Quiz 1 Practice Problems

1 Asymptotic Notation

Decide whether these statements are **True** or **False**. You must briefly justify all your answers to receive full credit.

1. Let f and g be asymptotically nonnegative functions. Then, at least one relationship of $f(n) = O(g(n))$ and $g(n) = O(f(n))$ must always hold.

2. For every two positive functions f and g, if $g(n) = O(n)$, then $f(g(n)) = O(f(n))$.

3. Suppose $f(n) = 4\, f(n/4) + n$ for $n > 8$, and $f(n) = O(1)$ for $n \leq 8$. Similarly, suppose $g(n) = 3\, g(n/4) + n \lg n$ for $n > 8$, and $g(n) = O(1)$ for $n \leq 8$. Then $f(n) = \Theta(g(n))$.

4. Let $f(n) = \sqrt{n}$ and $g(n) = n \cdot (n \bmod 100)$. Then $f(n) = O(g(n))$.

5. Let $f(n)$ and $g(n)$ be two asymptotically positive functions. If $f(n) = \Theta(g(n))$ then $|f(n) - g(n)| = \Theta(f(n) + g(n))$.

2 Binary Search Trees

1. Width of Binary Trees

   Let T be a binary tree on n nodes. Recall that the *height of* T is the number of nodes in the longest path from the root. We divide the nodes in T into "levels", where a nodes is in level i if it is at distance i from the root. We define the *width of* T as the size of its largest level.

   (a) Describe a construction of a binary tree on n nodes with width $\Theta(1)$. What is the height of this tree?

   (b) Describe a construction of a binary tree on n nodes with $n/2$ leaves and width $\Theta(1)$.

   (c) Describe a construction of a binary tree on n nodes with height $\Theta(\log n)$ and width $\Theta(n)$.

   (d) Describe a construction of a binary tree on n nodes with height $\Theta(n)$ and width $\Theta(n)$.

   (e) Describe a construction of a binary tree on n nodes with height $\Theta(\log n)$ and width $\Theta(n/\log n)$.

3 Hashing

Decide whether these statements are **True** or **False**. You must briefly justify all your answers to receive full credit.

1. Assuming very low chance of a spurious match, it is possible to search for k different words of length n in a string of length N using Rabin-Karp, with complexity $O(N + nk)$.

2. Inserting in a hashtable with open addressing takes $O(1)$ time in the worst case.

3. In the worst case open addressing with linear probing has assymptotically better look-up times than collision resolution by chaining.

Do the following problems.

4. Suppose that a hash table containing n slots has collisions resolved by chaining, and suppose that n keys are inserted into the table. Under the assumption of simple uniform hashing, explain why the probability P_k that any slot contains k or more keys satisfies

$$P_k \leq n \binom{n}{k} (1/n)^k \; .$$

5. Given two sets X and Y, each containing n integers, show how hashing can be used to find the size of $|X \cap Y|$ in expected $\Theta(n)$ time.

4 Dynamic Programming

Decide whether these statements are **True** or **False**. You must briefly justify all your answers to receive full credit.

1. It is possible to speed up any algorithm by creating a dynamic programming version of the algorithm.

Do the following problems.

2. Problem 15-2: Printing Neatly from CLRS, page 354.

3. You bring an ℓ–foot log of wood to your favorite sawmill. You want it cut in k specific places: $\ell_1, \ell_2, \ldots, \ell_k$ feet from the left end. The sawmill charges $\$x$ to cut an x–foot log any place you want.

   (a) Give a dynamic programming algorithm that determines the order in which they should cut your log in order to minimize the cost. The running time of your algorithm should by polynomial in k. Analyze the running time.

   (b) Paul Bunyan says, "Always make the cut as close to the center of the log as possible," and to do this recursively on the two pieces. Show that Bunyan's algorithm does not minimize the cost. (Hint: Give a counter example.)