Homework 04 - hash++

Focus: hash tables

Due: **Mon Feb 3, 2020 (bring to class)**

Reading:

- ❏ Open Data Structures, Chapter 5 Hash Tables, opendatastructures.org/ods-java/5_Hash_Tables.html
- ❏ Sedgewick Algorithms 3.4 Hash Tables, algs4.cs.princeton.edu/34hash
- ❏ Hash table animation, www.cs.usfca.edu/~galles/visualization/Algorithms.html
- ❏ Prof Bill Hash table notes, wtkrieger.faculty.noctrl.edu/csc210-spring2020/docs/hash_table_notes.pdf

thanks...yow, bill

Hash table terms

Define the following terms:

hash table hash function hash code associative array map key-value pair (K, V)	collision collision resolution linear probing, open addressing chaining, closed addressing load factor resizing rehash
java: hashCode(), HashMap, HashSet override modulo, %	

Q&A

Answer these:

1. Why does Prof Bill say that hash tables the most important data structure we will cover this semester?
2. Why are hash tables built with arrays and not linked lists?
3. Hash table get() and put() give O(1) expected performance. The worst case performance of a hash table is O(n). When does this terrible worst case happen? Why doesn't it happen very often?
4. What are the three rules for hash functions?
5. How is modulo used in a hash function?
6. Instead of all this chaining and linear probing, why don't we just create a hash function that doesn't cause collisions?
7. What are the two steps needed to resize a hash table?
8. I resize a hash table when it reaches some load factor. One program resizes at load factor of 50%; another at 75%. How does this impact the performance and memory used in each program?
9. How are String objects hashed in Java?
10. In my Java program, I have a hash table of GardenGnome objects. Should I override hashCode() for my GardenGnome class? What happens if I don't?