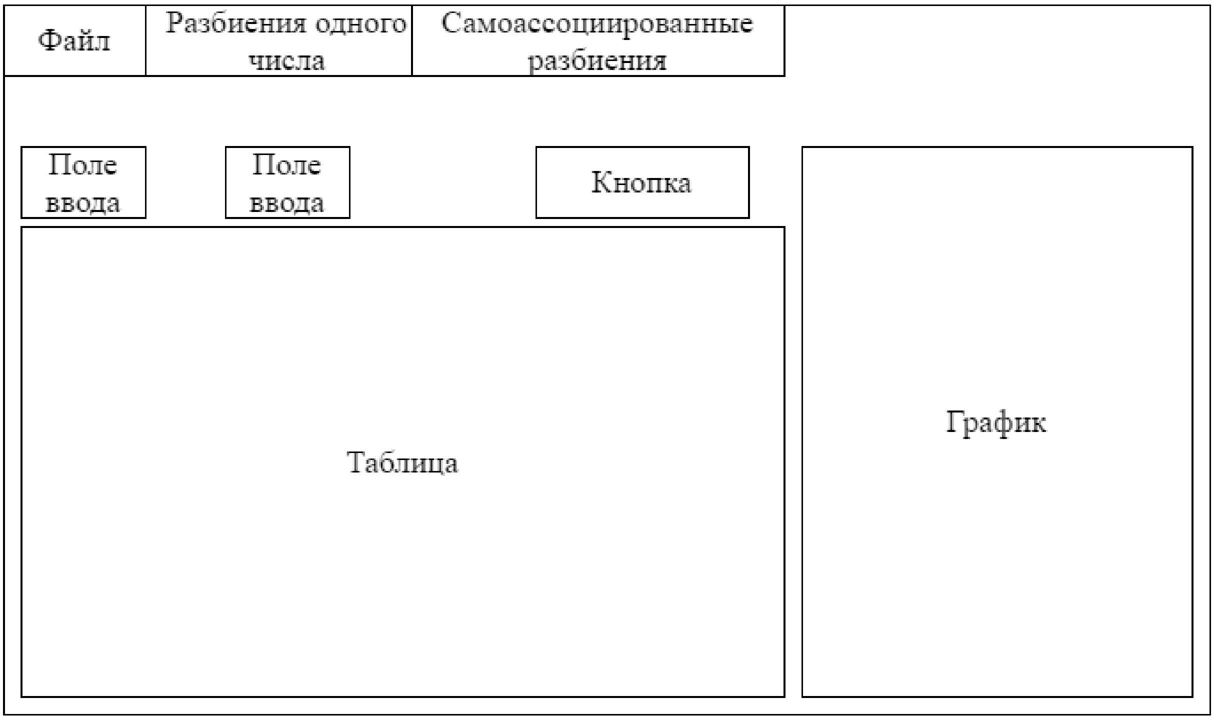

Рисунок 3 – Макет приложения (самоассоциированные разбиения)

3.3. Разработка алгоритмов

Алгоритмы решено представить в виде блок-схем и псевдокода. Такой вид представления помогает визуально изобразить процессы. Так же это поможет объяснить последовательность действий и оставляет возможность для совершенствования разработанных решений.

Введем следующие обозначения:

1) n – натурально число;
2) $Partition$ – разбиение числа n;
3) m – длина $Partition$;
4) n_1 – список элементов от 0 до $n-1$.

Алгоритм перехода от разбиения к вектору

Переход от разбиения к вектору можно представить в виде двух невложенных циклов. Алгоритм приведен на рисунке 4.

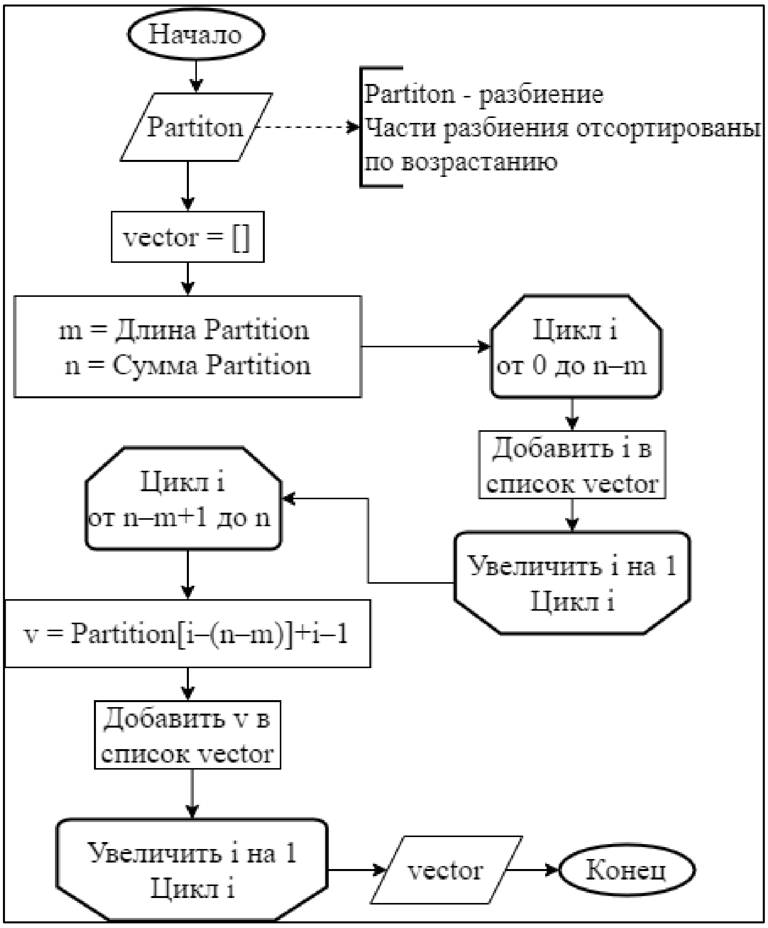

Рисунок 4 – Алгоритм перехода от разбиения к вектору разбиения

Алгоритм характеристики разбиения и ранга характеристики

На рисунке 5 представлена блок-схема вычисления характеристики разбиения. За «&» и «\», будем принимать операции пересечения и разности множеств соответственно.

