CS41 Lab 7

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

The goal of this week's exercises is to practice design and analysis techniques with divide-and-conquer and dynamic programming problems.

1. **Divide and conquer minimum spanning trees?**

   Lila has a really cool idea for a divide and conquer algorithm which will find a MST. Given a connected, undirected graph $G = (V, E)$ with weighted edges, Lila's algorithm does the following:

   - Divides the graph into two pieces, $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. ($V_1 \cup V_2 = V$ and V_1 and V_2 are disjoint. E_1 is the edges in E with both endpoints in V_1, and E_2 is the edges in E with both endpoints in V_2.)
   - Recursively finds the MSTs M_1 for G_1 and M_2 for G_2.
   - Finds the lowest-weight edge $e = (u, v)$ with $u \in V_1$ and $v \in V_2$.
   - Returns the minimum spanning tree $M_1 \cup M_2 \cup \{e\}$.

   Unfortunately, this algorithm does not work.

   (a) Give an example input graph G with weights and describe a run of this algorithm where the algorithm does not return a minimum spanning tree on G.

   (b) Is it possible to "patch" this algorithm to work, if the vertex partition is chosen cleverly? That is, can we do a little bit of conquering *before* the divide step(s), which will make this divide-and-conquer MST algorithm work?

   If YES, then describe how to fix this divide and conquer algorithm to be correct. If NO, then argue why no rule for dividing G can make the algorithm correct.

2. **Database Queries** (K&T 5.1)

   You are interested in analyzing some hard-to-obtain data from two separate databases. Each database contains n numerical values (so there are $2n$ values total). You'd like to determine the *median* of this set of $2n$ values, defined as the n-th smallest value.

   The only way you can access these values is through *queries* to the databases. In a single query, you can specify a value k to one of the two databases, and the chosen database will return the k-th smallest value it contains. Since queries are expensive, you would like to compute the median using as few queries as possible.

   Design an algorithm that finds the median value using at most $O(\log n)$ queries. Full pseudocode is not necessary, but you must clearly explain how it works, and you must handle all edge cases; e.g., do not assume that n is even.

3. **Cassie's Convenience Stores.** Cassie plans to open a chain of convenience stores along Baltimore Pike. Using market research, Cassie identified a series of n locations where she can open stores. For each location, Cassie calculated (again using market research) how much annual profit she is likely to gain by placing a store at this location. She can build as many convenience stores as she wants, as long as they are not too close (otherwise, they will compete with each other for business and lose money).

   In this problem, you will design an algorithm that helps Cassie determine how much annual profit she can make. The input to this problem is an integer K, and two arrays $L[1 \cdots n]$ and $P[1 \cdots n]$.

   $L[i]$ represents the the i^{th} potential location for a store, where 0 is the westernmost terminus of Baltimore Pike, and N is the easternmost terminus. $P[i]$ is the profit from placing a store at location $L[i]$. Assume that $0 \leq L[i] \leq N$ for each i, and that L is sorted in increasing order.

   The goal of this problem is to output the maximum possilbe profit by placing convencience stores at locations from $L[1 \cdots n]$ such that the distance between any two locations is at least K.

   (a) If Cassie decides to build a convenience store at location $L[k]$, what is the closest location to the east or west that she can build, given her list of locations? Write an algorithm $West[k]$ that returns the index of the closest location k' to k such that $L[k'] < L[k]$.

   (b) Design a dynamic program that computes the maximum annual profit Cassie can earn by placing her convenience stores.

   (c) Modify your dynamic program so it returns the set of locations that maximize Cassie's profit.