

Хэш-функции (Hash functions)

- **Хэш-функция (Hash function)** – это функция, преобразующая значения ключа (например: строки, числа, файла) в целое число
- Значение, возвращаемое хэш-функцией, называется **ХЭШ-КОДОМ** (hash code), контрольной суммой (hash sum) или хэшем (hash)

```
#define HASH_MUL 31
#define HASH_SIZE 128

// Хэш-функция для строк
unsigned int hash(char *s)
{
    unsigned int h = 0;
    char *p;

    for (p = s; *p != '\0'; p++)
        h = h * HASH_MUL + (unsigned int)*p;
    return h % HASH_SIZE;
}
```

$$T_{hash} = O(|s|)$$

Хэш-функции (Hash functions)

- **Хэш-функция (Hash function)** – это функция, преобразующая значения ключа (например: строки, числа, файла) в целое число
- Значение, возвращаемое хэш-функцией, называется **хэш-кодом (hash code)**, контрольной суммой (hash sum) или хэшем (hash)

```
#define HASH_MUL 31  
#define HASH_SIZE 128
```

i	v	a	n	o	v
105	118	97	110	111	118

```
int main()  
{  
    unsigned int h = hash("ivanov");  
}
```

Хэш-функции (Hash functions)

```
#define HASH_MUL 31
#define HASH_SIZE 128
```

i	v	a	n	o	v
105	118	97	110	111	118

```
unsigned int hash(char *s) {
    unsigned int h = 0;
    char *p;

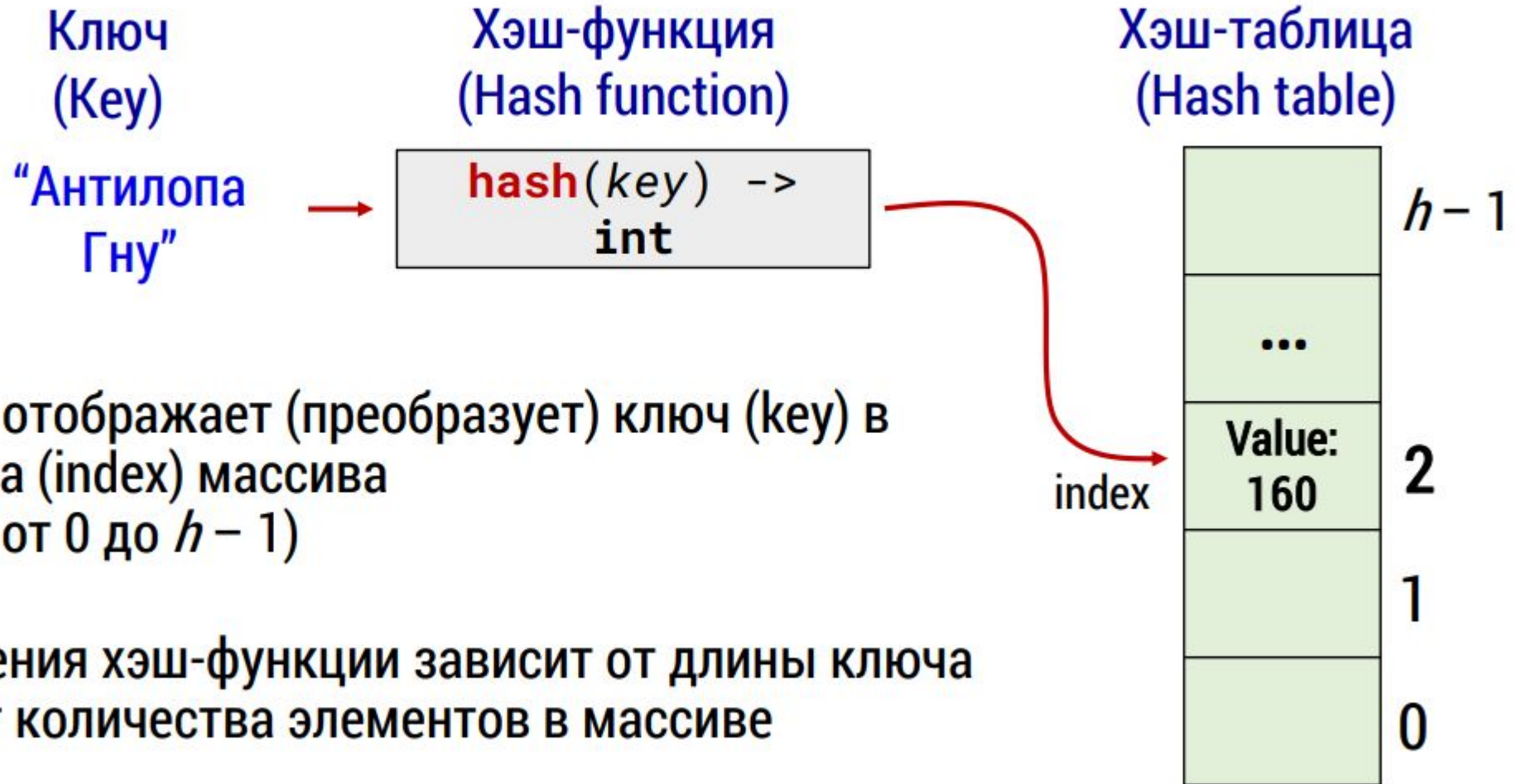
    for (p = s; *p != '\0'; p++)
        h = h * HASH_MUL + (unsigned int)*p;
    return h % HASH_SIZE;
}
```

```
h = 0 * HASH_MUL + 105
h = 105 * HASH_MUL + 118
h = 3373 * HASH_MUL + 97
h = 104660 * HASH_MUL + 110
h = 3244570 * HASH_MUL + 111
h = 100581781 * HASH_MUL + 118
return 3118035329 % HASH_SIZE // hash("ivanov") = 1
```


