Учреждение образования

«Белорусский государственный технологический университет»

**Кафедра информационных систем и технологий**

**«Отчёт по лабораторной работе 11»**

“ ИССЛЕДОВАНИЕ КРИПТОГРАФИЧЕСКИХ

ХЕШ-ФУНКЦИЙ”

**Выполнил:** студент 3 курса

4 группы специальности ПОИТ

Кравцова Диана Вячеславовна

**Проверил:** преподаватель

Блинова Евгения Александровна

Минск 2021

## 1. Описание приложения

Приложение написано на языке программирования C# и использует алгоритм MD5 из встроенной библиотеки. Приложение обрабатывает входное сообщение, длина которого определена спецификацией

## 2. Методика выполнения поставленных задач

На вход алгоритма поступает входной поток данных, хеш которого необходимо найти. Длина сообщения измеряется в битах и может быть любой (в том числе нулевой). Запишем длину сообщения в L. Это число целое и неотрицательное. Кратность каким-либо числам необязательна. После поступления данных идёт процесс подготовки потока к вычислениям.

**Шаг 1. Выравнивание потока**

Сначала к концу потока дописывают единичный бит. Затем добавляют некоторое число нулевых бит такое, чтобы новая длина потока L' стала сравнима с 448 по модулю 512. Выравнивание происходит в любом случае, даже если длина исходного потока уже сравнима с 448.

**Шаг 2. Добавление длины сообщения**

В конец сообщения дописывают 64-битное представление длины данных (количество бит в сообщении) до выравнивания. Сначала записывают младшие 4 байта, затем старшие. Если длина превосходит 2^64-1, то дописывают только младшие биты. После этого длина потока станет кратной 512. Вычисления будут основываться на представлении этого потока данных в виде массива слов по 512 бит.

**Шаг 3. Инициализация буфера**

Для вычислений инициализируются четыре переменные размером по 32 бита, начальные значения которых задаются шестнадцатеричными числами (порядок байтов little-endian):

А = 01 23 45 67; // 67452301h

В = 89 AB CD EF; // EFCDAB89h

С = FE DC BA 98; // 98BADCFEh

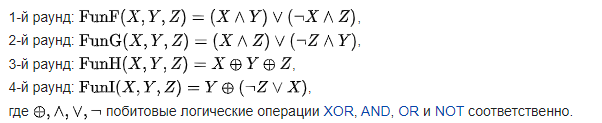
D = 76 54 32 10. // 10325476h

В этих переменных будут храниться результаты промежуточных вычислений. Начальное состояние ABCD называется инициализирующим вектором.

**Шаг 4. Вычисление в цикле**

Определим функции и константы, которые понадобятся нам для вычислений.

Для каждого раунда потребуется своя функция. Введём функции от трёх параметров — слов, результатом также будет слово:



Определим таблицу констант T[1…64]; T[i] = int(2^32\*|sin(n)|)

Каждый 512-битный блок проходит 4 этапа вычислений по 16 раундов. Для этого блок представляется в виде массива X из 16 слов по 32 бита. Все раунды однотипны и имеют вид: [abcd k s i], определяемый как



, где k — номер 32-битного слова из текущего 512-битного блока сообщения, <<< s — циклический сдвиг влево на s бит полученного 32-битного аргумента. Число s задается отдельно для каждого раунда.

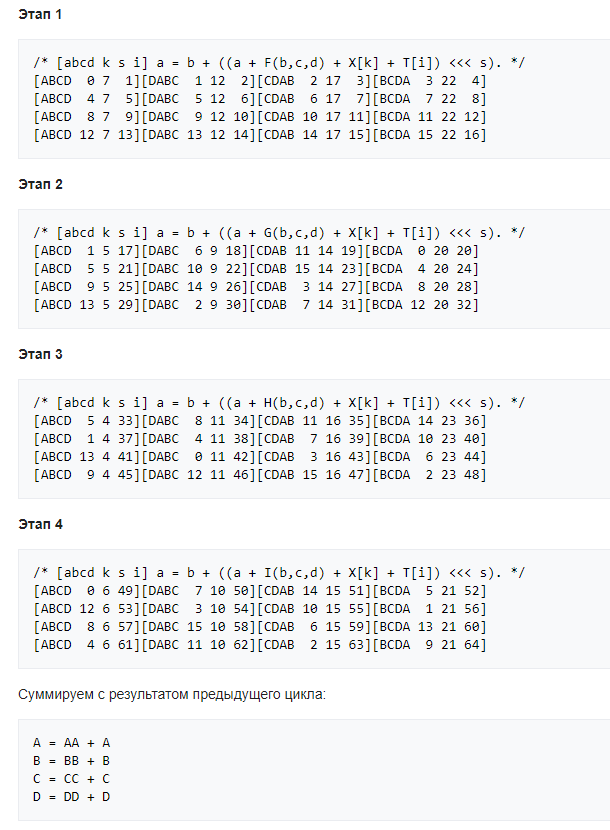
Заносим в блок данных элемент n из массива 512-битных блоков. Сохраняются значения A, B, C и D, оставшиеся после операций над предыдущими блоками (или их начальные значения, если блок первый).

AA = A

BB = B

CC = C

DD = D

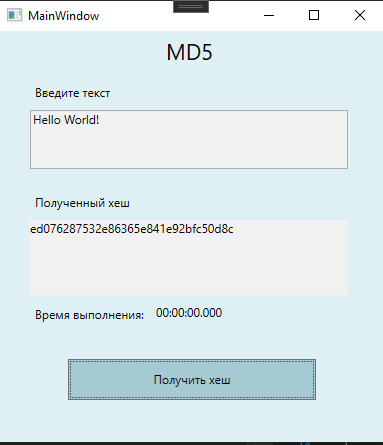


После окончания цикла необходимо проверить, есть ли ещё блоки для вычислений. Если да, то переходим к следующему элементу массива (n + 1) и повторяем цикл.

**Шаг 5. Результат вычислений**

Результат вычислений находится в буфере ABCD, это и есть хеш. Если выводить побайтово, начиная с младшего байта A и заканчивая старшим байтом D, то мы получим MD5-хеш. 1, 0, 15, 34, 17, 18…

Результат работы приложения:



## Вывод

В ходе лабораторной работы были изучены алгоритмы хеширования и приобретены практические навыки их реализации и использования в криптографии. Также была оценена скорость вычисления кодов хеш-функции MD5.