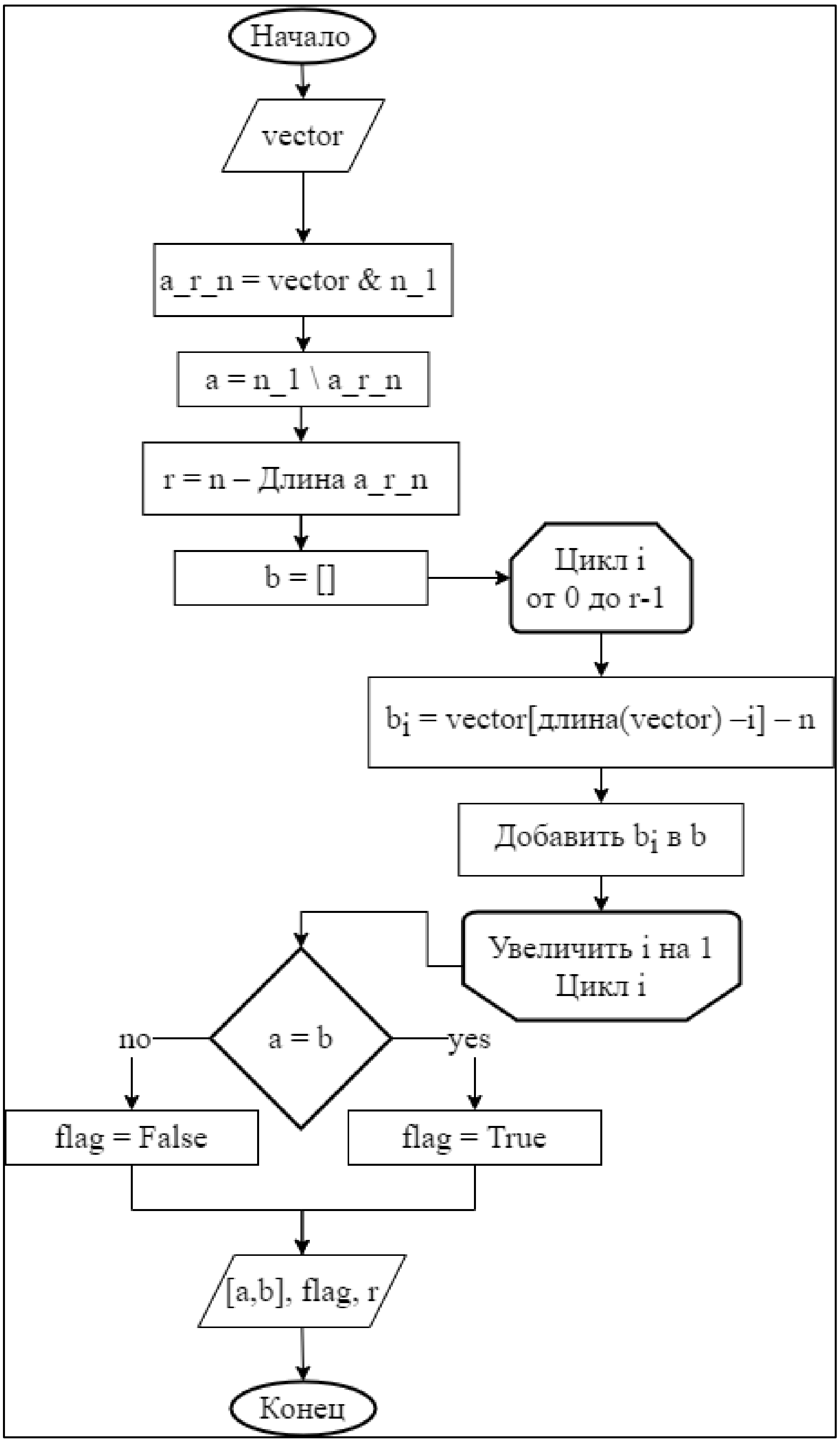

Рисунок 5 – Блок-схема вычисления характеристики и ранга разбиения

Характеристика разбиения вычисляется через вектор разбиения и для ее вычисления используются теоретико-множественные операции: пересечение и разность, представленные на рисунке 6.

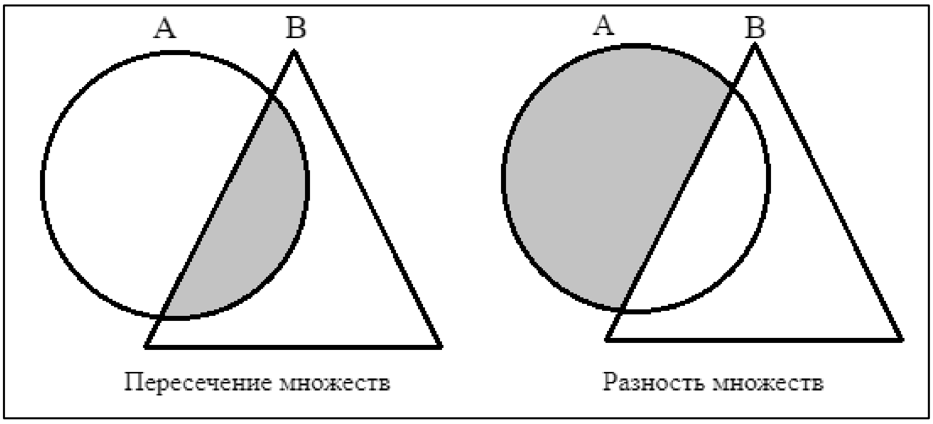

Рисунок 6 – Теоретико-множественные операции

Также, согласно определению 5, помимо ранга характеристики вектора разбиения и характеристики вектора разбиения возвращается $flag$. Если $flag = True$, то это означает, что разбиение самоассоциированное.

Следует учесть, что множества a и b имеют одинаковую длину, а ранг r удовлетворяет условию (12).

Алгоритм вычисления частного степени разбиения

Для вычисления степени разбиения необходимо вычислить факториал каждого вектора характеристики (a, b), а также составить матрицу и вычислить определитель Вандермонда (13).

В листинге 1 приведен псевдокод для вычисления факториала.

Листинг 1 – Псевдокод для факториала

```
Algorithm Factorial (Num ∈ N )
   F := 1
   for i := 1 to Num
      F := F * i
   return F
```

Перед тем как вычислять определитель Вандермонда необходимо составить матрицу, представленную на рисунке 7.

$$
V = V(x_0, x_1, \cdots, x_m) = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^n \end{bmatrix}
$$

Рисунок 7 – Матрица Вандермонда

Матрица Вандермонда представляет собой матрицу с членами геометрической прогрессии. В листинге 2 приведен псевдокод для составления матрицы Вандермонда.

Листинг 2 – Псевдокод для составления матрицы Вандермонда

```
Algorithm Vandermond_matrix(a)
  n := Length(a)
  matrix(n,n) := 0
  for i := in 1 to n
    for j := in 1 to n
      matrix(i,j) := i^j
  return matrix
```

Определитель Вандермонда равен произведению всевозможных положительных разностей, что приведено в формуле (13). В листинге 3 приведен псевдокод для вычисления определителя Вандермонда.

Листинг 3 – Псевдокод для вычисления определителя Вандермонда

```
Algorithm Determinate_Vandermond(matrix)
  det := 1
  for i := 1 to n
    for j := i to n
      det := det * (matrix(j,i) − matrix(i,i))
      for k := i to n
        matrix(j,k) := matrix(j,k) − matrix(i,k) * (matrix(j,i) / matrix(i,i))
  return det
```

В листинге 4 приведен псевдокод для вычисления частного степени разбиения. Сначала необходимо получить произведение факториалов и определитель Вандермонда для каждого вектора характеристики $(\vec{a}, \vec{b})$. Далее произведения факториалов перемножаются и помножаются на $\prod_{i,j=1}^{r}(a_i + b_j + 1)$, где r – ранг характеристики (длина вектора $\vec{a}$ или $\vec{b}$). в дальнейшем, получившиеся произведение нужно поделить на произведение определителей Вандермонда векторов характеристики.

Листинг 4 – Псевдокод для вычисления частного степени разбиения

$\boldsymbol{Algorithm}\ Degree\left(n \in \mathrm{N},\quad charecteristic \in \mathrm{N}^{2\times Lenght(a_{facrotial})}\right)$
$\quad a_{facrotial} \coloneqq \boldsymbol{Factorial}(harecteristic[1])$
$\quad b_{factorial} \coloneqq \boldsymbol{Factorial}(harecteristic[2])$
$\quad a_{Vandermond} := \boldsymbol{Vandernond\_Matrix}(charecteristic[1])$
$\quad b_{Vandermond} := \boldsymbol{Vandernond\_Matrix}(charecteristic[2])$
$\quad determinate_a := Determinate\_Vandermond(\ a_{Vandermond})$
$\quad determinate_b := Determinate\_Vandermond(\ b_{Vandermond})$
$\quad Pow\_a_{facrotial} \coloneqq 1$
$\quad \boldsymbol{for}\ i \coloneqq 1\ \boldsymbol{to}\ Lenght(a_{facrotial})$
$\qquad Pow\_a_{facrotial} \coloneqq Pow\_a_{facrotial} * i$
$\quad \boldsymbol{for}\ i \coloneqq 1\ \boldsymbol{to}\ Lenght(b_{facrotial})$
$\qquad Pow\_b_{facrotial} \coloneqq Pow\_b_{facrotial} * i$
$\quad power \coloneqq 1$
$\quad \boldsymbol{for}\ i \coloneqq 1\ \boldsymbol{to}\ \boldsymbol{Length}(charecteristic[1])$
$\qquad \boldsymbol{for}\ j \coloneqq 1\ \boldsymbol{to}\ \boldsymbol{Length}(charecteristic[2])$
$\quad power \coloneqq power * (charecteristic(1, i) + charecteristic(1, j) + 1)$
$\quad Z := (\ Pow_{a_{facrotial}} * \quad Pow_{b_{facrotial}} * power)/(determinate_a * determinate_b)$
$\quad degZ \coloneqq \boldsymbol{Factorial}(n)/Z$
$\quad \boldsymbol{return}\ degZ$

3.4. Представление полученных вычислений

Полученные при вычислениях данные будут храниться в файле формата CSV (Comma-Separated Values) – это текстовый формат для представления табличных данных.

Преимущества хранения данных в таком формате – это:

1) легкость чтения;
2) поддержка большинства программ (Microsoft Excel, Google Sheets, Python, R и др.);
3) экономия места;
4) прозрачность и расширяемость;
5) поддержка различных типов данных.

В таблице 3 представлена структура файла csv: поля, и типы данных.

