1 # CS3134 #19

11/11/03

Janak J Parekh

2 # Administrivia

- HW3 returned Thursday
- HW4 due today
- HW5 will be out shortly

3 # Agenda

- Finish hashing
- Heaps

4 # Collision handling: open addressing

- Just put the result in another cell
- *Linear probing*: put it in the very next cell
  - Leads to “clusters” making the hash table very inefficient
- *Quadratic probing*: space ’em out
  - x+1, x+4, x+9, x+16, x+25
  - Wraparound if necessary
  - Has other clustering properties

5 # Collision handling: open addressing (II)

- *Double hashing:*
  - Hash the key using a different function, and use that result as a step size (x+y)
    - Hash function must *never* return a zero, and should not be the same as the first hash function
    - stepSize = constant – (key % constant)
    - (constant is a prime less than table size)
  - Table size must be prime
- Other considerations
  - Duplicates are a problem with this method
  - Deletes?
  - Consider expanding the array: rehashing required
    - Load factor of the hash table very important

6 # Hash functions

- What makes a good hash function?
  - Fast to compute
- Random keys?
  - If already random distribution, just mod it
- Non-random keys
  - Need to “compress” information
  - Use as much data as possible
  - Table size should be prime
  - Book’s String example on page 565
- Folding: Break into groups and add together – for example, SSN
  - 1000 cells => 3-digit numbers

7 # Hashing efficiency

- All O(1) in theory, but…

- Load factor: % of table actually used – directly affects performance
- In general, quadratic probing and double hashing fare better than linear probing as the load factor goes up
- Separate chaining: linear function of load factor (can be > 1, since multiple entries per cell)
  - Generally want to avoid high loads…

8 What can't you do with hash tables?

- Specific ordering – it's essentially random
- Growable – can't use a linked list and maintain performance metrics
- Expect it to be automagically fast – need good hash functions
  - Although Java does have a number of hash functions built in…

9 Next time

- Heaps