Хэш-таблицы (Hash tables)

- **Хэш-таблица (hash table)** – структура данных для хранения пар «ключ – значение»
- Доступ к элементам осуществляется по ключу (key)
- Ключи могут быть строками, числами, указателями, ...
- Хэш-таблицы позволяют в среднем за время $O(1)$ выполнять добавление, поиск и удаление элементов

Хэш-таблицы (Hash tables)



- **Хэш-функция** отображает (преобразует) ключ (key) в номер элемента (index) массива (в целое число от 0 до $h - 1$)
- Время вычисления хэш-функции зависит от длины ключа и не зависит от количества элементов в массиве
- Ячейки массива называются buckets, slots

ХЕШ-ТАБЛИЦА

Единственная сложность при работе с хеш-таблицами – коллизии, возникающие, когда для двух разных объектов хеш-функция возвращает один и тот же ключ.

Можно решить одним из двух способов:

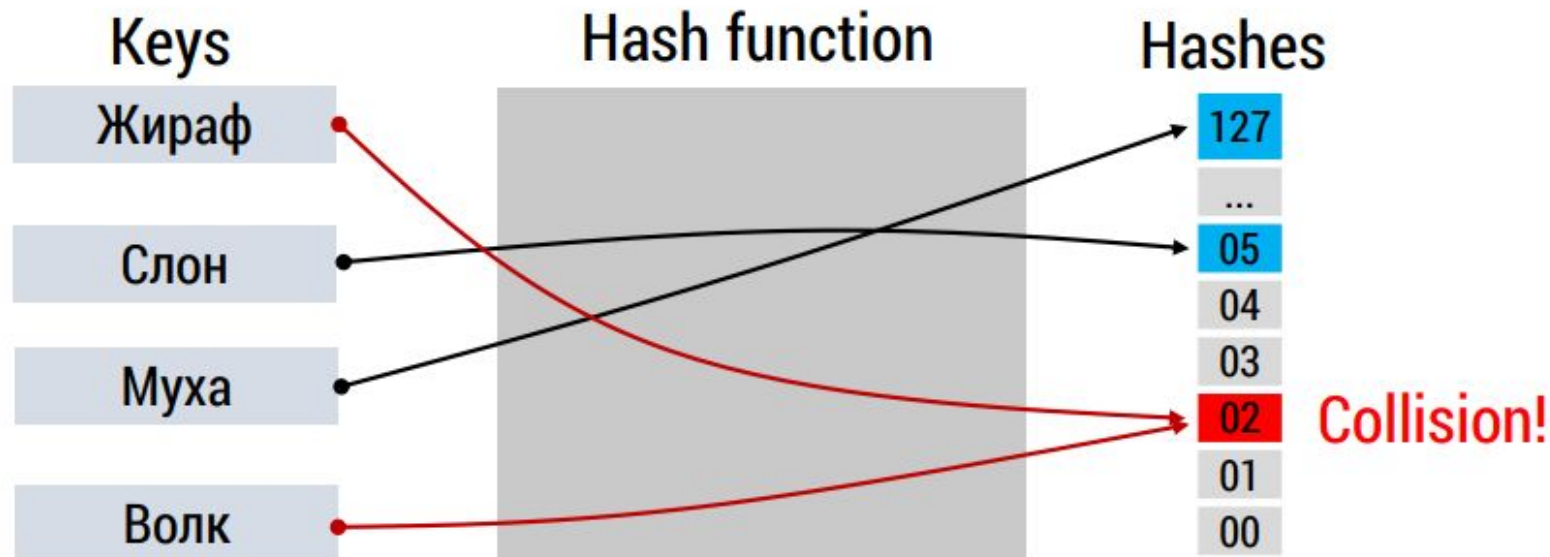
- разрешение коллизий на основе СПИСКОВ, (закрытая адресация)
- разрешение коллизий на основе табличного спуска. (открытая адресация)

Понятие коллизии (Collision)

- Коллизия (Collision) – это совпадение значений хэш-функции для двух разных ключей

$\text{hash}(\text{"Волк"}) = 2$

$\text{hash}(\text{"Жираф"}) = 2$



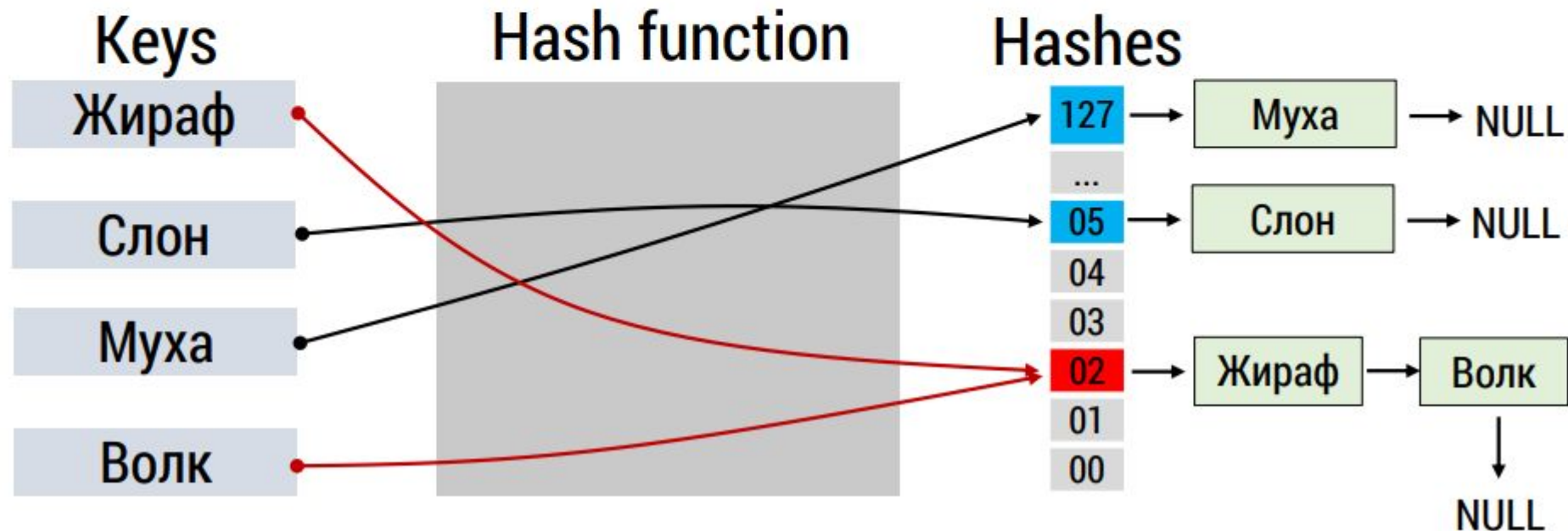
Существуют хэш-функции без коллизий –
совершенные хэш-ф функции (perfect hash function)