Таблица 3 – Структура файла CSV

Поле	Тип данных
Натуральное число	Целое натуральное число
Количество слагаемых	Целое натуральное число
Разбиение	Одномерный массив
Вектор разбиения	Одномерный массив
Ранг	Целое натуральное число
Характеристика вектора разбиения	Двумерный массив
Частное степени разбиения	Целое натуральное число
Самоассоциированные разбиения	Число 0 или 1

Файл CSV хранит в себе натуральные числа в диапазоне от 1 до 64. Для каждого разбиения из этого диапазона посчитаны: вектор разбиения, ранг, характеристика вектора разбиения, частное степени разбиения и найдены самоассоциированные разбиения. Размер файла составляет 12308138 строк и 8 колонок. Вес файла 4,16 ГБ.

Чтобы создать файл CSV нужно спроектировать класс `Partition`, который будет считать все показатели разбиения. Диаграмма класса представлена на рисунке 8.

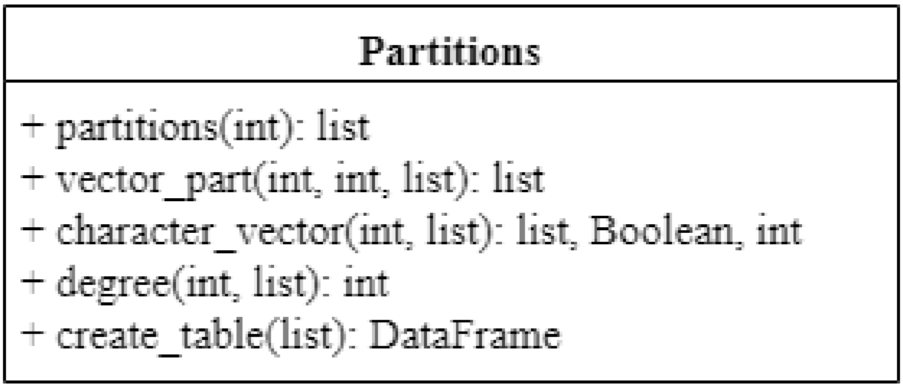

Рисунок 8 – Диаграмма класса `Partition`

Из диаграммы видно, что у класса нет полей, но есть 5 публичных методов, в скобках представлены типы входных параметров, за двоеточием тип выходных параметров.

Публичный метод класса в объектно-ориентированном программировании (ООП) является методом, который может быть вызван извне класса, то есть из других частей программы или модулей. Публичные методы предоставляют внешний интерфейс для взаимодействия с объектом класса. Они обычно выполняют определенные действия или операции над состоянием объекта.

Класс состоит из пяти публичных методов.

1. Метод `partitions` принимает целое число и возвращает список. Он отвечает за генерацию разбиений, основываясь на переданном целом числе.

2. Метод `vector_part` принимает три параметра: два целых числа и список: натуральное число, количество слагаемых, список разбиений. Возвращает список – вектор разбиения.

3. Метод `character_vector` принимает целое число и список: натуральное число и его разбиение, возвращает список, булево значение и целое число: вектор разбиения, самоассоциированное разбиение, ранг разбиения.

4. Метод `degree` принимает целое число и список: натуральное число и его разбиение, возвращает целое число – частное степени разбиения.

5. Метод `create_table` принимает список – разбиение числа и возвращает объект DataFrame – таблицу со всеми показателями разбиения.

4. РЕАЛИЗАЦИЯ ПРОГРАММНОЙ СИСТЕМЫ

Реализация алгоритмов

Алгоритмы были реализованы на языке Python. Были использованы следующие типы и структуры данных:

1) int32 – 32-битное натуральное число в диапазоне $(-2147483648;\ 2147483647)$;

2) int64 – 64-битное натуральное число в диапазоне $(-9223372036854775808;\ 9223372036854775808)$;

3) float64 – значения с плавающей запятой двойной точности;

4) массив – структура данных, содержащая данные с однородным типом;

5) множество – неупорядоченная коллекция уникальных элементов.

Следует отметить, что все данные имеют целочисленный тип, использование float64 обмуслено тем, что при вычисление частного степени разбиения необходимо считать факториал больших чисел, а также их перемножать.

Разбиения хранятся в файле .csv. Рост разбиений, согласно формуле (1) имеет экспоненциальных рост. Так в таблице 4 приведен диапазон и количество данных (количество разбиений для каждого натурально числа из диапазона), для которых нужно посчитать атрибуты, описанные в главе 3.

Таблица 4 – Количество разбиений для вычислений

Диапазон	Количество разбиений
[1…10]	138
[10…20]	2617
[20…30]	26542
[30…50]	1272946
[1…50]	1295970
[1…100]	1644288537

Для получения разбиений была использована система GAP. GAP – это свободный, открытый и расширяемый пакет для вычислений дискретной абстрактной алгебре. GAP – это язык программирования, библиотеки из тысячи функций, реализующих алгебраические алгоритмы, написанные на языке GAP, а также библиотеки данных алгебраических объектов. Данная система широко применяется в исследованиях и преподавании для изучения групп и их представлений, колец, векторных пространств, алгебр, комбинаторных структур и пр. система распространяется свободно, включая источник, что позволяет расширить ее для индивидуального использования [8].

Переход от вектора разбиения – это самый первый шаг в анализе остальных показателей разбиения, так как именно вектор разбиения позволяет получить характеристику вектора. Вектор – массив с натуральными числами. Получение вектора разбиения показан в листинге 5.

Листинг 5 – Реализация алгоритма перехода от разбиения к вектору

```
def vector_part(n:int, summands:int, partition:list)->list:
    vector = []

    for i in range(n-summands):
        vector.append(i)
    while len(vector)!= n:
        for i in range(len(partition)):
            vector.append(partition[i]+(n-summands)+i)
    return(vector)
```

Характеристика вектора – это двумерный массив, из двух векторов одинаковой длины, получаемых за счет теоретико-множественных операций. Алгоритм получения вектора разбиения представлен в листинге 6.

Листинг 6 – Реализация алгоритма характеристики вектора разбиения

```
def character_vector(n:int, vector:list):
    flag = 0
    n_1 = [i for i in range(n)]
    characteristic = set(vector).intersection(set(n_1))
    r = n - len(characteristic)
    n_x = [n-1-i for i in vector]
    a_r = set(n_x).intersection(set(n_1))
    a = set(n_1).difference(set(a_r))
    a = list(a)

    b_r = [i for i in vector[-r:]]
    b = [i - n for i in b_r]
    c = []
    a.sort()
    b.sort()
    if a == b:
        flag = 1
    c.append(a)
    c.append(b)
    return c, flag
```

Алгоритм получения частного степени разбиения требует больше всего накладных расходов, так как производятся операции над большими числами. Такие операции как: произведение факториалов каждого элемента векторов $\vec{a}$ и $\vec{b}$; произведение определителей Вандермонда над векторами $\vec{a}$ и $\vec{b}$. В листинге 7 приведен псевдокод для вычисления частного степени разбиения.

Листинг 7 – Реализация алгоритма вычисления частного степени разбиения

```
def degree(n: int, characteristic: list)->int:
    a_factorial = [math.factorial(i) for i in characteristic[0]]
    b_factorial = [math.factorial(i) for i in characteristic[1]]
    a_factorial, b_factorial = np.array(a_factorial,dtype='float64'),
np.array(b_factorial,dtype='float64')
    a_factorial = reduce(lambda x, y: x*y, a_factorial)
    b_factorial = reduce(lambda x, y: x*y, b_factorial)
    a_factorial, b_factorial = np.array(a_factorial), np.array(b_factorial)
    a_Vander = np.vander(characteristic[0],increasing=True)
    b_Vander = np.vander(characteristic[1],increasing=True)
    a_detVander = int(np.linalg.det(a_Vander))
    b_detVander = int(np.linalg.det(b_Vander))
    power = 1
    for i in range(len(characteristic[0])):
        for j in range(len(characteristic[1])):
            power *= characteristic[0][i]+characteristic[1][j]+1
    Z = (a_factorial*b_factorial*power)/(a_detVander*b_detVander)
    deg_Z = math.factorial(n)/Z

    return int(deg_Z)
```

Вышеперечисленные алгоритмы стали определяющими методами в классе `Partition`. Дополнительно был реализован метод для представления вычислений в DataFrame [10]. Метод представлен в листинге 8.

