Program 5 due 10 pm **Friday**, May 6th

**Homework 10** due 10 pm **Wednesday**, May 4th

**Last Time**
- Hashing
  - terminology
  - designing a good hash function
  - choosing table size

**Today**
- Hashing
  - choosing table size (from last time)
  - expanding a hash table
  - handling collisions

Java Support for Hashing
Tree Map vs. Hash Map

**Next Time**
- **Read**: continue *Sorting*
  - Sorting Intro
  - Basic Sorts
    - bubble sort
    - insertion sort
    - selection sort
  - Better Sorts
    - heap sort
    - merge sort
Resizing the Hash Table

Naïve Expand

| 30 | 17 | 88 |

Rehashing

1.

2.

Complexity
Collision Handling using Open Addressing

Open Addressing

Linear Probing

166
359
263

| 440 | 266 | 124 | 246 | 337 | 351 |
Collision Handling using Open Addressing

Quadratic Probing

166
359
263

| 440 | 266 | 124 | 246 | 337 | 351 |

Double Hashing

probe sequences assuming \( H_k \) is index 0:

<table>
<thead>
<tr>
<th>Step size</th>
<th>Table size 10</th>
<th>Table size 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Collision Handling using Buckets

Buckets

Array Buckets

“Chained” Buckets

Tree Buckets
Java API Support for Hashing

hashCode method
- method of Object class
- returns an int
- default hash code is BAD - computed from object’s memory address

Guidelines for overriding hashCode:

Hashtable<K, V> and HashMap<K, V> class
- in java.util package
- implement Map<K, V> interface
  K
  V
  operations:

  - constructors allow you to set
    initial capacity (default = 16 for HashMap, 11 for HashTable)
    load factor (default = 0.75)
  - handles collisions with chained buckets
  - HashMap only:
  - Hashtable only:
## TreeMap vs HashMap

<table>
<thead>
<tr>
<th>TreeMap</th>
<th>HashMap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>