CS41 Homework 7

This homework is due 11am Wednesday November 2. Submit this homework using github. For this homework, you will work with a partner. It’s ok to discuss approaches with others at a high level, but most of your discussions should just be with your partner. The only exception to this rule is work you’ve done with a lab partner while in lab. In this case, note who you’ve worked with and what parts were solved during lab. If there are questions about academic integrity, please visit the section on Academic Integrity on the course website.

0. Before final submission, make sure to fill out the README file.

1. Carrie’s Convenience Stores. Carrie is planning to open a chain of convenience stores along Baltimore Pike. Using market research, Carrie identified a series of locations $L_1, \ldots, L_n$. For each location, she calculated (again using market research) how much profit $P_i$ 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). Write a program to help Carrie determine how much profit she can gain from her chain of convenience stores.

   **Note:** Your program should read from standard input, and write to standard output.

   **Input:** The input consists of three lines. The first line of input contains two integers $N$ and $K$. $N$ is the number of possible store locations that Carrie has identified. $K$ is the minimum distance between stores. The next line contains $N$ integers $1 \leq L_1 < \cdots < L_n \leq 1000000$. $L_i$ represents the location of $L_i$ on Baltimore Pike.\(^1\) The final line contains $N$ positive integers $P_1, \ldots, P_n$. Each $P_i$ is positive and less than 1000 and represents the profit Carrie generates by opening store $i$, assuming it is more than $K$ away from any other store.

   **Output:** The maximum profit Carrie can achieve.

   **Sample Input:**
   
   3 2
   3 4 6
   100 200 102

   **Sample Output:** 202

2. Optimization vs Decision Problems. Recall that a decision problem requires a YES/NO answer, and an optimization problem requires the “best possible answer”, which often means maximizing or minimizing over some cost or score.

   For most optimization problems, there is an obvious analogue as a decision problem. For example, consider the following problem:

   **VC-OPT:** Given a graph $G = (V, E)$, return the size of the smallest vertex cover in $G$.

   **VC-OPT** has a natural decision problem, namely VERTEX-COVER. In fact, every optimization problem can be converted to a decision problem in this way.

---

\(^1\)have you driven all of Baltimore Pike? It is a long road.
(a) Show that \textsc{Vertex-Cover} \leq_p \textsc{VC-Opt}.
(b) Let \( B \) be an arbitrary optimization problem, and let \( A \) be the decision version of \( B \). Show that
\[ A \leq_p B. \]
(c) Show that \( \textsc{VC-Opt} \leq_p \textsc{Vertex-Cover} \).

3. \textbf{Polynomial-time Verifiers}. Recall the definition of a \textit{polynomial-time verifier}: Call \( V \) an efficient \textit{verifier} for a decision problem \( L \) if

(a) \( V \) is a polynomial-time algorithm that takes two inputs \( x \) and \( w \).
(b) There is a polynomial function \( p \) such that for all strings \( x, x \in L \) if and only if there exists \( w \) such that \( |w| \leq p(|x|) \) and \( V(x, w) = \text{yes} \).

Give a polynomial-time verifier for \textsc{Factoring}.

4. \textbf{Memoization vs Tabulation}. (Extra Credit) You’ve seen two different methods of implementing a dynamic program—memoization and tabulation. In practice, which performs better? Is one always a better performer?

Pick a dynamic programming problem we’ve seen already (or select a new one!) and implement the DP both using memoization and tabulation. Next, test your programs on varying inputs of varying sizes. Is one implementation method always better? Is memoization always better on certain kinds of inputs (which ones?) Does the relative performance of memoization vs tabulation depend only on the input size, or are there examples where one technique is better on some inputs, but worse on others? How much better/worse can memoization be vs tabulation?

Include a description of what you implemented, along with any observations/conclusions, in your hw7.tex file.

5. \textbf{Optimization vs Decision Problems}. (Extra Credit) Let \( B \) be an arbitrary optimization problem, and let \( A \) be the decision version of \( B \). Does it \textit{always} hold that
\[ B \leq_p A? \]

Answer \textit{yes} or \textit{no}. Justify your response.