Разрешение коллизий (Collision resolution)

Метод цепочек (Chaining) – закрытая адресация

Элементы с одинаковым значением хэш-функции объединяются в связный список. Указатель на список хранится в соответствующей ячейке хэш-таблицы

- При коллизии элемент добавляется в начало списка
- Поиск и удаление элемента требуют просмотра всего списка



Разрешение коллизий (Collision resolution)

Открытая адресация (Open addressing)

В каждой ячейке хэш-таблицы хранится не указатель на связный список, а один элемент (ключ, значение). Если ячейка с индексом $\text{hash}(\text{key})$ занята, то осуществляется поиск свободной ячейки в следующих позициях таблицы.

Линейное хэширование (linear probing) – проверяются позиции:

$\text{hash}(\text{key}) + 1, \text{hash}(\text{key}) + 2, \dots, (\text{hash}(\text{key}) + i) \bmod h, \dots$

Если свободных ячеек нет, то таблица заполнена.

Пример:

- $\text{hash}(D) = 3$, но ячейка с индексом 3 занята
- Просматриваем ячейки: 4 – занята, 5 – свободна

Hash	Элемент
0	B
1	
2	
3	A
4	C
5	D
6	
7	

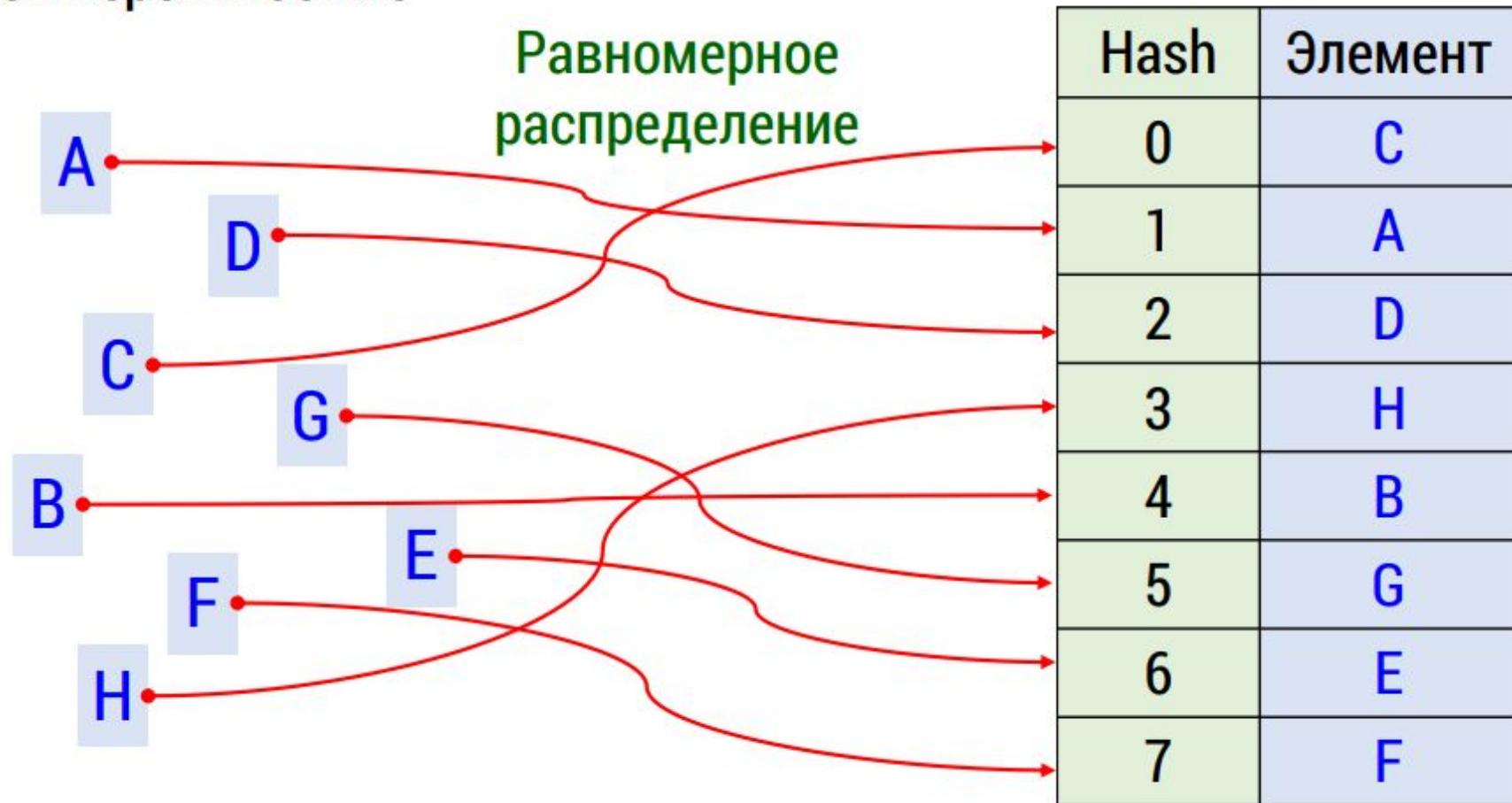
Требования к хэш-функциям

- **Равномерность (uniform distribution)** – хэш-функция должна равномерно заполнять индексы массива возвращаемыми номерами
- Желательно, чтобы все хэш-коды формировались с одинаковой равномерной распределенной вероятностью



Требования к хэш-функциям

- **Равномерность (uniform distribution)** – хэш-функция должна равномерно заполнять индексы массива возвращаемыми номерами
- Желательно, чтобы все хэш-коды формировались с одинаковой равномерной распределенной вероятностью



Криптографические хэш-функции

Хэш-функцией называется всякая функция $h: X \rightarrow Y$, легко вычисляемая и такая, что для любого сообщения M значение $h(M) = H$ (*свертка*) имеет фиксированную битовую длину.

- **Ключевые**
- **Бесключевые**

1. Устойчивость к коллизиям
2. Необратимость
3. Открытость к вычислению

Ключевые Называются *кодами аутентификации сообщений (КАС)* (*message authentication code (MAC)*).

Дают возможность без дополнительных средств гарантировать как правильность источника данных, так и целостность данных в системах с доверенной средой.

