Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий и систем связи

Лабораторная работа №2

Вариант №2

Выполнил(и:)

Алексеев Т.

Бабаев Р.

Белисов Г.

Проверил

Мусаев А.А.

Санкт-Петербург,

2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной работы являлось знакомство с различными методами хэширования входных данных: умножением, делением, сложением, CRC-32, MD-5.

Для достижения данной цели необходимо было выполнить следующие задания:

1. Задание 1: написание метода для хэширования входных данных с помощью метода умножения.
2. Задание 2: написание метода для хэширования входных данных с помощью метода CRC-32.

Решения данных задач будут находиться на GitHub по ссылке: https://github.com/NorthPole0499/Algoritms_task_7

1. ЗАДАНИЕ 1

В задании было необходимо реализовать хэширование умножением. Для этого в начале случайно выбирается ключ от 1 до 50 и константа в интервале от 0 до 1.

Для самого хэширования была реализована функция hash_mult(table, key, const, text). В переменную n записывается произведение константы и ключа. Далее для каждого символа во входных данных высчитывается hash_key с помощью n и длины словаря table. К конечному итогу, в словарь записывается hash_key, а к переменной result он просто прибавляется. В переменной result содержится получившийся хэш входных данных.

Также, для проверки результата была реализована функция dehash_multi(table, hash_text). Учитывая, что у нас имеется заполненный словарь и хэш входных данных, мы можем простым перебором без труда восстановить исходные данные.

```
Введите текст, который нужно хэшировать: Кот
Ключ: 4 Константа: 0.7120620365471113
Хэшированный текст (умножение): 025
Исходный текст (умножение): Кот

Process finished with exit code 0
```

Приложение 1 – Пример вывода программы для Задания 1

2. ЗАДАНИЕ 2

Вторая задача заключается в том, что необходимо реализовать метод CRC-32.

Для этого были написаны три функции: основная encode_data(data, key) и вспомогательные division(dividend, divisor) и xor(a, b).

Функция xor применяет булеву функцию исключающее «или» для всех соответствующих элементов строк a и b. Если элементы совпадают, записываем 0. Иначе 1.

Функция division выполняет «деление в столбик» битовой строки входных данных и ключа. Функция работает с помощью двух важных переменных. Переменной curr, которая имитирует пограничный индекс при применении xor, и переменной segment, которая является срезом, с которой и происходит сравнение ключа (вызов функции xor). Таким образом, segment после выполнения функции становится закодированным суффиксом исходной битовой строки.

А функция encode_data собирает вышеописанные функции в одну. Сначала объявляется переменная appended_data, которая содержит в себе исходные битовые данные с приписанным необходимым количеством нулей. Далее вызывается функция division, которая возвращает необходимый суффикс. После этого исходные данные и суффикс склеиваются и возвращаются.

```
Введите текст, который нужно хэшировать: Кот
Хэшированный текст (CRC-32): 10000011010100001111101000100001011010000011101100000111101101l0

Process finished with exit code 0
```

Приложение 2 – Пример вывода программы для Задания 2

ЗАКЛЮЧЕНИЕ

Таким образом, в ходе выполнения данной лабораторной работы были достигнуты все поставленные цели. Произошло знакомство с различными методами хэширования входных данных, а именно, умножением и CRC-32, и применение их на практике.

СПИСОК ЛИТЕРАТУРЫ

1) Метод умножения – Сутдопедия // URL: https://studopedia.ru/5_159617_metod-umnozheniya.html (дата обращения: 05.03.2023)