Announcements/Reminders:

- HW6 due tonight
- P3 due tomorrow
- P4 assigned

Last class:

- Hashing (cont'd)
  - Examples of hash functions
  - Non-Integer Keys
  - Collision Handling with open addressing and buckets

Today:

- Hashing (finish)
  - Open addressing (finish)
  - Hashing in Java

- Sorting
Hashing

What problem are we solving?

Idea:

Terminology:

- Key
- Hash table
- Table Size (TS)
- Load Factor (LF)
- Hash function
- Collision
- Ideal hashing
Collision Handling (advanced): Open Addressing

Linear probing

Quadratic probing

Double Hashing
Hashing in Java

hashCode() method:

- All Objects have it
- returns an int
- default is computed from memory address (NOT good!)

Overriding hashCode():

```java
public class CoordinatePair {
    private int x, y;

    public int hashCode() {
        return (x * 31) ^ y;
    }

    public boolean equals(Object o) {
        if (o instanceof CoordinatePair) {
            CoordinatePair other = (CoordinatePair) o;
            return (x == other.x && y == other.y);
        }
        return false;
    }
}
```
Hashtable<\texttt{K}, \texttt{V}> and HashMap<\texttt{K}, \texttt{V}> in Java

Both implement \texttt{Map<\texttt{K}, \texttt{V}>} interface

What are \texttt{K}, \texttt{V}?

Constructors

Operations

Collision handling

Differences
TreeMap vs. HashMap
Sorting

Problem

Solutions

Complexity (comparison sorts vs. others)

In-place Sorting