Approach:

Our approach employs a dynamic programming matrix to systematically calculate the optimal alignment score between

two sequences. This matrix uses a predefined scoring matrix to determine the scores for matches, mismatches, and

gaps. The algorithm initializes the first row and column based on gap penalties and then iteratively fills in the rest of the

matrix. A traceback procedure is then followed to reconstruct the alignment from the computed scores.

Time Analysis:

The time complexity of the dynamic programming approach to sequence alignment is O(mn), where m and n are the

lengths of the two sequences being aligned. This quadratic time complexity arises because each cell in the m-by-n

matrix must be filled, and each cell requires a constant amount of time to compute. The algorithm is efficient for small to

medium-sized sequences but can become computationally intensive for very long sequences.

Conclusion:

In our implementation, the dynamic programming algorithm achieved an approximate score slightly above the expected

value, which can be attributed to the nuances of floating-point arithmetic in Python. This discrepancy highlights the

importance of considering computational precision in algorithmic design and analysis.