Quiz 1

- Do not open this quiz booklet until you are directed to do so. Read all the instructions first.
- The quiz contains 4 problems, several with multiple parts. You have 80 minutes to earn 80 points.
- This quiz booklet contains 10 pages, including this one, and a sheet of scratch paper which can be detached.
- This quiz is closed book. You may use one double-sided letter ($8\frac{1}{2}'' \times 11''$) or A4 crib sheet. No calculators or programmable devices are permitted. Cell phones must be put away.
- Write your solutions in the space provided. If you run out of space, continue your answer on the back of the same sheet and make a notation on the front of the sheet.
- Do not waste time deriving facts that we have studied. It is sufficient to cite results from class.
- When we ask you to "give an algorithm" in this quiz, describe your algorithm in English or pseudocode, and provide a short argument for correctness and running time. You do not need to provide a diagram or example, unless it helps make your explanation clearer.
- Do not spend too much time on any one problem. Generally, a problem's point value is an indication of how many minutes to spend on it.
- 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. Please be neat.
- Good luck!

Problem	Title	Points	Parts	Grade	Initials
0	Name	1	1		
1	True or False	24	8		
2	Translation	25	5		
3	All Pairs Shortest Red/Blue Paths	18	3		
4	Telephone Psetting	12	3		
Total		80			

Name: _____

Circle your recitation:

R09	R10	R03	R04	R07	R08	R05	R06
F10	F11	F12	F1	F1	F2	F2	F3
Yuri	Yuri	Igor	Igor	Sarah	Sarah	Lin	Lin

Problem 0. Name. [1 point] Write your name on every page of this exam booklet! Don't forget the cover.

Problem 1. True or False. [24 points] (8 parts)

Circle **T** or **F** for each of the following statements to indicate whether the statement is true or false, and briefly explain why. Your justification is worth more points than your true-or-false designation.

(a) T F The solution to the recurrence $T(n) = 2^n T(n-1)$ is $T(n) = \Theta((\sqrt{2})^{n^2+n})$. (Assume $T(n) = 1$ for n smaller than some constant c).

(b) T F The solution to the recurrence $T(n) = T(n/6) + T(7n/9) + O(n)$ is $O(n)$. (Assume $T(n) = 1$ for n smaller than some constant c).

(c) T F In a simple, undirected, connected, weighted graph with at least three vertices and unique edge weights, the heaviest edge in the graph is in no minimum spanning tree.

(d) T F The weighted task scheduling problem with weights in the set $\{1, 2\}$ can be solved optimally by the same greedy algorithm used for the unweighted case.

(e) T F Two polynomials p, q of degree at most $n - 1$ are given by their coefficients, and a number x is given. Then one can compute the multiplication $p(x) \cdot q(x)$ in time $O(\log n)$.

(f) T F Suppose we are given an array A of n distinct elements, and we want to find $n/2$ elements in the array whose median is also the median of A. Any algorithm that does this must take $\Omega(n \log n)$ time.

(g) T F There is a density $0 < \rho < 1$ such that the asymptotic running time of the Floyd-Warshall algorithm on graphs $G = (V, E)$ where $|E| = \rho |V|^2$ is better than that of Johnson's algorithm.

(h) T F Consider the all pairs shortest paths problem where there are also weights on the vertices, and the weight of a path is the sum of the weights on the edges and vertices on the path. Then, the following algorithm finds the weights of the shortest paths between all pairs in the graph:

APSP-WITH-WEIGHTED-VERTICES(G, w):

1 **for** $(u, v) \in E$
2 Set $w'(u, v) = (w(u) + w(v))/2 + w(u, v)$
3 Run Johnson's algorithm on G, w' to compute the distances $\delta'(u, v)$ for all $u, v \in V$.
4 **for** $u, v \in V$
5 Set $d_{uv} = \delta'(u, v) + \frac{1}{2}(w(u) + w(v))$

Problem 2. Translation [25 points] (5 parts)

You have been hired to manage the translation process for some documentation. Unfortunately, different sections of the documentation were written in different languages: n languages in total. Your boss wants the entire documentation to be available in all n languages.

There are m different translators for hire. Some of those translators are volunteers that do not get any money for their services. Each translator knows *exactly* two different languages and can translate back and forth between them. Each translator has a non-negative hiring cost (some may work for free). Unfortunately, your budget is too small to hire one translator for each pair of languages. Instead, you must rely on chains of translators: an English-Spanish translator and a Spanish-French translator, working together, can translate between English and French. Your goal is to find a minimum-cost set of translators that will let you translate between every pair of languages.

