ANLY550–Spring, 2019 Homework 5 Out: April 15, 2019
Due: April 26, 2019

For all homework problems where you are asked to give an algorithm, you must prove the correctness of your algorithm and establish the best upper bound that you can give for the running time. You should always write a clear informal description of your algorithm in English. You may also write pseudocode if you feel your informal explanation requires more precision and detail. As always, try to make your answers as clear and concise as possible.

1. Prove that there is a unique minimum spanning tree on any connected undirected graph when the edge weights are unique.

2. Suppose we have an array A containing n numbers, some of which may be negative. We wish to find indices i and j so that
$$\sum_{k=i}^{j} A[k]$$
is maximized. Find an algorithm that runs in time $O(n)$.

3. Suppose we want to print a paragraph neatly on a page. The paragraph consists of words of length $\ell_1, \ell_2, \ldots, \ell_n$. The maximum line length is M. (Assume $\ell_i \leq M$ always.) We define a measure of neatness as follows. The extra space on a line (using one space between words) containing words ℓ_i through ℓ_j is $M - j + i - \sum_{k=i}^{j} \ell_k$. The penalty is the sum over all lines **except the last** of the **cube** of the extra space at the end of the line. This has been proven to be an effective heuristic for neatness in practice. Find a dynamic programming algorithm to determine the neatest way to print a paragraph. Of course you should provide a recursive definition of the value of the optimal solution that motivates your algorithm.

   For this problem, besides explaining/proving your algorithms as for other problems on the set, you should also code up your algorithm in python to print an optimal division of words into lines. Call the program neatness.py. The output should be the text split into lines appropriately, and the numerical value of the penalty. You should assume that a *word* in this context is any contiguous sequence of characters not including blank spaces.

   After coding your algorithm, download the text file containing a review of the Season 1 Buffy DVD posted at `http://people.cs.georgetown.edu/jthaler/BuffyReview.txt`, which was apparently written by Ryan Crackell for the Apollo Guide. Determine the minimal penalty for nearly printing the review, for the cases where $M = 40$ and $M = 72$.

4. Another type of problem often suitable for dynamic programming is problems on tree graphs. For example, suppose we have a graph $G = (V, E)$ that is a tree with a root r. Derive a recursion to find the size of the maximum-sized independent set of of G. (An independent set is a subset of graph vertices, such that no two have an edge between them.) For full credit, show that you can find the size of the maximum-sized independent set and the set itself in linear time.