Семинар 9

Хеш-таблицы

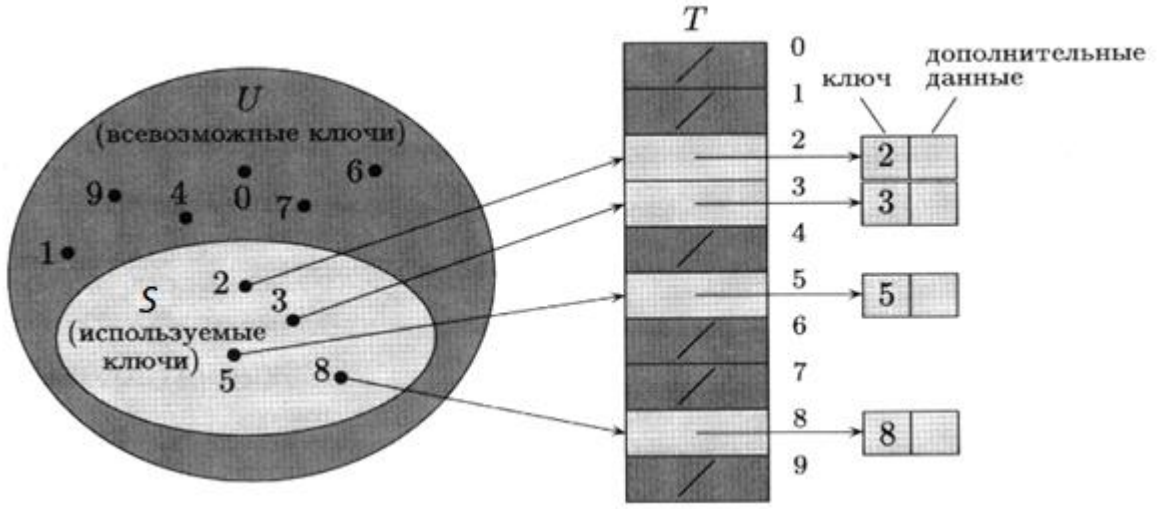
Что такое хеш-таблица (Х-Т)?

- Структура данных для некоторого динамического множества, поддерживающая операции «найти (просмотреть)», «вставить», «удалить». Операции должны выполняться быстро (за O(1)).
- Напоминает массив. По ключу записи (с помощью **хеш-функции**) определяется номер позиции (индекс) в Х-Т, где размещается ссылка на данные, ассоциированные с этим ключом.
- U – универсум (совокупность возможных ключей), S – множество используемых ключей

Схема *прямой адресации*:

Количество ключей =n= размер Х-Т

Если $|U|$ невелико, то это удобно.



Задача 1.

- Предположим, что динамическое множество *S* представлено таблицей *T* с прямой адресацией длины *n*. Опишите процедуру, которая находит максимальный элемент *S.* Чему равно время работы этой процедуры в наихудшем случае?

Задача 2.

- *Битовый вектор* - это массив битов дины *n*. Б.в. Занимает значительно меньше места, чем массив из n указателей. Каким образом можно использовать б.в. для представления динамического множества различных элементов без сопутствующих данных? Словарные операции должны выполняться за O(1).

$$b[x] = \begin{cases} 0 \\ 1 \end{cases}$$

Задача 3 о сумме двух чисел (варианты решения)

- Попытка №1: перебрать все пары x и y и сравнить их сумму с t. Сложность $O(n^2)$.
- Попытка №2: $(\forall x\ \exists!\ y = t - x)$

Вход: массив A из n целых чисел и целевое целое число t.

Выход: «да», если $A[i] + A[j] = t$ для некоторых $i, j \in \{1, 2, 3, ..., n\}$, «нет» — в противном случае.

```
for i = 1 to n do
   y := t − A[i]
   if A содержит y then     // линейный поиск
      return «да»
return «нет»
```

Сложность $O(n^2)$.

Задача о сумме двух чисел - продолжение

- Попытка №3: отсортируем массив

СУММА ДВУХ ЧИСЕЛ (РЕШЕНИЕ НА ОСНОВЕ ОТСОРТИРОВАННОГО МАССИВА)

Вход: массив A из n целых чисел и целевое целое число t.

