Alex Bernstein
10/7/15
CSE 502N

Lab 2: Hash Table

My lab 2 implements a standard hash table with record insertion, removal, find, and table extension methods. The table is doubled when it is found to be full (i.e. when the insertion method fails). The table constructor has a "maxSize" parameter. This parameter is ignored.

Removal replaces the record with a premade "deleted" record. I initially had a new "deleted" record created for each deleted element, but I realized that this was unnecessary and created a single "deleted" record within the class that is placed at each "deleted" slot.

I accomplish table doubling by cloning the old table, and iteratively looking at each spot in that clone and calling my insert method to rehash every record (except for "deleted") in the array. This iteratively looks at each slot in the old array, which may be slow for long array values. Note that my table starts with a length of two, and doubles that value repeatedly, ensuring that the length of the table is always a power of two. An alternative to this would be to ensure the value of the array is always prime. In order to do this, one could maintain a list of previously used prime table lengths, and repeatedly multiply them together and subtract 1 in order to generate a new prime table length.

Find works by hashing the key values, and iterating through slots according to multiplicative hashing rules. The iteration quits on the first null slot, or when the multiplicative hashing returns the base value (which must be the first to recur). For each non-null and non-"deleted" slot, find first compares each Record's stored hash value (an integer). If the integer is correct, it then does a string compare to ensure that two different strings didn't hash to the same value. If both comparisons pass, the string is returned.