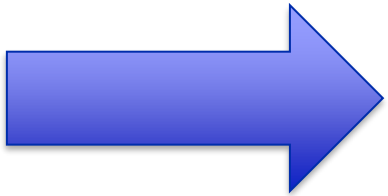
УНИВЕРСИТЕТ ИТМО

# Лекция 2
# Хэш-функции

**Хэширование** – метод адресации данных для быстрого поиска по ключевым выражениям.

Различные алгоритмы хеширования позволяют записывать текст в зашифрованном виде для повышения безопасности, экономии памяти или по другим частным причинам.

Алгоритм хэширования



8A6BEAB4358AAEE38CD41B1C4EF84BA9

Пример методов хэширования: деление, умножение, сложение (аддитивный метод).

**Метод деления** – используется остаток от деления ключа (К) на размер массива (М).

Длинна массива: 11
Ключи: 11, 122, 123, 154, 15, 217,…
11 mod 11=0
122 mod 11=1
154 mod 11=0
15 mod 11=4
217 mod 11=8

(mod – остаток от деления, записанный как целое число)

Ситуация, когда разным ключам соответствует один индекс называется **коллизией**.

**Метод умножения** – используется выражение M*((K*C)mod1), где С – константа из интервала [0..1]

Длинна массива: 10
Ключ:25
С: 0,1

10*((25*0,1) mod 1)=5

**Метод сложения (аддитивный метод)** – используется для символьных ключей. Заключается в последовательном суммировании цифровых кодов символов с последующим делением на M (как правило, M=256).

Длинна массива: 10
Ключ: ABC
A=1, B=2, C=3
(1+2+3) mod 10 = 5

Совокупность различных базовых алгоритмов привела к появлению нескольких популярных хэш-функций: CRC-32, MD5, SHA-1, SHA-2, ГОСТ Р 34.11-2012 «Стрибог».

**CRC (Cyclic Redundancy Check, циклическая проверка излишков).**

Применяется для защиты данных и обнаружении ошибок в потоке информации. Стал известен благодаря следующим качествам: легко и быстро обнаруживает ошибки, требует минимальные расходы, прост в эксплуатировании.

Для определенной последовательности битов составляется полином вида:

$$\sum_{n=0}^{N-1} a_n x^n$$

Последовательность: 11010
Полином: $P(X)=1*x^4+1*x^3+0*x^2+1*x^1+0*x^0$

Значение CRC (R(x)) получается по следующей формуле:

$$R(x) = P(x) \cdot x^N \mod G(x)$$

P(x) – многочлен, представляющий из себя входные данные
G(x) – порождающий многочлен
N – степень порождающего многочлена

Популярные G(x):

| Название | Представления: нормальное / реверсированное / реверсированное от обратного |
|---|---|
| CRC-1 | 0x1 / 0x1 / 0x1 |
| CRC-7 | 0x09 / 0x48 / 0x44 |
| CRC-8-Dallas/Maxim | 0x31 / 0x8C / 0x98 |
| CRC-8 | 0xD5 / 0xAB / 0xEA |
| CRC-12 | 0x80F / 0xF01 / 0xC07 |
| CRC-16-IBM | 0x8005 / 0xA001 / 0xC002 |
| CRC-16-CCITT | 0x1021 / 0x8408 / 0x8810 |
| CRC-16-T10-DIF | 0x8BB7 / 0xEDD1 / 0xC5DB |
| CRC-16-DNP | 0x3D65 / 0xA6BC / 0x9EB2 |
| CRC-32-IEEE 802.3 | 0x04C11DB7 / 0xEDB88320 / 0x82608EDB |
| CRC-32C (Castagnoli) | 0x1EDC6F41 / 0x82F63B78 / 0x8F6E37A0 |
| CRC-32K (Koopman) | 0x741B8CD7 / 0xEB31D82E / 0xBA0DC66B |
| CRC-32Q | 0x814141AB / 0xD5828281 / 0xC0A0A0D5 |
| CRC-64-ISO | 0x000000000000001B / 0xD800000000000000 / 0x8000000000000000D |

**MD5** (*Message Digest 5*).

Базируется на 128-битном (16-байтном) фундаменте. Применяется для хранения паролей, создания уникальных криптографических ключей и ЭЦП. Используется для аудита подлинности и целостности документов в ПК. Недостаток – сравнительно легкое нахождение коллизий.

Алгоритм состоит из 5 этапов:  выравнивание потока, добавление длины, инициализация буфера, вычисление,  представление результата.

**Этап № 1. Выравнивание потока.**

К изначальному потоку добавляется в начале «1», а затем добавляются «0» до тех пор, пока длина сообщения не будет сравнима с 448 по модулю 512. Т.е. **длина = n*512+448**, где n – натуральное число, а длина ≈ длине блока исходного.

**Этап № 2. Добавление длины.**

В конец сообщения дописывается 64-битное представление данных (64-битное представление длины исходного сообщения).

**Этап № 3. Инициализация буфера.**

Добавляются четыре 32-битные переменные

A: 01 23 45 67
B: 89 ab cd ef
C: fe dc ba 98
D: 76 54 32 10

**Этап № 4. Вычисление.**

1. Записываются четыре вспомогательные функции со следующей логикой:

F(X,Y,Z) = XY v not(X) Z
G(X,Y,Z) = XZ v Y not(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z) = Y xor (X v not(Z))

2. Реализуется шумовая составляющая:

$$T[i]=4,294,967,296*abs(sin(i))$$

3. Каждый 16-битный блок X копируется в отдельные массивы и производится замена:

$$AA = A$$
$$BB = B$$
$$CC = C$$
$$DD = D$$

4. Реализуются четыре этапа четырех преобразований:

$$A = B + ((A + F(B,C,D) + X[k] + T[i])$$

A, B, C, D — регистры, F(B,C,D) — одна из логических функций, X[k] — k-тый элемент 16-битного блока, T[i] — i-тый элемент таблицы «белого шума»

2. Суммируются результаты предыдущего цикла (результаты вычислений):

```
A = A + AA
B = B + BB
C = C + CC
D = D + DD
```

**Этап № 5. Представление данных.**

Выводится зашифрованное сообщение.