Some sample problems for final exam
Use this in conjunction with Exam 1 and Exam 2 Practice Problems
Final Exam Date: **Wednesday**, Dec 18, 7 - 9 pm
Sections A and D: Lagomarcino 1155
Section C: Lagomarcino 1411

- Final exam is comprehensive.
- For any problem or sub-problem, if you write "*Do not grade*" and nothing else, you will receive 20% credit for it.
- For all algorithm problems, part of the grade depends on the efficiency.
- In general you may use the following algorithms and their time bounds as a "black box" on the exam (that is, unless modifications are needed, you do not need to write the code nor derive runtime for them): sorting, breadth-first search, depth-first search, Prim's algorithm, Kruskal's algorithm, Dijkstra's algorithm, topological sort.

  However, if you modify any of these methods, then you must write a complete description of the modified method. Merely stating the modification does not suffice.
- You can assume that the following problems are known to be NP-complete: SAT, 3-SAT, Vertex Cover, Set Cover, Independent Set, Set Packing, 3-dimensional Matching, Subset Sum, Knapsack, Hamiltonian Cycle, Traveling Salesman Problem.
- You should understand the definitions of NP and NP-completeness and the meaning of a polynomial-time reduction ($\leq_p$). We will not ask you to create and prove concrete reductions (such as "prove that 3-SAT $\leq_p$ Vertex Cover").

Some sample problems

1. Let $A = [a_1, \cdots a_n]$ be an array of positive and negative integers. Find i and j such that $1 \leq i \leq j \leq n$ and such that
$$\sum_{k=i}^{j} a_k$$
is maximized. Design a dynamic programing algorithm (without memoization/recursion) for this problem. State the recurrence relation.

2. Given an undirected graph $G = (V, E)$, a subset S of V is called a *vertex cover* if for every edge $\langle u, v \rangle \in E$, at least one of u or v belongs to S. Let T be a binary tree. Give a dynamic programming algorithm (without memoization/recursion) that computes the smallest size vertex cover of T. State the recurrence relation.

3. Given an undirected graph $G = (V, E)$, a subset S of V is called an *independent set* if for every $x, y \in S$, there is no edge from x to y in G.

   Let $G = (V, E)$ be a graph such that $V = \{v_1, v_2, \cdots, v_n\}$ and $E = \{\langle v_i, v_{i+1} \rangle \mid 1 \le i < n\}$. (That is, G is a "line graph".) Assume that each vertex v has a non-negative weight $w(v)$. Given a set S of vertices, the weight of S is

   $$\sum_{v \in S} w(v).$$

   Give a dynamic programming based algorithm (without memoization/recursion) to compute a maximum weight independent set for a line graph G. State the recurrence relation.

4. Let $G_1 = (V, E_1)$ and $G = (V, E_2)$ be two undirected graphs such that $V = \{1, 2, \cdots, n\}$. We say that G_1 is isomorphic to G_2 if there is a permutation $\Pi : \{1, 2, \cdots n\} \to \{1, 2, \cdots, n\}$ such that for every $1 \le i \le j \le n$, if $\langle i, j \rangle \in E_1$, then $\langle \Pi(i), \Pi(j) \rangle \in E_2$. Show that the decision problem of whether two input graphs are isomorphic or not belongs to NP.

5. Given an undirected graph $G = (V, E)$, a set $S \subseteq V$ is a *dominating set* if every vertex $v \in V$ either belongs to S or is adjacent to a vertex in S. We are interested in the following decision problem: Given G and k, is there a dominating set of size $\le k$ in G. Show that this decision problem is in NP.

6. Suppose you are a faculty member at a large midwestern university. A student comes to your office making the following claim: "I came up with an algorithm that, given a weighted directed graph G and an integer k, produces (in polynomial time) a bipartite graph G' with the property that there is a TSP tour of total cost $\le k$ in G if and only if there is a perfect matching in G'. " What should you conclude?

7. Suppose that the Bulletproof Rabbit problem has been shown to be in NP, and that Bulletproof Rabbit $\le_p$ 3-SAT. Can we conclude that Bulletproof Rabbit is NP-complete?

8. Suppose you have just proved that Vertex Cover reduces to the Invisible Canteloupe problem. Identify which additional facts you can conclude, and explain briefly.

   - Invisible Canteloupe is also reducible to Vertex Cover
   - Invisible Canteloupe is not reducible to Vertex Cover
   - There isn't enough information to tell whether Invisible Canteloupe is reducible to Vertex Cover
   - Hamiltonian Cycle is reducible to Invisible Canteloupe

9. Consider the following problem: There are teams of middle-schoolers coming to a big Lego construction extravanza. The site hosting the event has k work areas and a collection U of Lego bricks. Each team has a plan for what they are going to build, and a list of the bricks

they need from U. Naturally, a given brick can't be used simultaneously by two different teams. You'd like to find a collection of k teams that can all work at the same time during the first shift of the event. Can this problem be solved in polynomial time? Justify your answer.

10. Consider the graph G in Figure 1 with edge capacities as indicated.

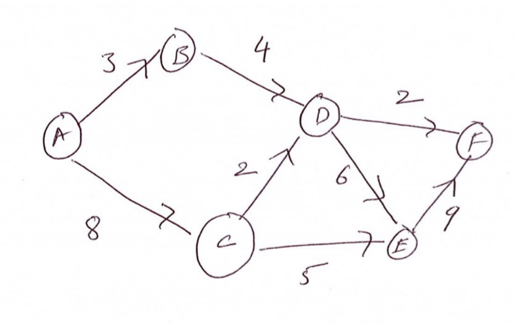

Figure 1: Graph with Flow Capacities

Suppose that we pushed a flow of 4 units in this graph as in Figure 2, where the numbers represent flow values along each edge.

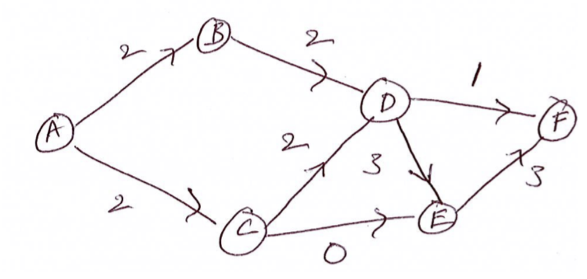

Figure 2: Graph with Flow Values

(a) Draw the residual graph

(b) Find an augmenting path P and in the residual graph

(c) Draw the flow graph obtained by augmenting the flow along P.