­­Бюджетное учреждение высшего образования   
Ханты-Мансийского автономного округа   
«Сургутский государственный университет»

Политехнический институт

Кафедра автоматики и компьютерных систем

**Отчет**

по лабораторной работе № 7

по дисциплине «Алгоритмы и структуры данных»

Выполнил: студент группы 609-21,

Шумилов И.Д.

Принял: старший преподаватель кафедры АиКС

Назаров Е.В.

2024 г.

**Цель работы:** изучить принципы построения хеш-функций, обладающих равномерным распределением, исследовать статистические свойства хеш-функций, закрепить навыки структурного программирования.

**Общее задание на работу:**

1. Разработать и реализовать функцию, осуществляющую хеширование данных (тип данных определяется вариантом).
2. Разработать и реализовать функцию-генератор, осуществляющую формирование значений ключей в соответствии с заданным типом данных. Генерируемые ключи должны быть уникальны.
3. Исследовать статистические свойства разработанной хеш-функции при заданных размерах хеш-таблицы и количестве ключей.
4. Составить отчет, в котором привести листинг хеш-функции, гистограммы распределений индексов, формируемых хеш-функцией (для двух значений размера хеш-таблицы) и выводы по работе (дать оценку зависимости от размера таблицы и от природы исходных данных – если таковые имеются; оценить качество разработанной хеш-функции).

**Индивидуальное задание на работу:**

Таблица 1 – Вариант 7 индивидуального ндивидуального задания

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип данных | Размеры хэш-таблицы и количество ключей | Примечание |
| 7 | Struct Person {  char\* surname;  } | M1 = 512  M2 = 511  K = 2000 | В качестве значений Surname использовать фамилии, приведенные в приложении Б. |

**Листинги функций**

Листинг 1. Функция генерации ключа.

std::set<int> usedKeys;

int HashTable::generateKey(Person value) {

int sum = 0;

int len = strlen(value.surname);

for (int i = 0; i < len; i++) {

sum += ((int)value.surname[i] \* (len - i)) % CAPACITY;

}

while (!usedKeys.insert(sum).second)

sum++;

return sum;

}

Листинг 2. Функция хэширования.

int HashTable::hashFunction(int key) {

return (key % CAPACITY + CAPACITY) % CAPACITY;

}

**Распределение индексов по хэш-таблице.**

Рис. 1 – Распределение индексов по таблице размера 512

При размере в 512 максимальное количество элементов на один индекс было 8 штук, а среднее значение по таблице было 3.53 элементов.

Рис. 2 – Распределение индексов по таблице размера 511

При размере в 511 максимальное количество элементов на один индекс было 7 штук, а среднее значение по таблице было 3.55 элементов.

Распределение вышло достаточно равномерным, с учетом того что для решения колизий используется метод цепочек, то среднее время поиска будет несильно отличаться от теоретических O(1).

Разработанная хэш-функция в основном проста и обеспечивает равномерное распределение ключей по массиву, что является важным качеством для хорошей хэш-функции. Однако ее эффективность может зависеть от размера таблицы и генерации ключей.

**Вывод:** были изучены принципы построения хеш-функций, обладающих равномерным распределением, исследованы статистические свойства хеш-функций, закреплены навыки структурного программирования.