Introduction to Algorithms	**Review Questions for Prelim 2**
CS 482 Spring 2000	

The second prelim is *April 11 at 7:30 pm.* It will be closed-book, closed-notes. The prelim is being held in three rooms: Phillips 101, Phillips 203, and Phillips 219.

The questions below are meant to help in studying for the prelim. The questions are not meant to be handed in, and solutions to them will not be handed out. Rather, we will discuss them at the prelim review session in Phillips 101 on Sunday, April 9, at 2:00 pm. You can also discuss them with us in office hours.

The coverage of the prelim is roughly the same as Problem Sets 4-6, and Chapters 5-6 of the book. All handouts up to this point in the course are available in Upson 303.

(1) You are given a graph $G = (V, E)$ with weights w_e on its edges $e \in E$. The weights can be negative or positive. The ZERO-WEIGHT-CYCLE problem is to decide if there is a simple cycle in G so that the sum of the edge weights on this cycle is exactly 0. Prove that this problem is NP-complete.

(2) Since the 3-DIMENSIONAL MATCHING problem is NP-complete, it is natural to expect that the corresponding 4-DIMENSIONAL MATCHING problem is at least as hard. Let us define 4-DIMENSIONAL MATCHING as follows. Given sets W, X, Y, and Z, each of size n, and a collection C of ordered 4-tuples of the form (w_i, x_j, y_k, z_ℓ), do there exist n 4-tuples from C so that no two have an element in common?

Prove that 4-DIMENSIONAL MATCHING is NP-complete.

(3) Some of your friends have recently graduated and started a small company called WebExodus, which they are currently running out of their parents' garages in Santa Clara. They're in the process of porting all their software from an old system to a new, revved-up system; and they're facing the following problem.

They have a collection of n software applications, $\{1, 2, \ldots, n\}$, running on their old system; and they'd like to port some of these to the new system. If they move application i to the new system, they expect a net (monetary) benefit of $b_i \geq 0$. The different software applications interact with one another; if applications i and j have extensive interaction, then the company will incur an expense if they move one of i or j to the new system but not both — let's denote this expense by $x_{ij} \geq 0$.

So if the situation were really this simple, your friends would just port all n applications, achieving a total benefit of $\sum_i b_i$. Unfortunately, there's a problem …

Due to small but fundamental incompatibilities between the two systems, there's no way to port application 1 to the new system; it will have to remain on the old system. Nevertheless, it might still pay off to port some of the other applications, accruing the associated benefit and incurring the expense of the interaction between applications on different systems.

So this is the question they pose to you: which of the remaining applications, if any, should be moved? Give a polynomial-time algorithm to find a set $S \subseteq \{2, 3, \ldots, n\}$ for which the sum of the benefits minus the expenses of moving the applications in S to the new system is maximized.