Листинг 8 – Метод для организации данных в таблицу

```
def create_table(self, part):
    columns = ['Натуральное число n', 'Кол-во слагаемых', 'Разбиение',
'Вектор разбиения', 'Ранг',
              'Характеристика вектора разбиения', 'Частное степени разбие-
ния', 'Самоассоциированные разбиения']
    data = pd.DataFrame(columns=columns)

    for i in part:
        i.sort()
        n = sum(i)
        summands = len(i)

        vector = self.vector_part(n, summands, i)
        c, a, r = self.character_vector(n, vector)
        Z_deg = self.degree(n, c)

        data = data.append({'Натуральное число n': n,
```

```
                                'Кол-во слагаемых': summands,
                                'Разбиение': i,
                                'Вектор разбиения': vector,
                                'Ранг': r,
                                'Характеристика вектора разбиения': [c],
                                'Частное степени разбиения': Z_deg,
                                'Самоассоциированные разбиения': [a]},
ignore_index=True)

    return data
```

Реализация приложения

Для реализации приложения было решено использовать пакет Tkinter – это стандартный интерфейс Python для набора инструментов Tcl/Tk GUI. Данный интерфейс является кроссплатформенным. Tcl/Tk не является единой библиотекой, а состоит из нескольких модулей, каждый из которых обладает отдельной функциональностью [9].

Виджеты, используемые при создании приложения на языке Tcl/Tk:

1) поле ввода Entry;
2) элемент выбора Combobox;
3) кнопка Button;
4) таблица Treeview;
5) canvas.

Для позиционирования элементов был определен метод place, который позволяет отображать виджеты по координатным осям.

5. ТЕСТИРОВАНИЕ

5.1. Тестирование алгоритма

Были проведены вычисления в диапазоне $1 \leq n \leq 64$. Согласно формуле 1 были вычислены 12308138 значений. Все полученные результаты размещены в таблице формата CSV [12]. Часть таблицы приведена в приложении В.

Для вычисления была использована среда Google Collaboratory. В таблице 5 приведено время вычисления.

Таблица 5 – Время выполнения программы на диапазоне от 1 до 50

На CPU	**С использование ускорителя T4 GPU**
`CPU times: user 2min 12s, sys: 3.64 s, total: 2min 16s` `Wall time: 2min 20s`	`CPU times: user 1min 51s, sys: 7.06 s, total: 1min 58s` `Wall time: 1min 55s`

CPU times – это общее количество процессорного времени, затраченное на уровне пользователя.

Sys – общее время процессорного времени, затраченного на выполнение системных операций.

Wall time – общее время реального выполнения, затраченное на выполнение кода, включая время задержки, связанное с операциями ввода-вывода или другими задачами, которые могут замедлить выполнения [14].

Из таблицы 5 видно, что вычисления на диапазоне $1 \leq n \leq 50$ (1295970 разбиений) без ускорителя занимают 2 минуты 16 секунд процессорного времени, если использовать ускоритель, то алгоритм работает быстрее и вычисления занимают 1 минуту 58 секунд процессорного времени.

Далее приведено время выполнения без ускорителя.

1. На диапазоне $51 \leq n \leq 57$ (2829871 разбиений) процессорное время составило 4 минуты 35 секунд.

2. На диапазоне $58 \leq n \leq 60$ (2513507 разбиений) процессорное время составило 4 минуты 14 секунд.

3. На диапазоне $61 \leq n \leq 64$ (5668790 разбиений) процессорное время составило 7 минут 56 секунд.

Проверим, что ранг не превосходит квадратного корня из n, согласно формуле (13). Для это построим график, где по оси y будут значение ранга, а по оси x натуральное число n. К дополнению к этому построим график функции $rank \ = \sqrt{n}$.

На рисунке 9 представлен график зависимости максимального ранга разбиений для данного натурального числа. Из графика видно, что полученные ранги согласуются с формулой (13) и не превосходят $\sqrt{n}$.

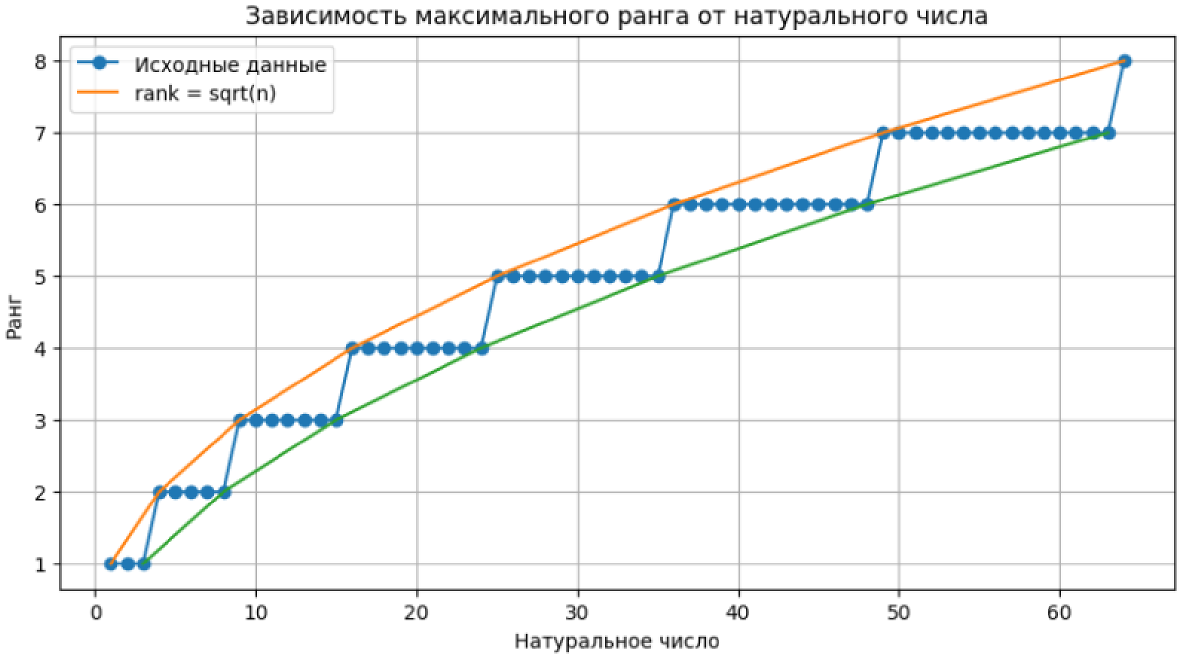

Рисунок 9 – График зависимости максимального ранга от натурального числа

5.2. Тестирование системы

В таблице 6 приведены результаты функционального тестирования приложения. Данные представлены в виде сравнительной оценки выполнения ключевых функций и сценариев, проведенных в ходе тестирования.

Каждая строка таблицы соответствует отдельному функциональному тесту с указанием статуса (пройден, не пройден, требует доработки) и ожидаемым результатом. Данная информация поможет определить качество функциональности приложения и выявить возможные проблемы.

Таблица 6 – Тестирование системы

№	Название теста	Ожидаемый результат	Результат
1.	Приключение между вкладками	При переключении между вкладками «Разбиения одного числа» и «Самоассоциированные разбиениями» система отчищает колонки в таблице и в таблице отображаются соответствующие колонки	Пройден
2.	Сохранение таблицы	По нажатию на соответствующий пункт меню будет сохранен файл конкретного расширения	Пройден
3.	Ввод некорректного натурального числа	Система выдаст всплывающее окно с предупреждением	Пройден
4.	Ввод некорректного диапазона	Система выдаст всплывающее окно с предупреждением	Пройден
5.	Ввод натурального числа	Система отобразит таблицу и график по запросу	Пройден
6.	Вывод таблицы при выборе «Разбиения одного числа»	Талица должна содержать следующие колонки: число, разбиение, вектор, ранг, характеристика, частное степени, самоассоциированное разбиение	Пройден
7.	Вывод таблицы при выборе «Самоассоциированные разбиения»	Талица должна содержать следующие колонки: число, разбиение, вектор, ранг, характеристика, частное степени. Последняя колонка представлена в виде произведения простых чисел	Пройден
8	Вывод графиков	Графики появляются после нажатия на кнопку справа от таблицы	Пройден

6. АНАЛИЗ ЧИСЕЛ, СВЯЗАННЫХ С РАЗБИЕНИЯМИ

Анализ рангов разбиения

Ранее на рисунке 9 показана зависимость максимального ранга от натурального числа.

На рисунке 10 приведен график зависимости рангов от количества разбиений для каждого натурального числа на диапазоне от 1 до 64.