Выход: «да», если $A[i] + A[j] = t$ для некоторых $i, j \in \{1, 2, 3, ..., n\}$, «нет» — в противном случае.

```
sort A     // используя подпрограмму сортировки
for i = 1 to n do
   y := t − A[i]
   if A содержит y then     // двоичный поиск
      return «да»
return «нет»
```

Сложность $O(n \log n + \log n) = O(n \log n)$.

- Попытка №4: решение на основе Х-Т

СУММА ДВУХ ЧИСЕЛ (РЕШЕНИЕ НА ОСНОВЕ ХЕШ-ТАБЛИЦЫ)

Вход: массив A из n целых чисел и целевое целое число t.

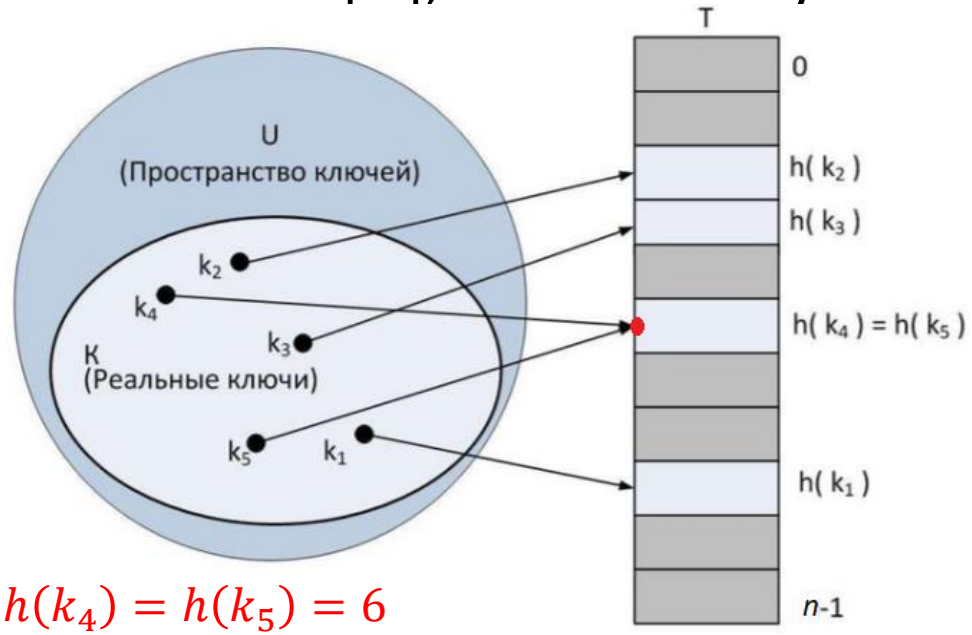
Выход: «да», если $A[i] + A[j] = t$ для некоторых $i, j \in \{1, 2, 3, ..., n\}$, «нет» — в противном случае.

```
H := пустая хеш-таблица
for i = 1 to n do
   Вставить A[i] в H
for i = 1 to n do
   y := t − A[i]
   if H содержит y then     // используя операцию «Просмотреть»
      return «да»
return «нет»
```

Сложность $O(n)$.

Пример: записная книжка с именами и телефонами ваших друзей и знакомых

- Операции: по имени найти телефон, вставить телефон нового знакомого, удалить телефон недруга. Если на имя отвести 10 символов, то всего имен будет $33^{10} = |U|$ *???*
- С помощью хеш-функции *h* по ключу(имени) будем вычислять номер ячейки, хранящей указатель на телефон друга с именем *k*.
- Длина Х-Т значительно меньше $|U|$, но может случиться ***коллизия.***



$k_4 = \text{Вася}, k_5 = \text{Маша}, h(k_4) = h(k_5) = 6$

Если $n \ll |S|$, то коллизии неизбежны.

Принцип Дирихле. Есть n голубей и m нор. Если $n > m$, то существует нора, в которой будет больше одного голубя.

ТЕСТОВОЕ ЗАДАНИЕ 12.3

Рассмотрим n человек со случайными днями рождения, причем каждый из 366 дней года равновероятен. (Предположим, что все n человек родились в високосный год.) Насколько большим должно быть n, прежде чем будет существовать 50 %-ный шанс, что у двух человек их день рождения будет совпадать?

