These are some review questions to test your understanding of the material. Some of these questions may appear on an exam.

1 Hash Tables

1.1 Name two desirable properties of a good hash function.

1.2 What is a "collision" in a hash table?

1.3 What is the "clustering" problem in hash tables?

1.4 Describe the "division" and "multiplication" methods of generating hash values.

1.5 Describe the "separate chaining" and "open addressing" collision resolution strategies.

1.6 What do the terms "linear probing" and "quadratic probing" mean?

1.7 An open-addressing hash function is a function:

$$h : U \times \{0, 1, \ldots\} \to \{0, 1, \ldots, m-1\}$$

where m is the size of the hash table and U is the set of key values. Using a hash-table size of 10, and $U = \{89, 18, 49, 58, 69\}$, write a hash function that uses linear probing. Use that hash function to generate the hash values for the elements of U. Do the same using quadratic-probing.

1.8 The average time performance of insertion and searching operations on a hash table are $O(1)$. This is much better than the performance of a binary search tree for the same operations. Given this wonderful performance of hash tables compared to binary search trees, when would you want to use a binary search tree instead of a hash table?

1.9 In an open-addressing hash table using linear probing, the average number of probes for successful, S, and unsuccessful, U, search are

$$S \approx \frac{1}{2}\left(1 + \frac{1}{1-\lambda}\right)$$

$$U \approx \frac{1}{2}\left(1 + \frac{1}{(1-\lambda)^2}\right)$$

where λ is the load factor of the table.

Suppose you want a hash table that can hold at least 1000 elements. You want successful searches to take no more than 4 probes on average and unsuccessful searches to take no more than 50.5 probes on average. What is the maximum load factor you can tolerate in your hash table? If table size is to be prime, what is the smallest table you can use?