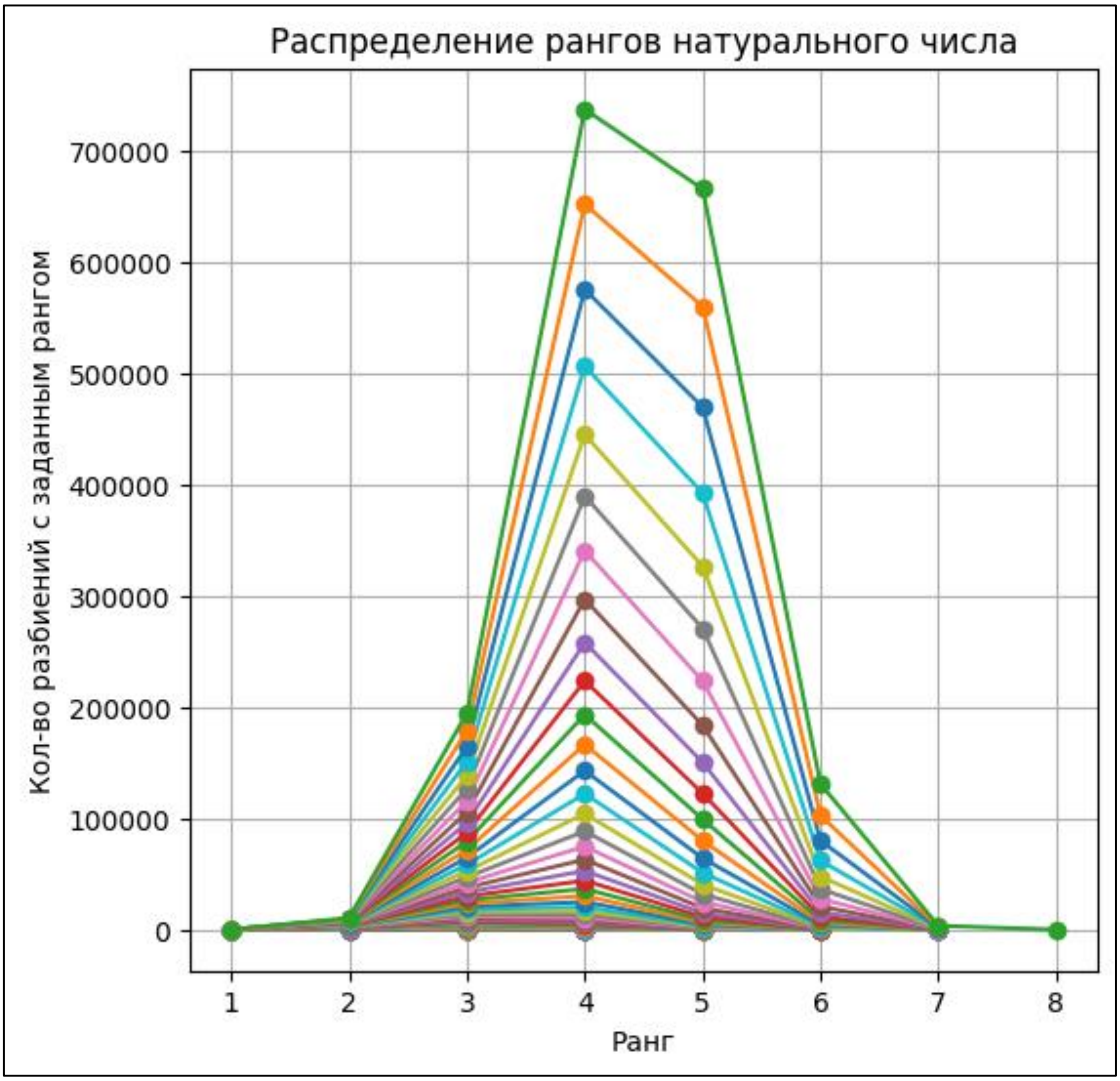

Рисунок 10 – График зависимости рангов от количества разбиений

Если рассматривать разбиения каждого числа по отдельности, то можно заметить, что для конкретного n количество разбиений с рангом равным 1 равно $n, n \in \mathbb{N}$. А также для числа n, из которого извлекается целочисленный квадратный корень, количество рангов равных $r = \sqrt{n}$ равно 1. Наблюдения представлены в таблице 7.

Можно заметить, что для каждого отдельного ранга количество разбиений с таким рангом тоже растет. Также согласно таблице 7 ранг равный единицы растет линейно: $r_1 = n$.

Таблица 7 – Начальные и конечные ранги

Натуральное число	**Ранг**	**Количество разбиений**
49	1	49
49	7	1
36	1	36
36	6	1
25	1	25
25	5	1
16	1	16
16	4	1
9	1	9
9	3	1
4	1	4
4	2	1

Следует отметить, что пик рангов смещается. Ранг – это половина от числа элементов характеристики, следовательно он зависит от количества частей разбиения. Смещения пиков представлены в приложении Б.

Полученные данные позволяют сделать следующие выводы.

1. Максимальный ранг разбиений для данного натурального числа вполне согласуется с теоретической оценкой и поэтому можно применять оценку максимального ранга для вычислений, связанных с различными ограничениями для разбиений. Например, для изучения частных степеней разбиения, самоассоциированных разбиений и их нечетных произведений.

2. Рисунок 10 показывает весьма неожиданное свойство ранга разбиения от количества разбиений. Это свойство приводит к предположению о нормальном распределении рангов разбиений. Однако, окончательные выводы пока делать преждевременно.

Анализ самоассоциированных разбиений

Рассмотрим самоассоциированные разбиения. Из графика на рисунке 11 видно, что количество самоассоциированных разбиений растет с ростом числа n.

Опишем самоассоциированные разбиения при помощи следующей функции:

$$Associated(n) = ae^{b\sqrt{n}} + c. \qquad (22)$$

Для нахождения коэффициентов a, b и c воспользуемся методом наименьших квадратов `curve_fit`. `curve_fit` – это функция из библиотеки `scipy.optimize`, которая используется для аппроксимации кривой к набору данных. Ее основное назначение – нахождение параметров модельной функции, которая наилучшим образом соответствует заданным данным.

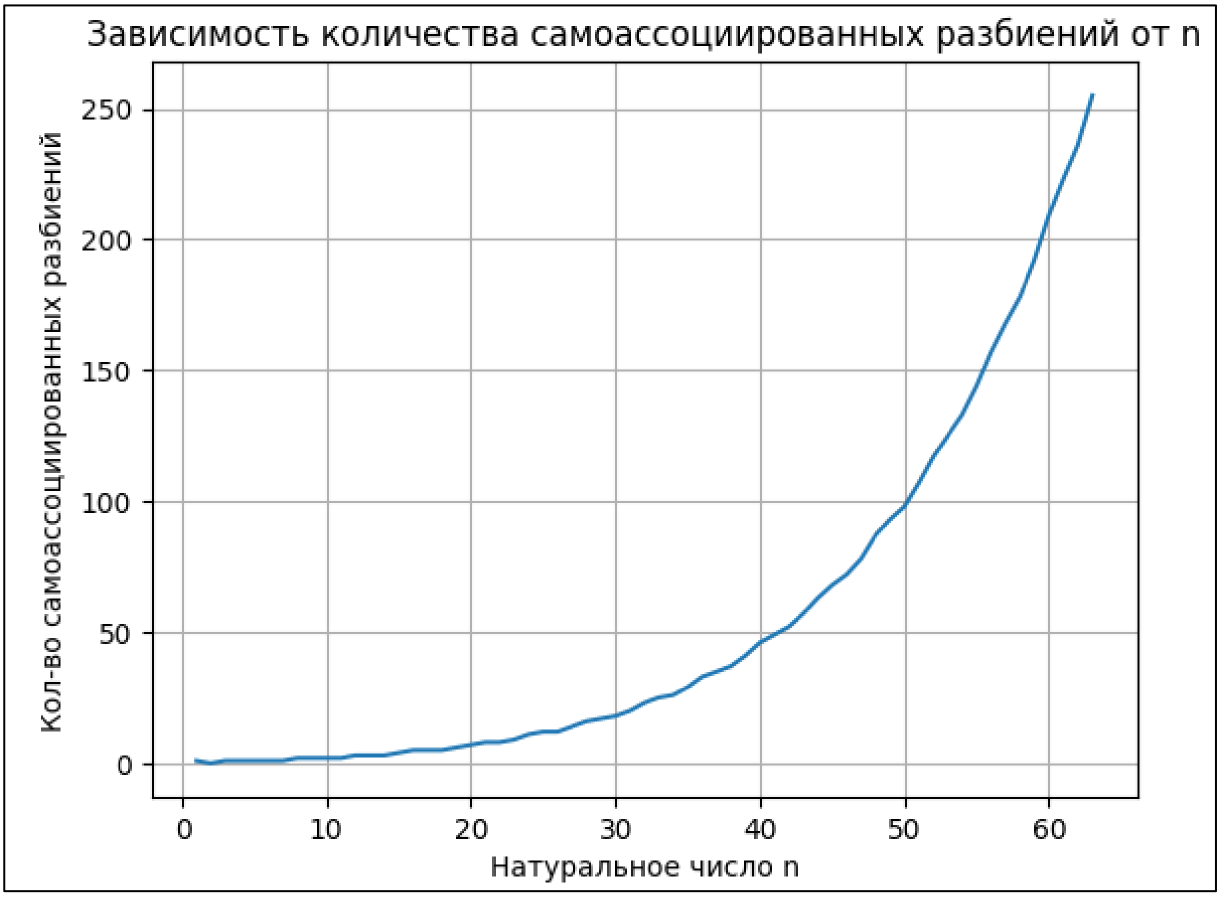

Рисунок 11 – График зависимости количества самоассоциированных разбиений от натурального числа