We may formulate this problem as a connected undirected graph $G = (V, E)$ with non-negative (i.e., zero or positive) edge weights w. The vertices V are the languages for which you wish to generate translations. The edges E are the translators. The edge weight $w(e)$ for a translator e gives the cost for hiring the translator $w(e)$. A subset $S \subseteq E$ of translators can be used to translate between $a, b \in V$ if and only if the subgraph $G_S = (V, S)$ contains a path between a and b. The set $S \subseteq E$ is a translation network if and only if S can be used to translate between all pairs $a, b \in V$.

(a) Prove that each minimum spanning tree of G is also a minimum-cost translation network.

(b) Give an example of a minimum-cost translation network that is not a minimum spanning tree of G.

(c) Give an efficient algorithm that takes G as input, and outputs a minimum-cost translation network of G. State the runtime of your algorithm in terms of the number of languages n and the number of potential translators m.

Your bosses have decided that the previous approach to translation doesn't work. When attempting to translate between Spanish and Portuguese — two relatively similar languages — it degrades the translation quality to translate from Spanish to Tagalog to Mandarin to Portuguese. There are certain clusters of languages that are more closely related than others. When translating between two languages that lie within the same cluster, such as Spanish and Portuguese, the translation is of high quality when the sequence of languages used to translate between them is completely contained within the cluster.

More formally, the language set V can be divided into disjoint clusters $C_1, \ldots, C_k$. Each cluster C_i contains languages that are fairly similar; each language is contained in exactly one cluster. Your bosses have decided that a translation between $a, b \in C_i$ is high-quality if and only if all of the languages used on the path from a to b are also in C_i. The translator set S is a high-quality translation network if and only if it is a translation network, and for any language cluster C_i and any languages $a, b \in C_i$, S can be used for a high-quality translation between a and b.

(d) Suppose that S is a minimum-cost high-quality translation network. Let $S_i = S \cap (C_i \times C_i)$ be the part of the network S that lies within the cluster C_i. Show that S_i is a minimum-cost translation network for the cluster C_i.

(e) Give an efficient algorithm for computing a minimum-cost high-quality translation network. Analyze the runtime of your algorithm in terms of the number of languages n and the number of translators m.

Problem 3. All Pairs Shortest Red/Blue Paths. [18 points] (3 parts)

You are given a directed graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}$. In addition, each edge of the graph is either red or blue. The shortest red/blue path from vertex $i \in V$ to vertex $j \in V$ is defined as the shortest path from i to j among those paths that go through *exactly* one red edge (if there are no such paths, the length of the shortest red/blue path is ∞).

We can represent this graph with two $n \times n$ matrices of edge weights, W_r and W_b, where W_r contains the weights of all red edges, and W_b contains the weights of all blue edges.

(a) Given the Floyd-Warshall algorithm below, how would you modify the algorithm to obtain the lengths of the shortest paths that only go through blue edges?

FLOYD-WARSHALL(W):

```
1  n = W.rows
2  D^(0) = W
3  for k = 1 to n
4      let D^(k) = (d_ij^(k)) be a new n × n matrix
5      for i = 1 to n
6          for j = 1 to n
7              d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
8  return D^(n)
```

(b) How would you modify your algorithm from part (a) to keep track not only of shortest paths with only blue edges, but also those with exactly one red edge, and to output the lengths of the shortest red/blue paths for all pairs of vertices in this graph?

(c) Prove the correctness of your algorithm using a loop invariant.

Problem 4. Telephone Psetting. [12 points] (3 parts)

Upon realizing that it was 8:30 PM on Wednesday and he had not yet started his 6.046 pset, Ben Bitdiddle found $n-1$ other students (for a total of n students) in the same situation, and they decided to do the pset, which coincidentally had n problems in total, together.

Their brilliant plan for finishing the pset in time was to sit in a circle and assign one problem to each student, so that for $i = 0, 1, \ldots, n-1$, student i did problem number i, and wrote up a solution for problem i meriting $p(i)$ points. Then, they each copied the solutions to all the other problems from the student next to them, so that student i was copying from student $i-1$ (and student 0 was copying from student $n-1$).

Unfortunately, they were in such a hurry that the copying chain degraded the quality of the solutions: by the time student i's solution to problem i reached student j, where $d = j - i \ (mod\ n)$, $d \in \{0, 1, \ldots, n-1\}$, the solution was only worth $\frac{1}{d+1}p(i)$ points.

(a) Write a formula that describes the total pset score $S(x)$ for student x, where the total pset score is the sum of the scores that student x got on each of the n problems.

(b) Describe a simple $O(n^2)$ algorithm to calculate the pset scores of all the students.

(c) Describe a $O(n \log n)$ algorithm to calculate the pset scores of all the students.

SCRATCH PAPER

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2012