Hashing and Operator Overloading

9-18-2003

Opening Discussion

- What did you think about the quiz?
- Do you have any questions about the assignment?
- In your program you have something like a vector or list of URLs/web pages. If a user were type in a URL what order would the operation be for you to find it? What ways can you think of that we could speed it up? How fast can you get it?

Finishing a Static Linked List

- Let's go look at the code for the static linked list.
- Also, to help you understand it I have updated the original linked list code so that it uses a sentinel.
- All this code is still on the web. I'm not moving it. If something isn't readable send me an e-mail to let me know.

Look-up in O(1)

- You are already aware that the ideal for functions that access data in a container is O(1). That is to say that no matter how many items are in the container, it takes us roughly the same amount of time to do our main functions.
- Think about what this means. When you are looking for something you only have to check a fixed number of possibilities.

Basics of Hashing

- A vector can give is O(1) operations for looking things up and setting them if we have an integer key for placement (and if we allow holes which the list interface doesn't).
- A hash just uses a function to figure out where a particular element goes in our vector instead of the full key. The functions maps keys to a smaller range.

Conflict Resolution by Linking

- By virtual of the fact that we have mapped the keys to a smaller range it is possible to have collisions. That is two keys that map to the same index in the hash.
- The easiest way to deal with this is to keep a linked list of all the elements that map to the same place. The obvious name for this is linking.

Hash Functions

- To make a hash efficient we have to have a good mapping function from keys to indexes.
- A mapping function is good if the keys for the elements we are inserting are distributed among the indexes with equal probability. That way we reduce collisions as much as possible.
- Sometimes other features are also required.

Division Method

- Perhaps the easiest method of doing a hash function is to mod the key by the number of elements in the hash table. This maps an integer key to the right range.
- Your book argues that you have to be careful about picking the size of the has table when doing this. They recommend a prime number that is not close to a power of two so all bits are used.

Multiplication Method

- In the multiplication method we use a hashing function of the following form.

$$h(k) = \lfloor m(k \cdot A \mod 1) \rfloor$$

- 0<A<1 and we can pick m to be a power of 2. Your book describes an efficient way of doing this without floating point numbers. Knuth suggests the following value for A. $A \approx (\sqrt{5} - 1)/2$

Operator Overloading

- C++ gives you the ability to redefine the meaning of operators. This can be a helpful tool when you create your own classes.
- Overloading << and >> allows easy use with the iostream libraries.
- Overloading comparisons helps with doing universal sorting.
- Overloading [] is good for containers.

Details in C++

- There are two ways to overload an operator. Both are similar. If the first argument for an operator is the type of the given class, you can define the operator with a method called operator? (where ? is replaced by the operator). The second argument (if any) is given in the argument list.
- When the first argument isn't that type you need to define a free standing function. The name is the same, but now the argument list must include all of the arguments.

Minute Essay

- Write a little code snippet for how you would overload the [] operator on a hash table. You don't have to include all the details. I'd use the division method just to keep the code simple.
- Assignment #2 is due a week from today. Start working on it now.