После использования `curve_fit` (листинг 9), были получены коэффициенты a, b и c, равные 0,07, 1,14и 0,3 соответственно. Результат аппроксимации приведен на рисунке 12.

Листинг 9 – Аппроксимация самоассоциированных разбиений

```
def mapping(x, a, b,c,d):
    return a * np.exp(b * np.sqrt(x))+c

args, covar = curve_fit(mapping, x_a, y_a)

x_fit = np.linspace(1, 64, 500)
y_fit = mapping(x_fit, *args)
```

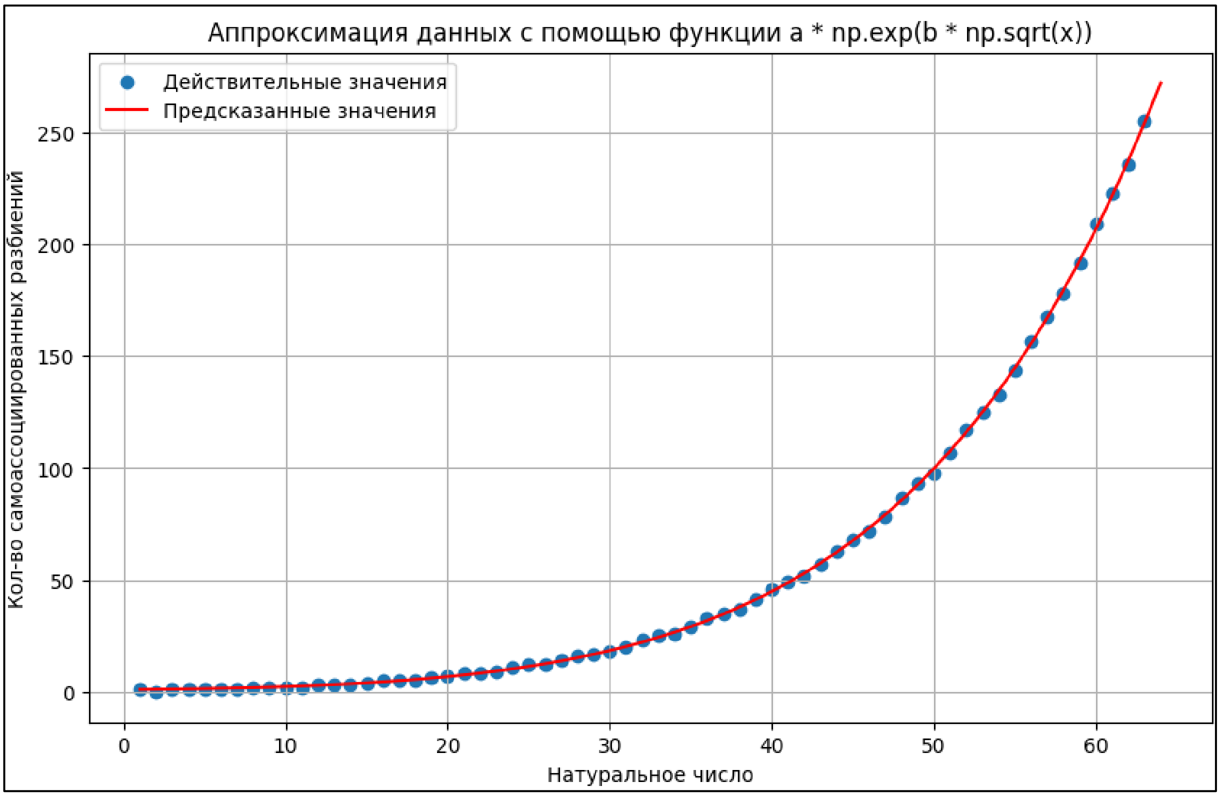

Рисунок 12 – График аппроксимации зависимости количества самоассоциированных разбиений от натурального числа

В качестве метрик оценки аппроксимации были получены среднее абсолютное отклонения (23), среднее квадратичное отклонение (24) и коэффициент детерминации (25).

Среднее абсолютное отклонение вычисляется по формуле:

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}, \tag{23}$$

где $\hat{y}_i$ – предсказанные значения, а y_i – фактические.

Среднее квадратичное отклонение вычисляется по формуле:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}, \tag{24}$$

где $\hat{y}_i$ – предсказанные значения, а y_i – фактические.

Коэффициент детерминации вычисляется по формуле:

$$R^2 = 1 - \frac{D[y|x]}{D[y]} = 1 - \frac{\sigma^2}{\sigma_y^2}, \tag{24}$$

где $D[y]$ – дисперсия случайной величины y, а $D[y|x]$ – условная дисперсия зависимой переменной.

В листинге 10 приведено вычисление метрик с использованием модуля `sklearn.metric`.

Листинг 10 – Получение метрик

```
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score

mae = mean_absolute_error(y_a, y_pred)
mse = mean_squared_error(y_a, y_pred)
r2 = r2_score(y_a, y_pred)
```

В таблице 8 приведены полученные метрики.

Таблица 8 – Метрики аппроксимации

MAE	**MSE**	$\boldsymbol{R^2}$
0,64	0,65	0,999

Если ориентироваться на метрики MAE и MSE, то присутствует некоторое отклонение прогнозируемых данных от реальных, связано это с тем, что данные рассмотрена на начальных диапазонах и для малых n имеет место быть некоторое отклонение. Метрика R^2 показывает, насколько хорошо модель соответствует данным и чем ближе она к единице, тем лучше модель объясняет вариабельность зависимой переменной.

ЗАКЛЮЧЕНИЕ

В рамках данной работы был разработано настольное приложение, которое позволяет получить следующие данные для разбиений натуральных чисел: частное степени разбиения, ранг характеристики разбиения, самоассоциированные разбиения.

Для достижения цели были спроектированы и разработаны алгоритмы для вычисления указанных данных. Было посчитано свыше 12 млн. разбиений и для каждого найдены необходимые вышеперечисленные показатели.

Предположено, что ранги натурального числа распределены нормально.

Для достижения цели были выполнены следующие задачи.

1. Изучена предметная область.
2. Разработан и реализован алгоритм нахождения характеристики и ранга характеристики вектора разбиения.
3. Разработан и реализован нахождения частного степени разбиения.
4. Разработан и реализован алгоритм нахождения самоассоциированного разбиения, нечетных разбиений и нечетных произведений для самоассоциированных разбиений.
5. Разработана программная система для вычисления данных, связанных с разбиениями.
6. Проведен анализ полученных результатов.

Дальнейших улучшений можно добиться за счет распараллеливания алгоритма и вычисления показателей для больших значений диапазонов натуральных чисел, что также позволит более точно аппроксимировать полученные результаты и установить смещение пика для ранга разбиений каждого натурального числа.