а) 23

б) 57

в) 184

г) 367

Решение:

$\overline{P}(n)$ – все люди родились в разные дни

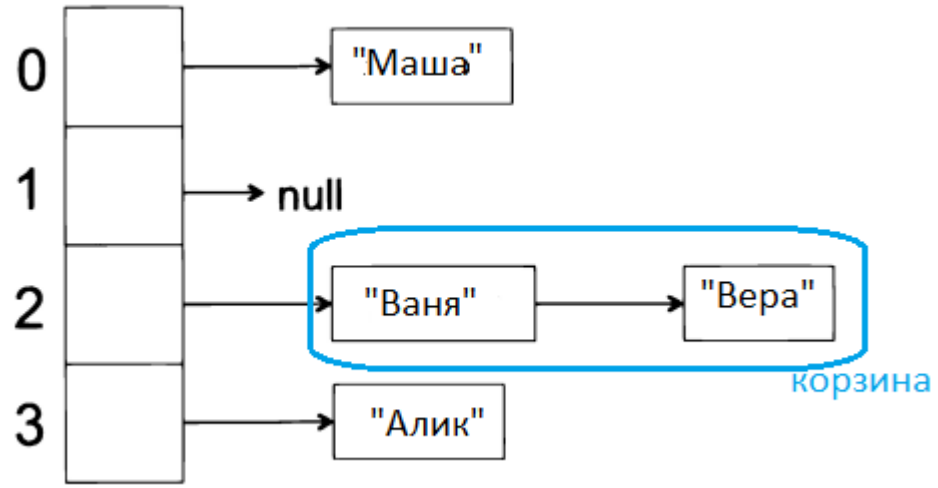
$(n \leqq 365)$

$P(n) = 1 - \overline{P}(n)$

Разрешение коллизий

1) Раздельное сцепление (метод цепочек).

Если записи вычисляются хеш-функцией h(k) в одну и ту же позицию Х-Т, то эти записи организуются в связный список, относящийся к этой позиции (корзину)



- Храним связный список в каждой корзине Х-Т.
- Для того, чтобы *Просмотреть/ Вставить/ Удалить* объект с ключом k, выполним эти операции в связном списке корзины $A[h(k)]$

Пусть X-T имеет размер n и хранит $100\ n$ объектов

Какой лучший и худший случаи хранения?

- По 100 в одной корзине;
- $100\ n$ в одной корзине?

Стремятся к равномерному хешированию.

$\alpha = \frac{|S|}{n}$ коэффициент заполнения

Среднее время операций неудачного поиска, удачного поиска, вставки и удаления: $\Theta(1 + \alpha)$

Задача.

- Продемонстрируйте происходящее при вставке в ХТ с разрешением коллизий методом цепочек ключей
- 5, 28, 19, 15, 20, 33, 12, 17, 10 $h(k) = k \ mod \ 9$

0	
1	28
2	
3	
4	
5	5
6	
7	
8	

2) Открытая адресация

$|S| \leq n$ (только *Вставить* и *Просмотреть*)

Каждый ключ связан с зондажной последовательностью. Первое число в последовательности – номер позиции, которую надо просмотреть первой, второе число – второй и т.д. Объект хранится в первой незанятой позиции зондажной последовательности.

а) Линейное зондирование (пробирование)- последовательность для ключа k:

$h(k), h(k)+1, \ldots$ - циклически переходит в начало.

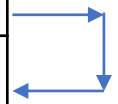
Пример: $h(k) = k \ mod \ 7$, ключи: 50, 700, 76, 85, 92, 73, 101

0	
1	
2	
3	
4	
5	
6	

0	
1	50
2	
3	
4	
5	
6	

0	700
1	50
2	
3	
4	
5	
6	76

0	700
1	50
2	85
3	
4	
5	
6	



0	
1	
2	
3	
4	
5	
6	

0	
1	
2	
3	
4	
5	
6	

Проблема – первичная кластеризация

Зондажные последовательности образуют группы. Требуется время, чтобы найти свободный слот или элемент.

б) Двойное хеширование

$h(k,i) = \left(h_1(k) + ih_2(k)\right) mod\ n$, i ячеек занято, $i = 0, 1, \dots, n-1$

h_1, h_2 - две хеш-функции

Пример: 79, 69, 72, 50, 98, 14

$h_1(k) = k\ mod\ 13$,
$h_2(k) = 1 + k\ mod\ 11$

Почему $h_2(k)$ д.б. взаимо-проста с n?

0	
1	79
2	
3	
4	69
5	98
6	
7	72
8	
9	14
10	
11	50
12	