CS 124 Project 5

Due on Blackboard by Wednesday, April 18

For this project, you will hash the 1000 objects from your data set. You will experiment to see what size hash table and which collision detection algorithm works best for your data.

Implement

- You should have your 1000+ objects stored in a vector.
- Insert your objects into two hash tables:
  1) One with separate chaining.
  2) One with linear or quadratic probing or double hashing.
- Calculations for each hash table:
  a) Record the number of hash elements you look at (read) for each insertion. What is the maximum and average number of reads?
  b) Change the initial size of your hash table to at least 5 different prime numbers and see what happens to the numbers you record. For the probing hash table, what is the size after inserting all of your elements? Which hash table size is best for the data set?
  c) Change your hash function and/or your getKey function and see what happens to the numbers you record. What is the effect on the placement of elements in the hash table? Which hash/getKey function is better for your data set?
  d) Consider how removing and searching compares with inserting, in terms of number of hash elements read. Would they read more, less, or the same?
  e) Include graphs of your data and answers to all of the above questions in your writeup.
- How often would each of the three operations (inserting, removing, searching) typically be used with your data set? Which hash collision method works best for your data set? Why? Include answers to these questions in your writeup.
- You must submit your .cpp and header file(s), your data file(s), and your writeup. Your source files should include output of all the numbers collected from the prompts above (preferably to files). Please submit your writeup in PDF format.

Extra Credit

For up to 10 points extra credit (at the grader's discretion), you can:

- Use timers to see how long it takes you to insert/find elements in the hash table(s).
- Compare those times with the time it takes to insert/find elements stored in other data structures (e.g. an unsorted vector, a sorted vector, a binary search tree, an AVL tree, a splay tree, a heap, etc.) The more structures you include, the better!

Grading

The project is out of 65 points.

5 pts	Program compiles and runs.
5 pts	Code style. Readable, naming style is consistent, comments where appropriate.
5 pts	Hashed at least 1000 objects using separate chaining.
5 pts	Hashed at least 1000 objects using probing.
5 pts	Used at least 5 different hash table sizes, as specified above.
5 pts	Used at least two hash functions and/or getKey functions, as specified above.
20 pts	Recorded the number of reads for each type of hashing.
10 pts	Analyzed the results and wrote about everything outlined above.
5 pts	Writeup: professional, grammatically correct, cites sources.