Practice Quiz 2

- Do not open this quiz booklet until you are directed to do so. Read all the instructions first.
- When the quiz begins, write your name on every page of this quiz booklet.
- The quiz contains four multi-part problems. You have 120 minutes to earn 108 points.
- This quiz booklet contains **16** pages, including this one. Extra sheets of scratch paper are attached. Please detach them before turning in your quiz.
- This quiz is closed book. You may use one handwritten A4 or $8\frac{1}{2}'' \times 11''$ crib sheet. No calculators or programmable devices are permitted.
- Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem. Do not put part of the answer to one problem on the back of the sheet for another problem, since the pages may be separated for grading.
- Do not waste time and paper rederiving facts that we have studied. It is sufficient to cite known results.
- Do not spend too much time on any one problem. Read them all through first, and attack them in the order that allows you to make the most progress.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.
- Good luck!

Problem	Points	Grade	Initials
1	40		
2	25		
3	25		
4	18		
Total	108		

Name: ________________________________
Circle your recitation letter and the name of your recitation instructor:

David A B Steve C D Hanson E F

Problem 1. True or False, and Justify [40 points]

Circle **T** or **F** for each of the following statements, and briefly explain why. The better your argument, the higher your grade, but be brief. Your justification is worth more points than your true-or-false designation. Each part is worth **4 points**.

(a) T F A preorder traversal of a binary search tree will output the values in sorted order.

(b) T F The cost of searching in an AVL tree is $O(\lg n)$.

(c) T F The expected amount of space used by a skip list on n elements is $O(n)$.

(d) T F The height of a randomly built binary search tree is $O(\log n)$ with high probability; therefore in randomized quicksort, every element is involved in $O(\log n)$ comparisons with high probability.

(e) T F Any mixed sequence of m increments and n decrements to a k-bit binary counter (that is initialized to zero and is always non-negative) takes $O(m+n)$ time in the worst case.

Spring 2004: See CLRS pg. 408-409

(f) T F Consider a directed acyclic graph $G = (V, E)$ and a specified source vertex $s \in V$. Every edge in E has either a positive or a negative edge weight, but G has no negative cycles. Dijkstra's Algorithm can be used to find the shortest paths from s to all other vertices.

(g) **T** **F** Given a bipartite graph G and a matching M, we can determine if M is maximum in $O(V + E)$ time.

(h) **T** **F** Given a random skip list on n elements in which each element has height i with probability $(1/3)^{i-1}(2/3)$, the expected number of elements with height $\lg_3 n$ is $O(1)$.

(i) T F Suppose you are given an undirected connected graph with integer edge weights in which each vertex is degree 2. An MST can be computed for this graph in $O(V)$ time.

(j) T F Consider an undirected, weighted graph G in which every edge has a weight between 0 and 1. Suppose you replace the weight of every edge with $1 - w(u, v)$ and compute the minimum spanning tree of the resulting graph using Kruskal's algorithm. The resulting tree is a maximum cost spanning tree for the original graph.

Problem 2. Short Answer Problems [25 points]

Give *brief*, but complete, answers to the following questions. Each problem part is worth **5 points**.

Amortized Analysis

You are to maintain a collection of lists and support the following operations.

(i) *insert*(item, list): insert item into list (cost = 1).

(ii) *sum*(list): sum the items in list, and replace the list with a list containing one item that is the sum (cost = length of list).

(a) Use the Accounting Method to show that the amortized cost of an *insert* operation is $O(1)$ and the amortized cost of a *sum* operation is $O(1)$.

(b) Use the Potential Method to show that the amortized cost of an *insert* operation is $O(1)$ and the amortized cost of a *sum* operation is $O(1)$.

Balanced Search Trees

(c) Show that the number of node splits performed when inserting a single node into a 2-3 tree can be as large as $\Omega(\lg n)$, where n is the number of keys in the tree. (E.g. give a general, worst-case example.)

Faster MST Algorithms

(d) Given an undirected and connected graph in which all edges have the same weight, give an algorithm to compute an MST in $O(E)$ time.

FFT

(e) Snowball Throwing

Several 6.046 students hold a team snowball throwing contest. Each student throws a snowball with a distance in the range from 0 to $10n$. Let M be the set of distances thrown by males and F be the set of distances thrown by females. You may assume that the distance thrown by each student is unique and is an integer. Define a team score to be the combination of one male and one female throw.

Give an $O(n \log n)$ algorithm to determine every possible team score, as well as how many teams could achieve a particular score. This multi-set of values is called a *cartesian sum* and is defined as:

$$C = \{m + f : m \in M \text{ and } f \in F\}$$

Problem 3. Dynamic Programming [25 points]

Santa Claus is packing his sleigh with gifts. His sleigh can hold no more than c pounds. He has n different gifts, and he wants to choose a subset of them to pack in his sleigh. Gift i has utility u_i (the amount of happiness gift i induces in some child) and weight w_i. We define the weight and utility of a *set of gifts* as follows:

- The weight of a set of gifts is the *sum* of their weights.
- The utility of a set of gifts is the *product* of their utilities.

For example, if Santa chooses two gifts such that $w_1 = 3, u_1 = 4$ and $w_2 = 2, u_2 = 2$, then the total weight of this set of gifts is 5 pounds and the total utility of this set of gifts is 8. All numbers mentioned are positive integers and for each gift i, $w_i \leq c$. Your job is to devise an algorithm that lets Santa maximize the utility of the set of gifts he packs in his sleigh without exceeding its capacity c.

(a) [5 points] A greedy algorithm for this problem takes the gifts in order of increasing weight until the sleigh can hold no more gifts. Give a small example to demonstrate that the greedy algorithm does not generate an optimal choice of gifts.

(b) **[10 points]** Give a recurrence that can be used in a dynamic program to compute the maximum utility of a set of gifts that Santa can pack in his sleigh. Remember to evaluate the base cases for your recurrence.

(c) **[7 points]** Write pseudo-code for a dynamic program that computes the maximum utility of a set of gifts that Santa can pack in his sleigh. What is the running time of your program?

(d) **[3 points]** Modify your pseudo-code in part **(c)** to output the actual set of gifts with maximum utility that Santa packs in his sleigh.

Problem 4. Reliable distribution

A communication network consists of a set V of nodes and a set $E \subset V \times V$ of directed edges (communication links). Each edge $e \in E$ has a ***weight*** $w(e) \geq 0$ representing the cost of using e. A ***distribution*** from a given source $s \in V$ is a set of directed $|V| - 1$ paths from s to each of the other $|V| - 1$ vertices in $V - \{s\}$. The ***cost*** of a distribution is the sum of the weights of its constituent paths. (Thus, some edges may be counted more than once in the cost of the distribution.)

(a) Give an efficient algorithm to determine the cheapest distribution from a given source $s \in V$. You may assume all nodes in V are reachable from s.

(b) One of the edges in the communication network may fail, but we don't know which one. Give an efficient algorithm to determine the maximum amount by which the cost of the cheapest distribution from s might increase if an adversary removes an edge from E. (The cost is infinite if the adversary can make a vertex unreachable from s.)