Хеш-таблицы

Лекция 8

Применение

- Set (множество)
  - вставка элемента
  - удаление
  - проверка на наличие
- Map (словарь, ассоциативный массив)
  - Вставка пары (ключ, значение)
  - Удаление
  - Получение значения по ключу

Закрытая адресация

- Время работы в среднем O(n / M)

```
put(k, v)
    a[h(k)].add({k, v})

get(k)
    for p : a[h(k)]
        if p.first == k
            return p.second
    return null

delete(k)
    a[h(k)].delete(k)
```

Открытая адресация

- Если n = M, то можем долго искать пустое место
- Если n = 2M, то вероятность того, что следующая ячейка будет пустая 50%

```
put(k, v)
    i = h(k)
    while a[i] ≠ empty
        i = (i + 1) % M
    a[i] = {k, v}

get(k)
    i = h(k)
    while a[i] ≠ empty
        if a[i].first == k
            return a[i].second
        i = (i + 1) % M
    return null
```

Открытая адресация

- Когда заканчивается место – расширяемся и перехешируем

```
delete(k)
    i = h(k)
    while a[i] ≠ empty
        if a[i].first == k
            a[i] = {rip, rip}
            ripCnt++
            break
        i = (i + 1) % M
    if ripCnt + size > M / 2
        doRehashing()
```

Хеш-функции

- Для чисел
  - h(x) = (Ax % P) % M
    - P > M и простое
    - A – случайное
- Для строк
  - Полиномиальный хэш
    - h(S) = ((A^0s0 + A^1s1 ...) % P) % M
      - P > M и простое
      - A – случайное (обычно берут простое больше алфавита)