The Following Problems are from textbook:

**Problem 6.21**
For a pair of strings \( v = v_1 \ldots v_n \) and \( w = w_1 \ldots w_m \), define \( M(v, w) \) to be the matrix whose \((i, j)\)th entry is the score of the optimal global alignment which aligns the character \( v_i \) with the character \( w_j \). Give an \( O(nm) \) algorithm which computes \( M(v, w) \).

**Problem 6.22** Define an overlap alignment between two sequences \( v = v_1 \ldots v_n \) and \( w = w_1 \ldots w_m \) to be an alignment between a suffix of \( v \) and a prefix of \( w \). For example, if \( v = \text{TATATA} \) and \( w = \text{AAATTT} \), then a (not necessarily optimal) overlap alignment between \( v \) and \( w \) is

\[
\begin{align*}
\text{ATA} \\
\text{AAA}
\end{align*}
\]

Optimal overlap alignment is an alignment that maximizes the global alignment score between \( v_1, \ldots, v_n \) and \( w_1, \ldots, w_j \), where the maximum is taken over all suffixes \( v_1, \ldots, v_n \) of \( v \) and all prefixes \( w_1, \ldots, w_j \) of \( w \).

Give an algorithm which computes the optimal overlap alignment, and runs in time \( O(nm) \).

**Problem 6.24** We have studied two approaches to sequence alignment: global and local alignment. There is a middle ground: an approach known as semiglobal alignment. In semiglobal alignment, the entire sequences are aligned (as in global alignment). What makes it semiglobal is that the “internal gaps” of the alignment are counted, but the “gaps on the end” are not. For example, consider the following two alternative alignments:

- **Sequence 1:** \( \text{CAGCA--CTTGATTCTCGG} \)
- **Sequence 2:** \( ---\text{CAGCGTGG}------- \)

- **Sequence 1:** \( \text{CAGC\text{ACTTGGATTCGCG}} \)
- **Sequence 2:** \( \text{CAGC}-----\text{G-T----GG} \)

The first alignment has 6 matches, 1 mismatch, and 12 gaps. The second alignment has 8 matches, no mismatches, and 10 gaps. Using the simplest scoring scheme (+1 match, -1 mismatch, 0 gap):
-1 gap), the score for the first alignment is -7, and the score for the second alignment is -2, so we would prefer the second alignment. However, the first alignment is more biologically realistic. To get an algorithm which prefers the first alignment to the second, we can not count the gaps “on the ends”.

Under this new (“semiglobal”) approach, the first alignment would have 6 matches, 1 mismatch, and 1 gap, while the second alignment would still have 8 matches, no mismatches, and 10 gaps. Now the first alignment would have a score of 4, and the second alignment would have a score of -2, so the first alignment would have a better score.

Note the similarities and the differences between the Fitting problem and the Semiglobal Alignment problem as illustrated by the semiglobal but not fitting alignment of ACGTCAT against TCATGCA:

Sequence 1: ACGTCAT---
Sequence 2: ---TCATGCA

Devise an efficient algorithm for the Semiglobal Alignment problem and illustrate its work on the sequences ACAGATA and AGT. For scoring, use the match premium +1, mismatch penalty -1, and indel penalty -1.

Problem 6.31 Given two strings $v_1$ and $v_2$ and a text $w$, find whether there is an occurrence of $v_1$ and $v_2$ interwoven (without spaces) in $w$. For example, the strings abac and bbc occur interwoven in cabbabccdw. Give an efficient algorithm for this problem.

Problem 6.32
A string $x$ is called a supersequence of a string $v$ if $v$ is a subsequence of $x$. For example, ABLUE is a supersequence for BLUE and ABLE.

Given strings $v$ and $w$, devise an algorithm to find the shortest supersequence for both $v$ and $w$. 