**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА**

ФедеральноеГосударственное Бюджетное Образовательное Учреждение

Высшего Образования

**«Петербургский Государственный Университет Путей Сообщения**

**Императора Александра I»**

**(ФГБОУ ВО ПГУПС)**

Факультет «Автоматизация и интеллектуальные технологии»

Кафедра «Информационные и вычислительные системы»

**ЛАБОРАТОРНАЯ РАБОТА №3**

по дисциплине

**«Структуры и алгоритмы обработки данных (на C++)»**

ТЕМА РАБОТЫ

**«Хеш-функции и хеш-таблицы»**

Выполнил:

|  |  |  |  |
| --- | --- | --- | --- |
| Обучающийся группы ИВБ-411 | 25.10.2025  подпись, дата | | Маер Константин Дмитриевич  И. О. Фамилия |
| Исправить замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | доцент, к.и.н. A. B. Забродин\_\_\_\_\_\_\_  учёное звание, И.О. Фамилия |
| Защита: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Зачтено/не зачтено | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | | доцент, к.и.н. A. B. Забродин \_\_\_\_\_  учёное звание, И.О. Фамилия |

**Санкт-Петербург**

**2025г.**

**Цель работы:** Реализовать хеш-таблицу с использованием метода цепочек или открытой адресации для разрешения коллизий. Реализовать стандартные операции: добавление, удаление и поиск элементов.

**Хеш-функции и хеш-таблицы**

**Порядок выполнения работы**

1. **Хеш-функция GetHash1**

Простая позиционно-взвешенная сумма.

Алгоритм: проходит по каждому элементу входного массива/строки, умножает значение элемента на его индекс и суммирует результат. Итоговая сумма берётся по модулю 1073741824 для ограничения размера числа.

Особенности: учитывает позицию символа в строке, но простое суммирование может давать больше коллизий на определённых паттернах данных.

1. **Хеш-функция GetHash2**

модифицированная версия алгоритма DJB2.

Алгоритм: начинается с фиксированного начального значения 5381. Для каждого элемента вычисляется: hash = (hash << 5) + hash + vec[i] \* i, что эквивалентно hash = hash \* 33 + vec[i]\*i.

Особенности: смещает старые значения влево на 5 бит и добавляет новое значение, умноженное на индекс. Хорошо распределяет значения для текстовых данных и снижает количество коллизий по сравнению с GetHash1.