ЛИТЕРАТУРА

1. Эндрюс Д. Теория разбиений. – М.: Наука, 1982. – 256 с.
2. Фробениус Г. Теория характеров и представлений групп. – Харьков: Гос. науч.-техн. изд. Украины, 1937. – 211 с.
3. RSA Factoring Challenge. [Электронный ресурс] URL: https://web.archive.org/web/20130502202924/http://www.rsa.com/rsalabs/node.asp?id=2094 (дата обращения: 01.04.2024 г.).
4. Ваулин А.Е., Назаров М.С. Сведение задачи факторизации натурального числа к задаче разбиения числа на части. – СПИИРАН, 2015. – Выпуск 39. С. 157–176.
5. Flajolet P., Sedgewick R. Analytic Combinatorics. – Cambridge university press, 2009. – 810 с.
6. Andrews G., Eriksson K. Integer Partitions. – Cambridge university press, 2004. – 139 с.
7. About the unified modeling language specification version 2.5.1. [Электронный ресурс] URL: https://www.omg.org/spec/UML/2.5.1 (дата обращения: 11.04.2024 г.).
8. GAP System for Computational Discrete Algebra [Электронный ресурс] URL: https://www.gap-system.org./ (дата обращения 12.04.2024 г.).
9. Tkinter – Python interface to Tcl/Tk. [Электронный ресурс]. URL: https://docs.python.org/3/library/tkinter.html (дата обращения: 25.02.2024 г.).
10. Pandas. [Электронный ресурс] URL: https://pandas.pydata.org/ (дата обращения: 11.04.2024 г.).
11. Numpy. [Электронный ресурс] URL: https://numpy.org/ (дата обращения: 11.04.2024 г.).
12. Common format and MIME Type for Comma-Separated Values (CSV) Files [Электронный ресурс] URL: https://www.rfc-editor.org/rfc/rfc4180.txt (дата обращения: 28.04.2024 г.).
13. Алеев Р.Ж., Каргаполов А.В., Соколов В.В. Ранги групп центральных единиц целочисленных групповых колец знакопеременных

групп. – Фундаментальная и прикладная математика, 2008. – Т. 14, № 7. – С. 15 – 21.

14. Timeit – Measure execution time of small snippets. [Электронный ресурс] URL: https://docs.python.org/3/library/timeit.html (дата обращения: 28.04.2024 г.).

15. Matplotlib 3.8.4 documentation. [Электронный ресурс] URL: https://matplotlib.org/stable/ (дата обращения: 28.04.2024 г.).

16. Пероцкая В.Н, Градусов Д.А. Основы тестирования программного обеспечения. – Владимир: Изд-во ВлГУ, 2017. – 100 с.

17. Scipy.optimize.curve_fit. [Электронный ресурс] URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html (дата обращения: 28.04.2024 г.).

18. Math – mathematical function. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/model_evaluation.html (дата обращения: 28.04.2024 г.).

19. Трофимова Е.А., Кисляк Н.В., Гилев Д.В. Теория вероятностей и математическая статистика. – Екатеренбург: Изд-во Урал. ун-та, 2018. – 160 с.

20. Metrics and scoring: quantifying the quality of predictions. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/model_evaluation.html (дата обращения: 28.04.2024 г.).

ПРИЛОЖЕНИЯ

Приложение А. Основной текст программы

В таблицах 1–3 приведены спецификации вариантов использования приложения.

Таблица 1 – Спецификация ВИ «Получить ранги разбиений»

Прецедент: получить ранг разбиения
ID: 2
Краткое описание Получение результатов по рангу характеристики
Основной актер Пользователь
Второстепенные актеры: Нет
Предусловие Запущено приложение, и пользователь находится на «Разбиения одного числа»
Основной поток 1. Пользователь вводит натуральное число 2. Пользователь выбирает в поле выбора «Получить ранг характеристики» 3. Пользователь нажимает кнопку «Показать результат» 4. Система отображает таблицу и график
Постусловие: показана таблица и график
Альтернативные потоки I. Ошибка ввода 1. Приложение очищает поле ввода 2. Приложение выводит сообщение об ошибке ввода II. Ошибка выбора категории 1. Приложение выводит сообщение об отсутствии выбора категории

Таблица 2 – Спецификация ВИ «Получить самоассоциированные разбиения»

Прецедент: получить самоассоциированные разбиения
ID: 3
Краткое описание Получение самоассоциированных разбиений
Основной актер Пользователь
Второстепенные актеры: Нет
Предусловие Запущено приложение, и пользователь находится на «Самоассоциированные разбиения»
Основной поток 1. Пользователь вводит натуральное число – начало диапазона 2. Пользователь вводит натуральное число – конец диапазона 3. Пользователь нажимает кнопку «Показать результат» 4. Система отображает таблицу и график

Окончание таблицы 2 приложения А

Постусловие: показана таблица и график
Альтернативные потоки I. Ошибка ввода 1. Приложение очищает поле ввода 2. Приложение выводит сообщение об ошибке ввода

Таблица 3 – Спецификация ВИ «Сохранить таблицу»

Прецедент: сохранить таблицу
ID: 4
Краткое описание Сохранить таблицу с показателями разбиений
Основной актер Пользователь
Второстепенные актеры: Нет
Предусловие Запущено приложение, и пользователь получил на экране таблицу
Основной поток 1. Пользователь нажимает на пункт меню «Файл» 2. Пользователь выбирает «Сохранить таблицу» 3. Система выдает всплывающие окно 4. Пользователь вводит название файла 5. Пользователь выбираем место на диске для сохранения таблицы в файл 6. Пользователь нажимает на кнопку сохранить
Постусловие: таблица сохранена на диск
Альтернативные потоки I. Ошибка в названии файла 1. Приложение очищает поле ввода 2. Приложение выводит сообщение об ошибке названия файла II. Файл с таким названием существует 1. Система выводит предупреждение 2. Если пользователь соглашается, то файл перезаписывается 3. Если пользователь не соглашается, то система предлагает переименовать файл

