Assignment 5

Foundations of Algorithms

1. [20 pts, tree traversal, induction]

   Consider the following algorithm for doing a post-order traversal of a binary tree with root vertex `root`. Prove that this algorithm runs in time $\theta(n)$ when the input is an n-vertex binary tree.

   ```
   def postorder(root):
       if root is not None:
           postorder(root.left)
           postorder(root.right)
           visit(root)
   ```

2. [40 pts, Greedy]

   Suppose you are acting as a consultant for the Port Authority of a small Pacific Rim nation. They are currently doing a multi-billion-dollar business per year, and their revenue is constrained almost entirely by the rate at which they can unload ships that arrive in the port. Here is a basic sort of problem they face. A ship arrives with n containers of weight w_1, w_2, ..., w_n. Standing on the deck is a set of trucks, each of which can hold K units of weight. (You may assume that K and w_i are integers.) You can stack multiple containers in each truck, subject to the weight restrictions of K. The goal is to minimize the number of trucks that are needed to carry all the containers. This problem is NP-complete.

   A greedy algorithm you might use for this is the following. Start with an empty truck and begin piling containers 1, 2, 3, ... onto it until you get to a container that would overflow the weight limit. (These containers might not be sorted by weight.) Now declare this truck "loaded" and send it off. Then continue the process with a fresh truck. By considering trucks one at a time, this algorithm may not achieve the most efficient way to pack the full set of containers into an available collection of trucks.

   a) Give an example of a set of weights and a value for K where this algorithm does not use the minimum number of trucks.

   b) Show that the number of trucks used by this algorithm is within a factor of two of the minimum possible number for any set of weights and any value of K.

3. [40 pts, dynamic programming]

   Consider Longest Increasing Path in a Matrix problem where the goal is finding the longest path that starts from any element of the matrix and follows a path where each consecutive element is greater than the previous. The path can follow left, right, up and bottom elements while watching the border of the matrix. List the recurrence rules and submit a working Python code to implement a dynamic programming algorithm (Hint: use recursion for simplicity). On the right is an example with length 6:

2	1 ↓	1	3
3	2 ⟶	3 ↓	6 ⤒
4	2	4 ⟶	5 ↑
5	1	5	4