

CS 124 Project 5

Due on Blackboard by Wednesday, December 6

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 a hash table with separate chaining.
 - 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 size of your hash table to at least 5 different prime numbers and see what happens to the numbers you record. Which hash table size is best for the data set?
 - c) Consider how removing and searching compares with inserting, in terms of number of hash elements read. Would they read more, less, or the same? You should collect the data to justify your answer.
 - d) Include answers to all of the above questions in your writeup.
- Insert your objects into a hash table with linear or quadratic probing.
 - a) You should be rehashing your table as in the code uploaded to Blackboard and the textbook code. What is the size of your hash table after inserting all of your elements?
 - b) Again record the number of hash elements looked at (read) for each insertion. What is the maximum and average?
 - c) Again record how removing and searching compares with inserting, in terms of number of hash elements read.
 - d) Include analysis of all of the above 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 different hash algorithms (i.e. change your hash methods). See what effect that has on the placement of elements in the hash table and the numbers you collect.
- Use timers to see how long it takes you to find elements in the hash table(s).
- Compare those times with the time it takes to 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 60 points.

- 5 pts Program compiles and runs.
- 5 pts Code style. Readable, naming style is consistent, comments where appropriate.
- 8 pts Hashed at least 1000 objects using separate chaining
- 3 pts Used at least 5 different hash table sizes, as specified above.
- 8 pts Hashed at least 1000 objects using probing.
- 16 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, at least 500 words.