Приложение Б. Графики смещения пиков

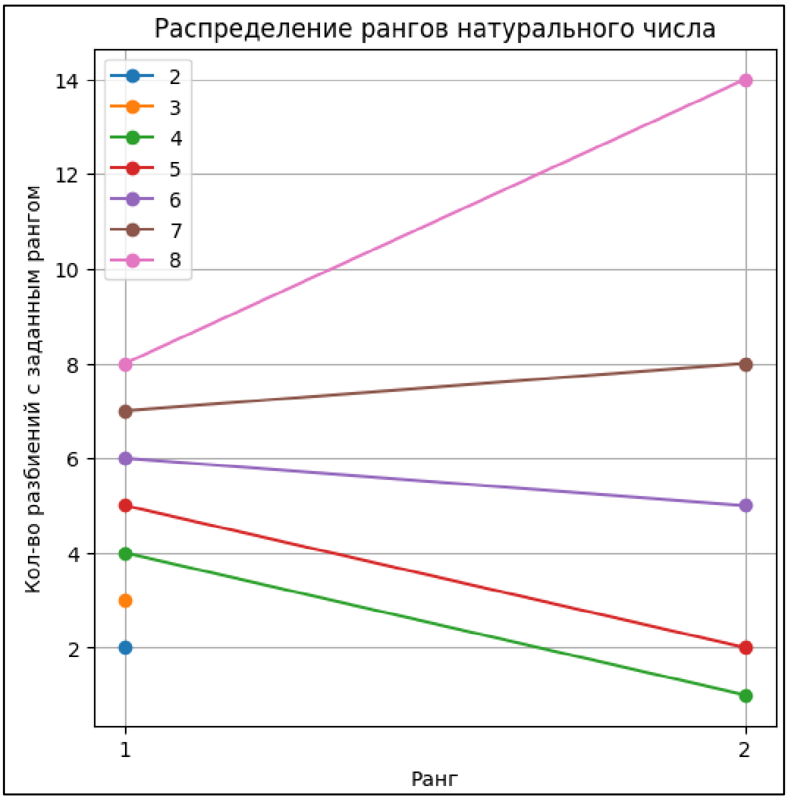

Рисунок 1 – График распределения разбиений для $n \in [2; 8]$

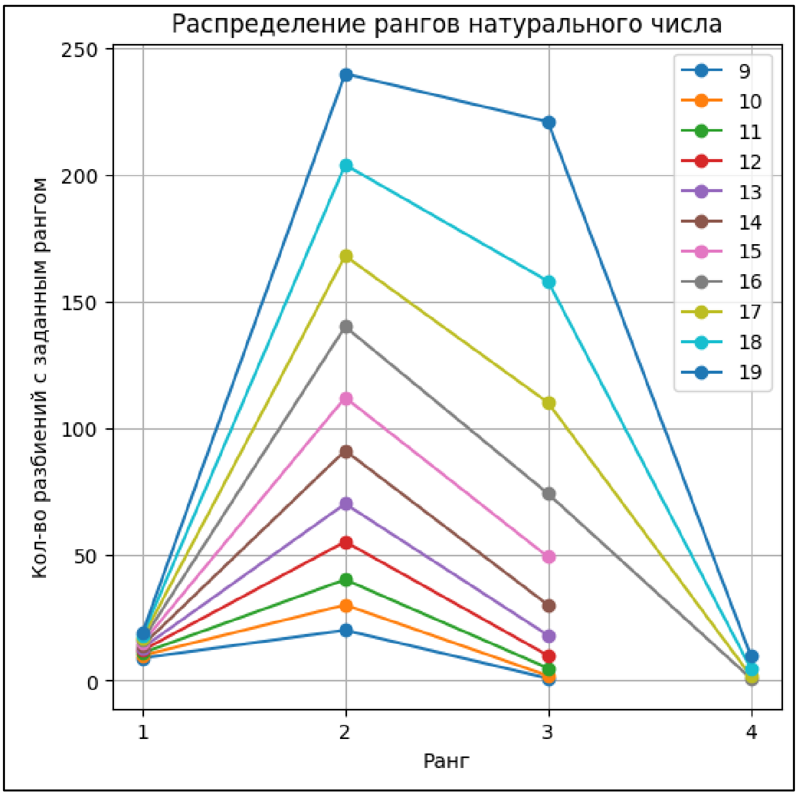

Рисунок 2 – График распределения разбиений для $n \in [9; 19]$

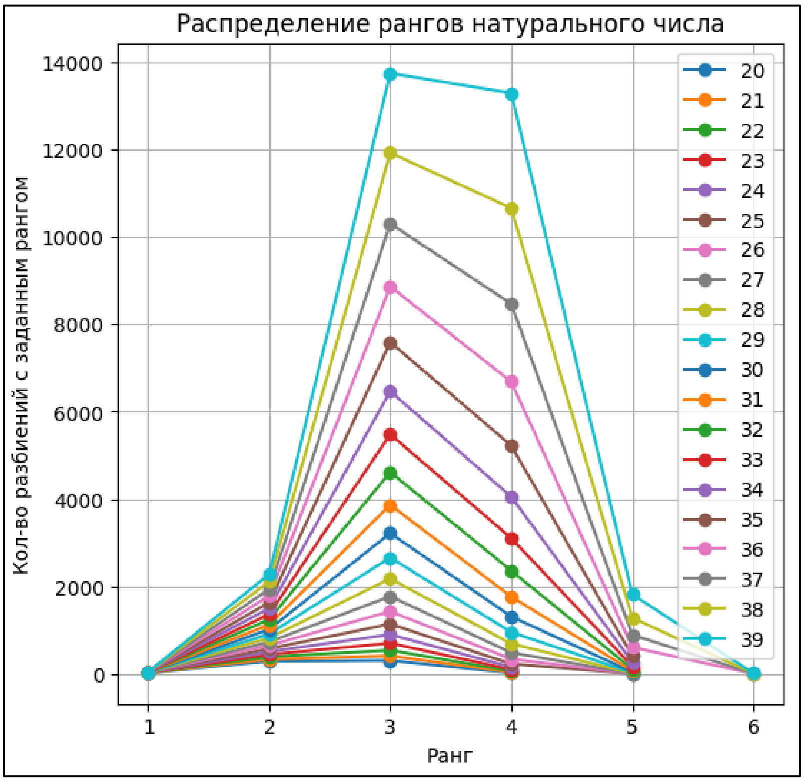

Рисунок 3 – График распределения разбиений для $n \in [20; 39]$

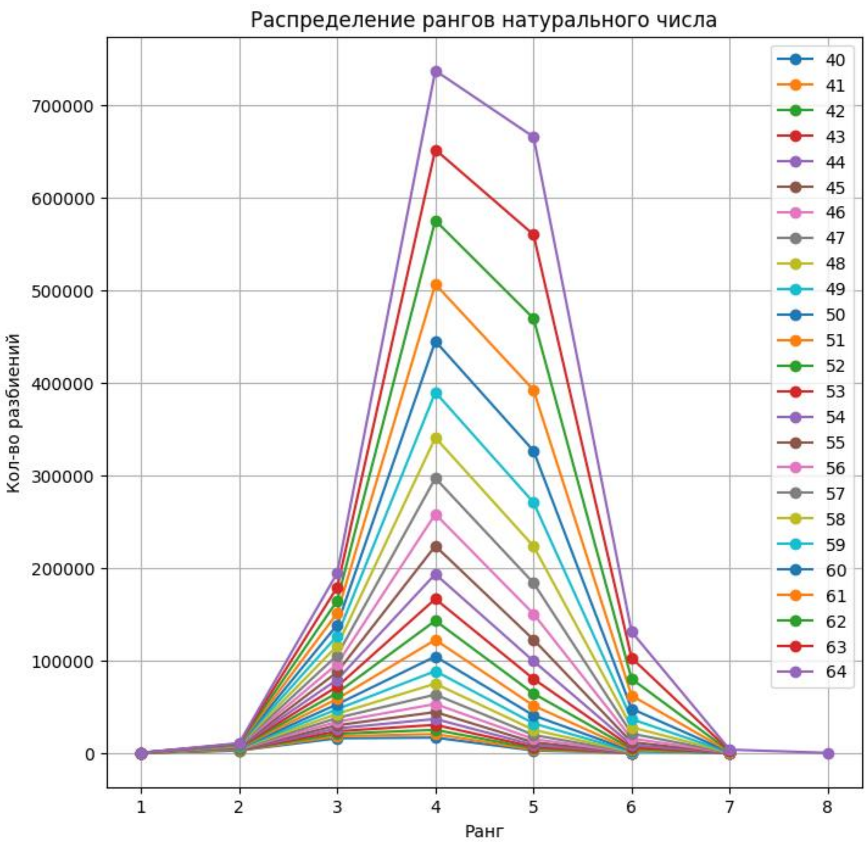

Рисунок 4 – График распределения разбиений для $n \in [40; 64]$

Приложение В. Таблица показателей разбиения

Таблица 4 – Показатели разбиения

Натуральное число n	Кол-во слагаемых	Разбиение	Вектор разбиения	Ранг	Характеристика вектора разбиения	Частное степени разбиения	Самоассоциированные разбиения
1	1	[1]	[1]	1	[[0], [0]]	1	1
2	2	[1, 1]	[1, 2]	1	[[1], [0]]	1	0
2	1	[2]	[0, 3]	1	[[0], [1]]	1	0
3	3	[1, 1, 1]	[1, 2, 3]	1	[[2], [0]]	1	0
3	2	[1, 2]	[0, 2, 4]	1	[[1], [1]]	2	1
3	1	[3]	[0, 1, 5]	1	[[0], [2]]	1	0
4	4	[1, 1, 1, 1]	[1, 2, 3, 4]	1	[[3], [0]]	1	0
4	3	[1, 1, 2]	[0, 2, 3, 5]	1	[[2], [1]]	3	0
4	2	[2, 2]	[0, 1, 4, 5]	2	[[0, 1], [0, 1]]	2	1
4	2	[1, 3]	[0, 1, 3, 6]	1	[[1], [2]]	3	0
4	1	[4]	[0, 1, 2, 7]	1	[[0], [3]]	1	0
5	5	[1, 1, 1, 1, 1]	[1, 2, 3, 4, 5]	1	[[4], [0]]	1	0
5	4	[1, 1, 1, 2]	[0, 2, 3, 4, 6]	1	[[3], [1]]	4	0
5	3	[1, 2, 2]	[0, 1, 3, 5, 6]	2	[[0, 2], [0, 1]]	5	0
5	3	[1, 1, 3]	[0, 1, 3, 4, 7]	1	[[2], [2]]	6	1
5	2	[2, 3]	[0, 1, 2, 5, 7]	2	[[0, 1], [0, 2]]	5	0
5	2	[1, 4]	[0, 1, 2, 4, 8]	1	[[1], [3]]	4	0
5	1	[5]	[0, 1, 2, 3, 9]	1	[[0], [4]]	1	0
6	6	[1, 1, 1, 1, 1, 1]	[1, 2, 3, 4, 5, 6]	1	[[5], [0]]	1	0
6	5	[1, 1, 1, 1, 2]	[0, 2, 3, 4, 5, 7]	1	[[4], [1]]	5	0
6	4	[1, 1, 2, 2]	[0, 1, 3, 4, 6, 7]	2	[[0, 3], [0, 1]]	9	0
6	3	[2, 2, 2]	[0, 1, 2, 5, 6, 7]	2	[[1, 2], [0, 1]]	5	0