Implementation of Sequence Alignment Algorithms

1 Introduction to the Assignment

In this assignment, you are required to implement two sequence alignment algorithms: Needleman-Wunsch (NW) for global alignment and Smith-Waterman (SW) for local alignment. Both algorithms will use a single *substitution matrix* that includes match, mismatch, and gap penalties.

DNAfull Substitution Matrix

The following matrix, will be used for both global and local alignment tasks:

	A	G	C	T
A	5	-4	-4	-1
G	-4	5	-4	-1
C	-4	-4	5	-1
T	-1	-1	-1	5

In this matrix:

- **Match**: A match between identical nucleotides (e.g., A with A) is awarded $+5$ points.
- **Mismatch**: A mismatch between different nucleotides (e.g., A with G) is penalized by either -4 or -1, depending on the pair.
- **Gap introduction**: Insertion an extension of a gap ('-') incurs a penalty of -2.

This matrix will be used for both global alignment (Needleman-Wunsch) and local alignment (Smith-Waterman).

2 Part 1: Implementation of Needleman-Wunsch Algorithm

In this part of the assignment, you are required to implement the Needleman-Wunsch algorithm, which performs global alignment of two DNA sequences. The algorithm should correctly fill the dynamic programming matrix based on the provided substitution matrix, find the n optimal alignments, print them and save to an output file in the following format:

```
Global alignment no. 1:
-TATA
ATAT-
Score: X

Global alignment no. 2:
TATA-
-ATAT
Score: X

etc...
```

Input parameters:

- n <int> (maximum optimal alignments)
- filepath to the substitution matrix in CSV format <str>
- GAP penalty (default -2) <int>
- output filename <str>

3 Part 2: Implementation of Smith-Waterman Algorithm

In this part, you are required to implement the Smith-Waterman algorithm, which performs local alignment of DNA sequences. The algorithm should also use the *DNAfull substitution matrix*.

Grading (Total 10 Points)

- 2 points for correctly filling the Needleman-Wunsch dynamic programming matrix.
- 2 points for finding all possible paths up to N alignments in NW.
- 2 points for adapting the algorithm to Smith-Waterman.
- 2 points for clean code (readability, modularity, comments, adherence to SOLID principles) and repository maintenance (e.g., Git).
- 2 points for unit tests verifying the correctness of the algorithm.