Lab 5: Hash Tables

Week of November 26, 2018

Objective

To replace an ADT Table implemented as a BST with an implementation using a hash table.

Exercise

The file `Lab5.java` and other files described below will not be provided until this Sunday 25 November 2018 after the Assignment 4 hand-in folder closes at 5:35 PM, since `Lab5.java` contains a copy of the solution to Assignment 4. Before that time, you can begin writing a standard `Table` class implemented as hash table, as described below.

The file `Lab5.java` contains a copy of the solution to Assignment 4, which reads in and executes Lenert programs, storing variable records (a variable identifier and the variable's value) in a Table that is implemented as a BST. Your job is to rewrite the Table class so that it is implemented with a hash table (details below), without changing anything else.

Note that the file `Lab5.java` contains THREE already-completed classes:

- A `Lab5` class, which prompts the user for an input file name (using keyboard input), then reads in and executes each of the Lenert programs in the file, printing out the variables at the end of each program. You must NOT change anything in this class.
- A `VariableRecord` class, which stores a variable's identifier (a `String`) and its value (an `int`). You must NOT change anything in this class, either.
- The `Table` class, which stores variable records using the variable's identifier as the key. You will change this class (details below).

The hash table implementation: Your new implementation of the `Table` class as a hash table must:

- Use an array of size 23.
- Use the polynomial hash code (implemented using Horner's method, of course) to hash the key (the variable's identifier) to an array position, using the coefficient $a = 13$.
- Use separate chaining to resolve collisions — each variable record stored at a position in the hash table array is stored in a node and those nodes (which all hashed to the same array location) are linked together into a linked list.

Also note that

- The chain of nodes at a particular array position does not have to be ordered.
- When printing the variable records stored in the table, you do not have to print the variables in any particular order.

Changes to the `Table` class: You will need to make many changes inside the `Table` class:

- The instance variable used for a BST must be changed to instance variables that are appropriate to a hash table.
- The `private Node` class must be changed to a `Node` class appropriate to a hash table using separate chaining.
- The bodies of the `public Table` methods must be changed to use the hash table. But the headers of the `public Table` methods should NOT be changed at all.
- You can add new `private` helper methods and constants as appropriate to the `Table` class.

Note that the comments at the beginning of the Table class have already been changed appropriately, but none of the other comments in that class have been changed.

Provided input files: Your program should run correctly on all three provided input files `Lab5testData.txt`, `Lab5data.txt`, and `Lab5data2.txt`.