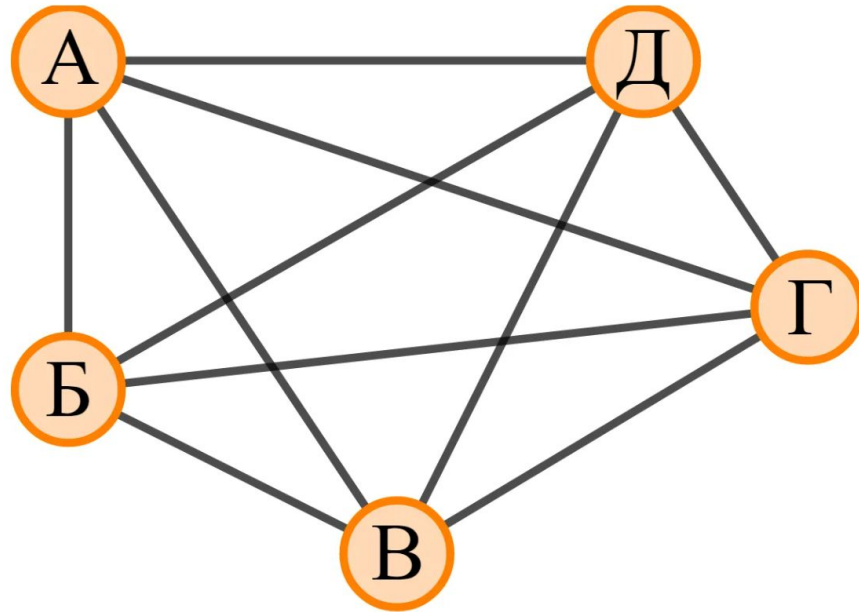
Бесключевые. Называются *кодами обнаружения ошибок* (*modification detection code (MDC)* или *manipulation detection code, message integrity code (MIC)*).

Дают возможность с помощью дополнительных средств (например, шифрования, использования защищенного канала или цифровой подписи) гарантировать целостность данных.

Основные криптографические хеш-функции

Название	Дата создания	Размер	Число раундов	Примечания
MD4 (<i>Message Digest 4</i>)	1990 г., Р.Риверст	128 бит	3	Уязвимости найдены в 1991 г., коллизии в 1996 г.
MD5 (<i>Message Digest 5</i>)	1991 г., Р.Риверст	128 бит	4	Псевдоколлизии найдены в 1993 г., коллизии в 1996 г., уязвимости найдены в 2004 г.
MD6 (<i>Message Digest 6</i>)	2008 г., Р.Риверст	переменный, $0 < d \leq 512$	переменное. По-умолчанию, Без ключа = $40 + \lceil d/4 \rceil$, с ключом = $\max(80, 40 + \lceil d/4 \rceil)$	
SHA-1 (<i>Secure Hash Algorithm Ver.1</i>)	1995, АНБ	160 бит	80	Уязвимости найдены в 2005 г.

Коллизия 2 рода



Коллизия 1 рода

Пусть есть 5 объектов принимающих 10 значений.

Значение конкретного элемента совпадает со значением другого элемента. (вероятность = $1/10$)

Коллизия 2 рода

Пусть есть 5 объектов принимающих 10 значений.

Значения каких-то двух объектов совпадают.

(Вероятность =?)

Парадокс дней рождения гласит, что в группе всего из 23 человек есть 50% шанс, что, по крайней мере, у двух людей совпадут даты дня рождения.

