1. Suppose we use a hash function $h$ to hash $n$ distinct keys into an array $T$ of length $m$. Assuming simple uniform hashing, what is the expected number of collisions? More precisely, what is the expected cardinality of $\{(x, y) : x \neq y : h(x) = h(y)\}$?

2. Show that if $|U| > nm$, there is a subset of $U$ of size $n$ consisting of keys that all hash to the same slot, so that the worst-case searching time for hashing with chaining is $\Theta(n)$.

3. Consider a version of the division method in which $h(k) = k \mod m$, where $m = 2^p - 1$ and $k$ is a character string interpreted in radix-$2^p$. Show that if string $x$ can be derived from string $y$ by permuting its characters, then $x$ and $y$ hash to the same value. Give an example of an application in which this property would be undesirable in a hash function.

4. Suppose that we use double hashing to resolve collisions; that is, we use the hash function $h(k, i) = (h_1(k) + ih_2(k)) \mod m$. Show that the probe sequence $(h(k, 0), h(k, 1), \ldots, h(k, m-1))$ is a permutation of the slot sequence $(0, 1, \ldots, m-1)$ if and only if $h_2(k)$ is relatively prime to $m$.

5. A hash table of size $m$ is used to store $n$ items, with $n \leq m/2$. Open addressing is used for collision resolution.
   (a) Assuming uniform hashing, show that for $i = 1, 2, \ldots, n$, the probability that the $i$-th insertion requires strictly more than $k$ probes is at most $2^{-k}$.
   (b) Show that for $i = 1, 2, \ldots, n$, the probability that the $i$-th insertion requires more than $2 \log n$ probes is at most $1/n^2$.
   (c) Let the random variable $X_i$ denote the number of probes required by the $i$-th insertion. You have shown in part (b) that $Pr\{X_i > 1/n^2\} \leq 1/n^2$. Let the random variable $X = \max_{1 \leq i \leq n} X_i$ denote the maximum number of probes required by any of the $n$ insertions. Show that $Pr\{X > 2 \log n\} < 1/n$.
   (d) Show that the expected length of the longest probe sequence is $O(\log n)$.

6. Give recursive algorithms that perform preorder and postorder tree walks in $\Theta(n)$ time on a tree of $n$ nodes.

7. If all keys in a binary search tree are distinct, the successor of a node $x$ is the node with the smallest key greater than $\text{key}[x]$. Develop a procedure that finds the successor of $x$ in $T$.

8. Describe a binary search tree on $n$ nodes such that the average depth of a node in the tree is $\Theta(\log n)$ but the height of the tree is $\omega(\log n)$. How large can the height of an $n$-node binary search tree be if the average depth of a node is $\Theta(\log n)$?

9. Show that the longest path from a node $x$ in a red-black tree to a descendant leaf has length at most twice that of the shortest path from node $x$ to a descendant leaf.

10. What is the largest possible number of internal nodes in a red-black tree with black-height $k$? What is the smallest possible number?
11. Show that any arbitrary $n$-node tree can be transformed into any other arbitrary $n$-node tree using $O(n)$ rotations. (Hint: First show that at most $n - 1$ right rotations suffice to transform any tree into a right-going chain.)