MCGILL UNIVERSITY – COMP 360

TEST 1

February 28, 2011

Student Number:	
Family Name(s):	
Given Name(s):	

- There are 7 pages in total (including this page).
- You have 60 minutes for this test.
- This test is worth 10% of your final mark.
- Answer each question directly on the test paper, in the space provided. Use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and indicate clearly the part of your work that should be marked.

Question 1 (out of 10)	
Question 2 (out of 10)	
Question 3 (out of 10)	
Total (out of 30)	

Question 1 [10pt]

(a)[1pt] True of False: SAT is is polytime black-box reducible to every **NP**-complete problem?

True

False

(b)[2pt] Give precise definition of polytime many-one reduction.

(c)[2pt] Consider the polytime reduction from SAT to 3CNF-SAT that was discussed in lecture. Give the 3CNF formula that results from transforming the following formula:

$$(\neg x_1 \lor x_2) \land (x_1 \land \neg x_2)$$

Clearly list the new variables and the clauses.

Question 1 continues here

(d)[1pt] Prim's algorithm is for which of the following problems (circle your choice):

Minimum Spanning Tree problem

Set Cover problem

Shortest path problem

(e)[2pt] Recall that MAX-Knapsack is the following optimization problem. The input to the problem consists of a set S of n items, and a number W. The i-th item of S has weight w_i and value v_i, so:

$$S = \{(w_1, v_1), (w_2, v_2), \dots, (w_n, v_n)\}$$

(W and v_i, w_i are positive integers and $w_i \leq W$ for $1 \leq i \leq n$.) The problem is to find the maximum total value of a subset of S whose total weight does not exceed W.

Clearly describe in plain English a greedy algorithm that achieves a $\frac{1}{2}$ approximation ratio, that is, it outputs a subset of the items that has total value at least $\frac{1}{2}opt$, where opt is the optimal solution to the problem. (You are not required to prove the correctness of your algorithm.)

(f)[2pt] Clearly describe Kruskal's algorithm in plain English. (You can also give pseudo-code with sufficient comment, but this is not recommended.)

Question 2 [10pt]

Recall that a set of vertices of an undirected graph G is said to be *independent* if there is no edge between any two of them. The Independent Set problem (IND) is as follows.

Input: (G, k), where G is an undirected graph and k is a positive integer.

Output: Accept if and only if there is an independent set in G of size at least k.

Recall that for the Multiple Interval Scheduling problem (MIS) each request consists of a *set* of intervals and there is a single processor that can process only one request at a time. Thus, if we schedule a request R then we cannot schedule any other request that contains some interval which overlaps one of the intervals in R. The MIS problem is as follows.

Input: A set of n requests $R_1, R_2, \ldots, R_n$, and an integer k. Here for $1 \leq i \leq n$ the request R_i is specified by a list of intervals

$$R_i = (s_1^i, f_1^i), (s_2^i, f_2^i), \ldots, (s_{m_i}^i, f_{m_i}^i)$$

where all s_j^i, f_j^i are nonnegative integers and $s_j^i \leq f_j^i$, for all $1 \leq j \leq m_i$. (Representation of numbers are not very important here, for example, they are written in unary.)

Output: Accept if and only if at least k requests can be scheduled together.

In this question you are asked to show that IND $\leq_m$ MIS, that is, to give a polytime many-one reduction from IND to MIS and prove its correctness. (Recall that in Assignment 4 you were asked to show that CLIQUE $\leq_m$ MIS.)

(a)[5pt] Clearly describe your polytime many-one reduction.

Question 2 continues here

(b)[5pt] Prove the correctness of your reduction.

Question 3 [10pt]

Consider the following problem of scheduling a given set of n tasks. Here all tasks require one unit of time, and there is only one processor which can execute one task at a time. The cost of executing the i-th task at time t is tc_i, where c_i a positive parameter associated with the task. The problem is to schedule all tasks in such a way that minimizes the total cost. (We start at time 0.)

Input: A set of n tasks and their associated parameters $c_1, c_2, \ldots, c_n$. Here all c_i are positive integers.
Output: A schedule of all tasks on a single processor with minimum total cost.

In this question you are asked to give a greedy algorithm that solves this problem in polynomial time and prove its correctness.

(a)[5pt] Clearly describe your algorithm in plain English or give pseudo-code for it (you do not have to give pseudo-code for sorting).

Question 3 continues here

(b)[5pt] Prove the correctness of your algorithm.