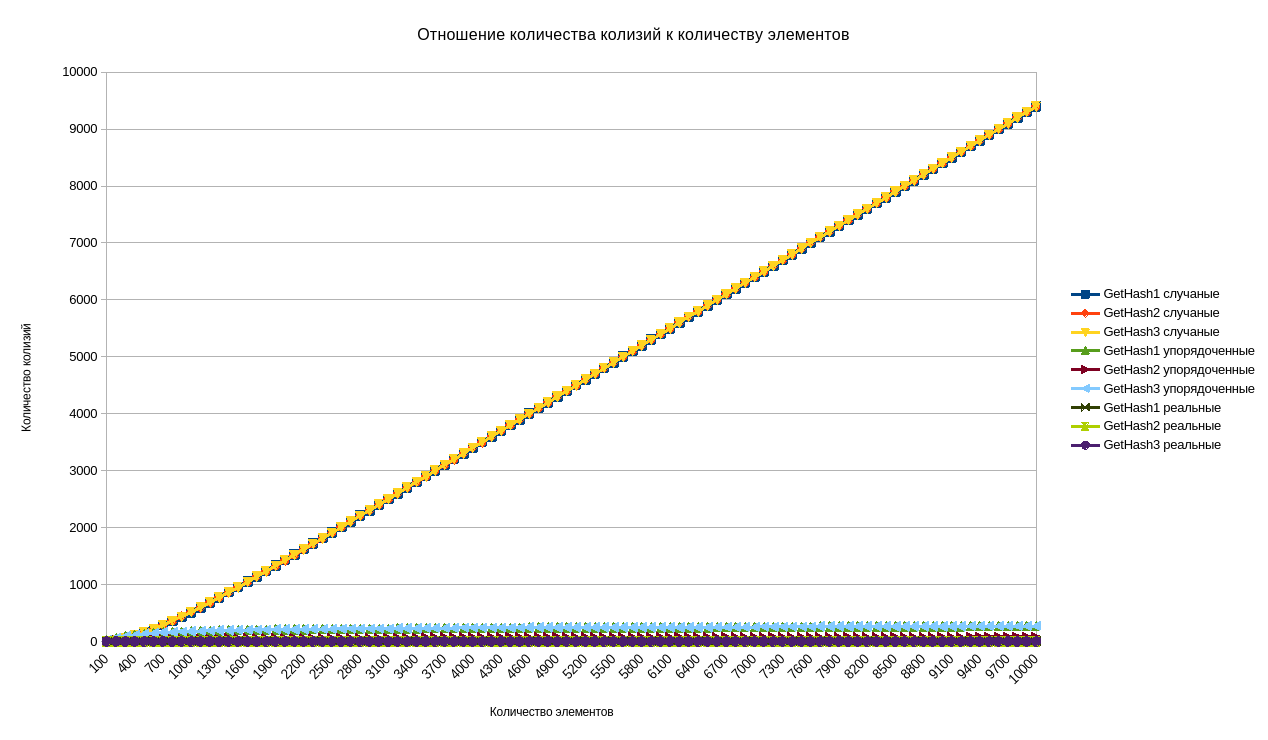
1. **Хеш-функция GetHash3**

мультипликативная хеш-функция.

Алгоритм: суммирует элементы массива, умноженные на их индексы, затем результат умножается на константу 2654435761 (часто используется в мультипликативных хешах, основанная на числе Фибоначчи). Итог берётся по модулю hash\_max через NORMALIZE.

Особенности: использование константы 2654435761 улучшает равномерность распределения хешей, особенно для числовых и упорядоченных данных, минимизируя кластеры и коллизии.

1. **Анализ результатов тестирования хеш-функций**



1) Общие наблюдения

Случайные данные

GetHash1: количество коллизий растет от 8 (100 строк) до 9400 (10000 строк).

GetHash2: стабильное количество коллизий 5–83.

GetHash3: количество коллизий 7–272.

Упорядоченные данные

GetHash1: резкий рост коллизий от 28 до 272.

GetHash2: минимальные значения коллизий 6–83.

GetHash3: умеренный рост коллизий 28–272.

Реальные данные

Все функции: 0 коллизий.

1. **Выводы**

В ходе работы была реализована хеш-таблица с разрешением коллизий методом цепочек и исследованы три собственные хеш-функции для строковых данных: простая позиционно-взвешенная сумма (GetHash1), модифицированная функция DJB2 (GetHash2) и мультипликативная функция (GetHash3).

Проведено тестирование на трех типах данных: случайные, упорядоченные и реальные строки. Анализ показал, что:

1. GetHash1 эффективна на небольших случайных данных, но плохо справляется с упорядоченными и большими наборами, демонстрируя значительное увеличение коллизий.

2. GetHash2 (DJB-подобная) показала наилучшую стабильность и равномерность распределения хешей, особенно для упорядоченных и больших наборов данных.

3. GetHash3 (мультипликативная) хорошо распределяет случайные и большие данные, снижая количество коллизий по сравнению с простой суммой, но чуть уступает GetHash2 на упорядоченных данных.

4. Для небольших и разнообразных реальных данных разница между функциями практически отсутствует — коллизий не выявлено.

Рекомендации: для практического использования в хеш-таблицах предпочтительнее применять функции типа GetHash2 или GetHash3, которые демонстрируют устойчивую эффективность и минимизируют коллизии. Простые позиционно-взвешенные хеши могут использоваться только для небольших и случайных наборов данных.

В целом работа позволила изучить влияние выбора хеш-функции на эффективность хеш-таблицы и важность правильного распределения хешей для минимизации коллизий.