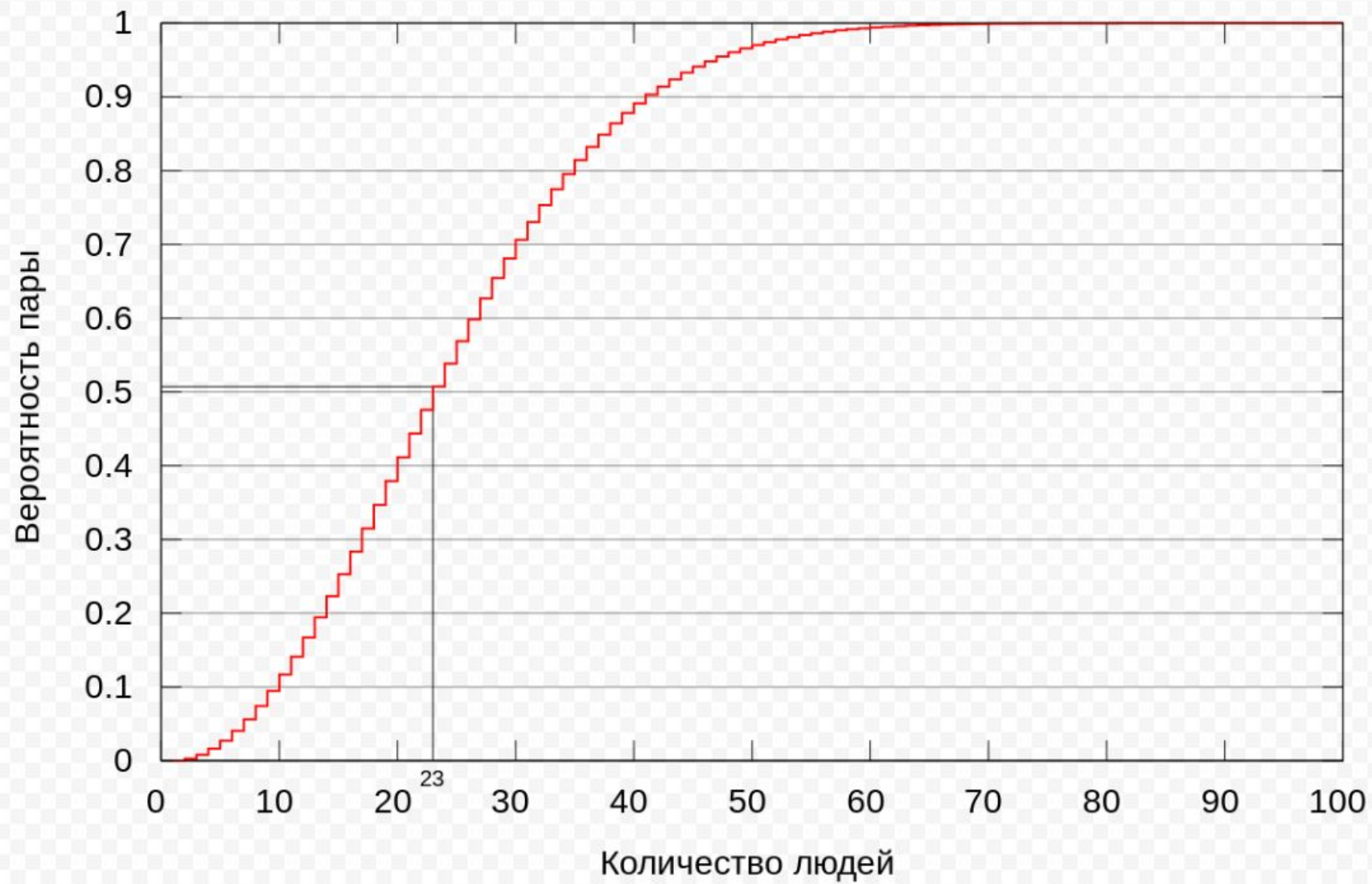
Вероятность, того, что дни рождения не совпадут

$$\begin{aligned}\bar{p}(n) &= 1 \times \left(1 - \frac{1}{365}\right) \times \left(1 - \frac{2}{365}\right) \cdots \left(1 - \frac{n-1}{365}\right) \\ &= \frac{365 \times 364 \cdots (365 - n + 1)}{365^n} \\ &= \frac{365!}{365^n (365 - n)!}\end{aligned}$$

Вероятность, того, что дни рождения совпадут

$$p(n) = 1 - \bar{p}(n)$$

Какова вероятность для 5-ти объектов с 10-ю вариантами значений?



Приближение

$$p(n) = 1 - \bar{p}(n) \approx 1 - e^{-\frac{n(n-1)}{2 \cdot 365}}$$

При $p(n)=1/2$, $N=365$ количество человек составит

$$n \approx \frac{1}{2} + \sqrt{\frac{1}{4} - 2 \cdot 365 \cdot \ln 0,5} = 22,9999.$$

Другое приближение

$$\frac{1}{2} \geq e^{-\frac{n^2}{2N}}$$

$$n \geq \sqrt{2 \ln 2 \cdot N} \approx 1,177\sqrt{N} \approx 23$$

Следствие

Если хеш функция имеет N возможных значений, достаточно создать примерно \sqrt{N} вариантов среди которых найдется 2 одинаковых (коллизия 2 рода) с вероятностью $1/2$