Final Exam

CPSC 311 Analysis of Algorithms
Andreas Klappenecker

- This exam contains 1+7 problems. You have 120 minutes to earn 100 points.
- This exam is closed book.
- You are allow to use a nonprogrammable calculator, although you will not need one.
- 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.
- Recall the Aggie code of honor. Cheating will have severe consequences.

NAME: ________________________ **TAMU ID:** __________________

The work shown in this exam is my own.

Signature required: ________________________________

NAME: ______________________ **TAMU ID:** ________________

Problem 0 (2 points) Write your name and TAMU ID on each page.

Problem 1 (13 points)
(a) Order the following eight functions according to their asymptotic growth. Write $f(n) \prec g(n)$ if and only if $f(n) \in O(g(n))$.

$$\ln n, \quad n, \quad 2^{(\log n)^{1/2}}, \quad e^n, \quad n2^n, \quad 4^{\log n}, \quad \log\log n, \quad \log(n!).$$

(b) Give an example of a single nonnegative function $f(n)$ such that for each function $g(n)$ in part (a), $f(n)$ is neither $O(g(n))$ nor $\Omega(g(n))$. Explain.

NAME: ________________________ **TAMU ID:** __________________

Problem 2 (15 points) Solve the following recurrence equations.
(a) $T(n) = 4T(n/2) + \Theta(n)$.

(b) $T(n) = 3T(n/3) + n \log_3 n$.

(c) $T(n) = T(n/2) + \Theta(\sqrt{n})$.

Problem 3 (15 points) Consider the following two problems.

P1 **Input:** A set of n squares, specified by their corner points.
Question: Is there any point in the plane that is covered by k or more squares?

P2 **Input:** A graph $G = (V, E)$ with n vertices, and a number k.
Question: Is there a set of k mutually adjacent vertices?

Clearly, both problems are in NP. There exists a simple translation from **P1** to **P2**: Make a vertex for each square and add an edge between a pair of vertices if the corresponding squares overlap.

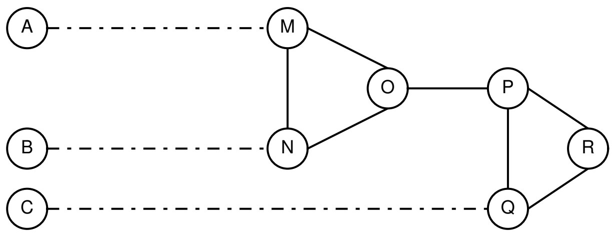
(a) If **P1** is NP-complete, would this translation imply that **P2** is NP-complete? Give a short and clear explanation.

(b) If **P2** is NP-complete, would this translation imply that **P1** is NP-complete? Give a short and clear explanation.

(c) If we replace squares by circles in **P1**, is then the simple translation still valid? For instance, does the existence of a triangle in the graph mean that there exists a point covered by at least three circles?

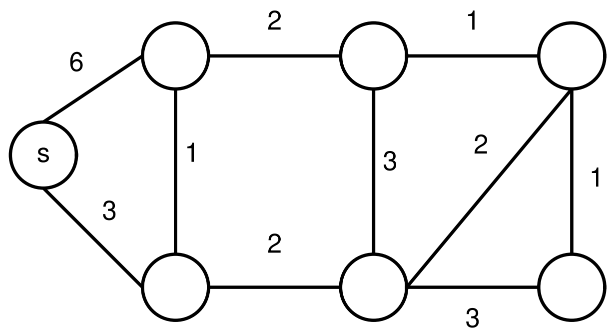
NAME: ______________________ **TAMU ID:** _________________

Problem 4 (10 points) Suppose that we have three colors *true*, *false*, and *green*.



Assume that R is colored *true*. When is this graph 3-colorable, assuming that A, B, C can be either colored *true* or *false*? Explain.

Problem 5 (15 points) (a) Use Kruskals algorithm to determine the minimum spanning tree of the following graph. Use the node labeled s in case the algorithm needs a starting node.



(b) Construct a graph with uniquely determined minimum spanning tree. The graph should have 5 vertices and 10 edges with weights 1 or 2.

(c) What is the time complexity of Kruskal's algorithm on input of a graph $G = (V, E)$?

(d) Does Kruskal's algorithm require that the graph is connected? Explain.

NAME: ________________________ **TAMU ID:** __________________

Problem 6 (10 points)

(a) Use the Euclidean algorithm to compute $\gcd(10, 7)$.

(b) Use the extended Euclidean algorithm to compute the multiplicative inverse of 7 modulo 10.

Problem 7 (20 points) A common subsequence of two strings A and B is a subsequence of A and a subsequence of B. We know how to write programs to determine the length of the longest common subsequence. Define the edit distance between the two strings A and B to be the smallest number of deletions and insertions of single letters to change A into B.

(a) Consider the two strings $A = \texttt{minimum}$ and $B = \texttt{maximum}$. Determine the length m of the longest common subsequence of A and B. Determine the edit distance d between A and B (use the definition above).

(b) Suppose that you have two input strings A and B, both of length n. Suppose that your program determines the length m of the longest common subsequence of A and B. Determine the edit distance between A and B using this information.

(c) Prove the result that you have given in (b).

(d) What is the complexity of the algorithm that we have developed in the first exam to determine the length of the longest common subsequence of two strings A and B? Assume that both input strings A and B are of length n.

(e) What algorithm design technique was used to derive this longest common substring algorithm?