CS 4349.006 Homework 4

Due Thursday, September 30 on eLearning

Please solve the following 2 problems.

1. A palindrome is any string that is exactly the same as its reversal, like I, or DEED, or RACECAR, or AMANAPLANACATACANALPANAMA.

   For the first two parts, our goal is to find the length of the longest subsequence that is also a palindrome of a given string $A[1 .. n]$.

   For example, the longest palindrome subsequence of the string MahDynamicProgrammingShowYouThem is MHLMRMYHM; thus, given that string as input, our algorithm should return 11.

   (a) For any integers $i$ and $j$ such that $1 \leq i \leq n + 1$ and $0 \leq j \leq n$, let $LPS(i, j)$ denote the length of the longest palindrome subsequence of $A[i .. j]$ (recall, we interpret $A[i .. j]$ to be empty if $i > j$). Complete the following recurrence for $LPS(i, j)$ by filling in the blanks.

   $$LPS(i, j) = \begin{cases} & \text{if } i > j \\ & \text{if } i = j \\ + LPS(\ldots, \ldots) & \text{if } i < j \text{ and } A[i] = A[j] \\ \max \{LPS(\ldots, j), LPS(i, \ldots)\} & \text{otherwise} \end{cases}$$

   Advice: In the last case, either $A[i]$ is not the first character or $B[j]$ is not the last character of the longest palindrome subsequence of $A[i .. j]$.

   (b) Describe and analyze a dynamic programming algorithm that fills a 2-dimensional array $LPS[1 .. n + 1, 0 .. n]$ with the solution to each subproblem $LPS(i, j)$ and then returns the length of the longest palindrome subsequence of $A[1 .. n]$.

   (c) Describe and analyze a dynamic programming algorithm that returns the length of the shortest supersequence that is also a palindrome of a given string $A[1 .. n]$. For example, the shortest palindrome supersequence of TWENTYONE is TWENTYOTNEWT, so given the string TWENTYONE as input, your algorithm should return 13.

2. The “new” swap-puzzle game *Candy Swap Saga XV* involves \( n \) cute animals numbered from 1 to \( n \). Each animal holds one of three types of candy: circus peanuts, Heath bars, or Cioccolateria Gardini chocolate truffles. You also have a candy in your hand; at the start of the game, you have a circus peanut.

To earn points, you visit each of the animals in order from 1 to \( n \). For each animal, you can either keep the candy in your hand or exchange it with the candy the animal is holding.

- If you swap your candy for another candy of the *same* type, you earn one point.
- If you swap your candy for a candy of a *different* type, you lose one point.
- If you visit an animal and decide not to swap candy, your score does not change.

You *must* visit the animals in order, and once you visit an animal, you can never visit it again.

Describe and analyze an efficient algorithm to compute your maximum possible score. Your input is an array \( C[1..n] \), where \( C[i] \) is the type of candy that the \( i \)th animal is holding.

*Advice: First, come up with a backtracking strategy. Then, use dynamic programming to make it efficient.*