Problem 1: (30 points) Sequence similarity measure. Let $x$ and $y$ be two given DNA sequences, represented as strings with characters in the set \{A, G, C, T\}. The similarity measure of $x$ and $y$ is defined as the maximum score of any alignment of $x$ and $y$, where the score for an alignment is computed by adding substitution score and deletion and insertion scores, as explained below. (Some operations have negative scores.)

The score for changing a character $x_i$ into a character $y_j$ is $\sigma(x_i, y_j)$, where $\sigma(\cdot, \cdot)$ is defined as follows:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>G</th>
<th>C</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0</td>
<td>-0.1</td>
<td>-0.1</td>
<td>-0.15</td>
</tr>
<tr>
<td>G</td>
<td>1.0</td>
<td>-0.15</td>
<td>-0.1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1.0</td>
<td>-0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

The score for deleting any letter $x_i$ (which we think of as changing $x_i$ into a ")" is $\sigma(x_i, \cdot) = -0.2$.

The score for inserting any $y_j$ is the same. For example, for the strings "AACCTGACATCTT" and "CCAGCGTCAACTT", the alignment

```
   A A C C T G A C - - - - A T C T T
   | | | | | | | | | | | | | | | | |
   C C A G C G T C A A - C T T
```

we will have similarity score

$$-0.2 - 0.2 + 1.0 + 1.0 - 0.15 + 1.0 - 0.2 + 1.0 - 0.2 + 1.0 - 0.2 + 1.0 + 1.0 + 1.0 = 6.25$$

a) (4 points) Let $x = GTGAACGCTGCGGCGTGCT$ and $y = AGCTAATACCCCATACGTTC$. Compute the similarity score and find the alignment of $x$ and $y$, using dynamic programming.

1. Write the recurrence
2. Find the similarity score (by filling a table)
3. Show the alignment (using back-tracking)

b) Develop a program that performs DNA sequence alignment, using dynamic programming.

1. (6 points) Describe your algorithm here (you can copy this description into your .py file as well)
2. (20 points) Develop a program in Python

Input: two DNA sequences;

Output: A) the similarity score, B) Sequence alignment

Note: If you have any questions on your programming assignment please contact Amirali (darvisha@cs.ucr.edu)

The specs for the programming assignment will be posted separately.

Problem 2: (10 points) Consider the following game. A "dealer" produces a sequence $s_1, \ldots, s_n$ of "cards," face up, where each card $s_i$ has a value $v_i$. Then two players take turns picking a card from the sequence, but can only pick the first or the last card of the (remaining) sequence. The goal is to collect cards of largest total value. (For example, you can think of the cards as bills of different denominations.) Assume $n$ is even.

a) Show a sequence of cards such that it is not optimal for the first player to start by picking up the available card of larger value. That is, the natural greedy strategy is suboptimal.

b) Give an $O(n^2)$ algorithm to compute an optimal strategy for the first player. Given the initial sequence, your algorithm should precompute in $O(n^2)$ time some information, and then the first player should be able to make each move optimally in $O(1)$ time by looking up the precomputed information.
**Problem 3: (10 points) EXTRA CREDIT**

Assume that you are given two arrays \( A = \{a_1, a_2, \ldots, a_n\} \) and \( B = \{b_1, b_2, \ldots, b_n\} \) of \( n \) real numbers.

1. Describe a greedy algorithm to determine an ordering of the elements of \( A \) and \( B \) such that \( W = \sum_{i=1}^{n} |a_i - b_i| \) is minimized.

2. Analyze the time complexity of your algorithm.

3. State and prove the greedy-choice property of your algorithm.

4. State and prove the optimal substructure property of your algorithm.

**Hint:** Prove the following fact. Given real numbers \( x_1 \leq x_2 \) and \( y_1 \leq y_2 \), then

\[
|x_1 - y_1| + |x_2 - y_2| \leq |x_1 - y_2| + |x_2 - y_1|
\]