Assignment 2
50 points

Assume that we have a hash table structured as a vector of lists, resolving collisions by sequential search of buckets, as in the generic hash table template `HashTable<K,D,H>` distributed in the file `hashtbl.h`. Without making any assumptions on the hash function used, show the following, where n is the number of items in the table and b is the number of buckets (or slots) in the table implementation.

Problem 1. Let $e(k)$ denote the expected number of buckets of size k. Show that

$$e(k) = bp(k).$$

Hint: b is the frequency, p is the probability.

Problem 2. Let $p(k)$ denote the probability a given bucket has exactly k items. Show that the expected bucket size is

$$\sum_{k=0}^{n} kp(k) = \frac{n}{b}.$$

Hint: The events "bucket has exactly k items" are mutually exclusive.

[Actual Case] Assume that we have a hash table structured as a vector of lists, as above, and we have an actual instance of a table with actual (possibly non-optimal) hash function.

Problem 3. Let $b(k)$ denote the actual number of buckets of size k. Show that the size of the table is

$$n = \sum_{k \geq 0} kb(k).$$

Hint: Count the items in different ways.

Problem 4. What is the smallest safe upper index for the sum in the formula above? Explain your answer.

Hint: When do we know buckets are empty?

[Best Possible Case] Assume that we have assigned keys to buckets randomly, so that each key has a uniformly likely probability of being assigned to a bucket. (This is called "simple uniform hashing" in the textbook - see page 259.)

Problem 5. Show that the probability that a given bucket is empty is

$$p(0) = \left(\frac{b-1}{b}\right)^n.$$

Hint: The probability that one particular item is not in a given bucket is the same as the probability the item is in one of the other $b - 1$ buckets.

Problem 6. Let $0 \le k \le n$. Show that the probability that a given bucket has size k is

$$p(k) = \binom{n}{k}\left(\frac{1}{b}\right)^k\left(\frac{b-1}{b}\right)^{n-k}.$$

Hint: For a given selection of k items to constitute a bucket of size k, they must each be in the bucket and the other $n-k$ items must not be in the bucket. How many such (mutually exclusive) selections are there?

Problem 7. Using the formulas above, show that the bucket size distribution $\{e(k)\}$ satisfies the iterative calculation:

$$\begin{aligned} e(0) &= b\left(\frac{b-1}{b}\right)^n && \text{initial condition} \\ e(k) &= \left(\frac{n-k+1}{k}\right)\left(\frac{1}{b-1}\right)e(k-1) && \text{for } k = 1\ldots n. \\ e(k) &= 0 && \text{for } k > n. \end{aligned}$$

Hint: Use the notation $p = 1/b$, $q = (b-1)/b$. Note that $p + q = 1$ and $p/q = 1/(b-1)$. Then look at $\frac{e(k)}{e(k-1)}$ and simplify.

The last three problems parallel the HashAnalysis programming assignment - construction of algorithms resulting in implementations of the `MaxBucketSize` and `Analysis` methods of `HashTable<KeyType, DataType, HashType>`, as specified in the Project document. The answers may be in the form of C or C++ pseudo-code (where arrays index from 0) and should be recognizable from your actual code turned in for the Hash Analysis project.

Problem 8. Use the results of the exercises above to devise algorithms that calculate:

(a) The number of non-empty buckets
(b) The maximum bucket size
(c) The actual average search time (assuming equally likely queries)

Provide an asymptotic analysis of the runtime and runspace of your algorithms.

Problem 9. Use the results of the exercises above to devise an algorithm that calculates the actual bucket size distribution $\{b(k)\}$ from the actual table instance. Provide an asymptotic analysis of the runtime and runspace of your algorithm.

Problem 10. Use the results of the exercises above to devise an algorithm that calculates the theoretical best-case bucket size distribution $\{e(k)\}$ for "simple uniform hashing" based on the number of buckets b and the number of table entries n. Provide an asymptotic analysis of the runtime and runspace of your algorithm.

Experience. Provide a short discussion of lessons learned during the testing of various hash functions and load factors on actual data.

Assemble your solution paper as follows:

- For each question, repeat the question (including assumptions) on the paper, and then provide your solution.
- Take some time to get your math typeset correctly.
- Convert to a pdf document and turn that document in via Blackboard under "Assignment 2".