Assignment #4:

Due Date: June 24

Proofs or counter-examples absolutely necessary for this assignment!

1. Let $G = [V, E]$ be an undirected graph. We want to check if it is connected. The only questions that we are allowed to ask are of the form: "Is there an edge between vertices i and j?". Using an adversary argument show that any correct deterministic algorithm to decide if G is connected must ask $\Omega(n^2)$ questions.

2. Exercise 16.2-7: Suppose you are given two sets A and B, each containing n positive integers. You can choose to reorder each set however you like. After reordering, let a_i be the i^{th} element of set A, and let b_i be the i^{th} element of the set B. You then receive a payoff of $\prod_{i=1}^{n}[(a_i)^{b_i}]$. Give an algorithm that will maximize your payoff. Prove that your algorithm maximizes the payoff, and state its running time.

3. The following problem is known in the literature as the **knapsack problem:** We are given n objects each of which has a weight and a value. Suppose that the weight of object i is w_i and its value is v_i. We have a knapsack that can accommodate a total weight of W. We want to select a subset of the items that yields the maximum total value without exceeding the total weight limit.

   (i)If all v_i are equal, what would the greedy algorithm yield? Is this optimal?

   (ii)If all w_i are equal, what would the greedy algorithm yield? Is this optimal?

   (iii) How should the greedy algorithm be designed in the general case? Is this optimal? [Be careful to distinguish between two versions of the problem: in one we are allowed to select fractional items and in the other we are not allowed to do this.]

4. Consider the following generalization of a scheduling example done in class:

   We have n customers to serve and m identical machines that can be used for this (such as tellers in a bank). The service time required by each customer is known in advance: customer i will require t_i time units ($1 \leq i \leq n$). We want to minimize $\sum_{i=1}^{n} C_i(S)$, where $C_i(S)$ represents the time at which customer i completes service in schedule S . How should the greedy algorithm work in this case? Is it guaranteed to produce optimal solutions?

5. Challenge Problem I: (Challenge Problems: Answer will not be provided!): 16.1-3: (a) Show that repeated Activity Selection does not work; (b) Find another greedy algorithm ; (c)Show that this works.

6. Challenge Problem II: A *celebrity* in a collection G of n people is a person who is known by all other $n-1$ people but who does not know any of them. We are given a collection G of n people and want to know if this collection has a celebrity in it and if one exists to identify the celebrity. We are allowed to ask questions of the form: "Does person A know person B?" for any two persons A and B. We want algorithm that asks minimum number of questions to decide whether the group has a celebrity. Derive a lower bound for the number of questions that need to be asked in the worst case.