Your work for this must be finished and shown to your lab TA by the end of this lab session or the start of your next session.

1. Download the hash table code (hash.zip) from the course webpage under Lab 9. You will be comparing three different open addressing schemes. Complete the following functions in hash.cpp:

```
void Hash::qinsert(int k) {
    // Insert k in the hash table.
    // Use open addressing with quadratic probing and hash(k) = k % m.

void Hash::linsert(int k) {
    // Insert k in the hash table.
    // Use open addressing with linear probing and hash(k) = k % m.

void Hash::dinsert(int k) {
    // Insert k in the hash table.
    // Use open addressing with double hashing. Use the existing hash function
    // and also implement a second hash function.
```

You must complete these functions so that they call tallyProbes on every successful insertion, with the number of probes required for that insertion.

2. Run some experiments to see how the average number of probes per insertion differs when using the different methods. Try to answer the following questions:

(a) Under what load factors is linear probing just as good as quadratic probing? When does quadratic begin to win out?

(b) How does the choice of hash function affect double hashing? Can you devise a hash function that makes double hashing just as good as quadratic probing? Can you make it even better?

(c) When does double hashing begin to win against both of the other schemes? (i.e. at what capacities and load factors do you see a significant gain over the other methods?)

(d) Finally, which open addressing scheme would you choose and why?

Note that the main method has been set up to perform some tests for you. For example, hash q 9000 10000 will insert 9,000 random keys into a hash table of size 10,000 using quadratic probing, while hash d 500000 1000000 will insert 500,000 random keys into a hash table of size 1,000,000 using double hashing. You are welcome (in fact, encouraged!) to modify the main method to automate some of your experiments.