Computer Science 477/677 Fall 2002 Examination, October 31, 2002

Name:____

No books, notes, scratch paper, or calculators. Use pen or pencil, any color. Use the rest of this page and the backs of the pages for scratch paper. If you need more scratch paper, it will be provided.

The entire test is 120 points.

1. [20 points] The following is a list of the edges of a weighted graph whose nodes are the letters from A to F. Draw the graph and find a minimal spanning tree.

A	B	3
A	D	4
B	D	2
B	C	7
B	E	6
C	E	1
C	F	1
D	E	5
D	F	6
E	F	2

2. [20 points] Find an optimal prefix code for the following weighted alphabet.

A	5
B	2
C	1
D	6
E	9
F	3

3. [40 points] Describe a dynamic programming algorithm to find the longest monotone increasing subsequence of a finite sequence of numbers. (For example, the longest monotone increasing subsequence of the sequence 1,4,9,2,5,2,3,7,1 is 1,2,2,3,7.) Hint: there is a fairly straightforward algorithm that takes $O(n^2)$ time, where n is the length of the sequence.

4. [40 points] Consider the sliding block puzzle described in the Science News article. Suppose that you decide to write a computer program to find the shortest solution, that is, to move the red car out of the parking lot in the smallest possible number of moves. In a very high level way, describe what you would do. Your answer should describe which abstract data structures you should use. For example, you might want to use a stack, a queue, a heap, a search structure, an array, a list, or whatever. You also will want to describe what algorithm or algorithms you will use. Do not write code; even pseudocode is unnecessary, although acceptable if you can't figure out how else to say it. I want something that could be generalized to other sliding block puzzles, rather than something that only works for that particular puzzle. The individual questions below are to guide you, but feel free to express your answer in a completely different form.

   Use the rest of this page and the next page for your answer. That should be much more than enough space.

   (a) The *initial configuration* is shown in the picture. What information would a *configuration* contain? How could you implement it?

   (b) What would a *final configuration* be?

   (c) What would a *move* be?

   (d) The configurations and moves together form an object of what *abstract data type*? (I want a **one word** answer.)

   (e) Which standard search algorithm would you use? What data structure(s) would you need?

   (f) Would you precompute all configurations and moves before you start running the search algorithm? Why or why not?

   (g) Would you use memoization? (Hint: I would.) How, and why? (Describe any needed data structures.)

   (h) Give a high-level description of your program, in at most one short paragraph.

(Continuation of your answer to problem **??**.)