Abstract

This assignment uses the idea of dynamic programming and the Smith Waterman algorithm to match sequencies recursively. Here we had two different Sequence, sequence X and sequence Y. In this assignment two different Smith Waterman algorithms were implemented. The first algorithm is a bottom-up version of the Smith Waterman algorithm, that used the function definition of M. The second algorithm implemented is the top-down Smith Waterman algorithm with memorization, which also used the recursive definition of M. In order to print the sequence alignment for X and Y, two functions were implemented Print Seq Align X and Print Seq Align Y, which helped printing out the Sequence Align prints. Additional code was also added to the main function to print out the H and P tables for testing purposes as well as demonstrating question 4 in the assignment that references Executing the pseudocode algorithm and to fill in the necessary tables H and P, in a bottom-up fashion.

Analysis

In this assignment we were not asked to perform multiple tests to analyze the running time of both algorithms, but I still wanted to point out that the Smith Waterman performs at a better running time then a naïve approach of doing sequence comparison. The Smith Waterman algorithm has a running time of O(nm), whereas a naïve approach would have a running time of O open parentheses n to the power of 3 to the power of blank end exponent m cubed close parentheses. When implementing both algorithms I also noticed that they do follow the dynamic programming framework, where the top down with memorization remembers the values already computed and does not continuously re-evaluate the same subproblems.

Conclusion

I can see how these Algorithms are a good way for gene-sequence and how the difference between bottom-up and memorization that shows with the use of dynamic programming, there is a way to save how often we execute a subproblem.

Question 4

For this question I wrote some additional code to show the maximum alignment for X= dcdcbacbb and Y= acdccabdbb by using the Smith Waterman Algorithm. My output of the H and P tables are as follows:

A picture containing background pattern

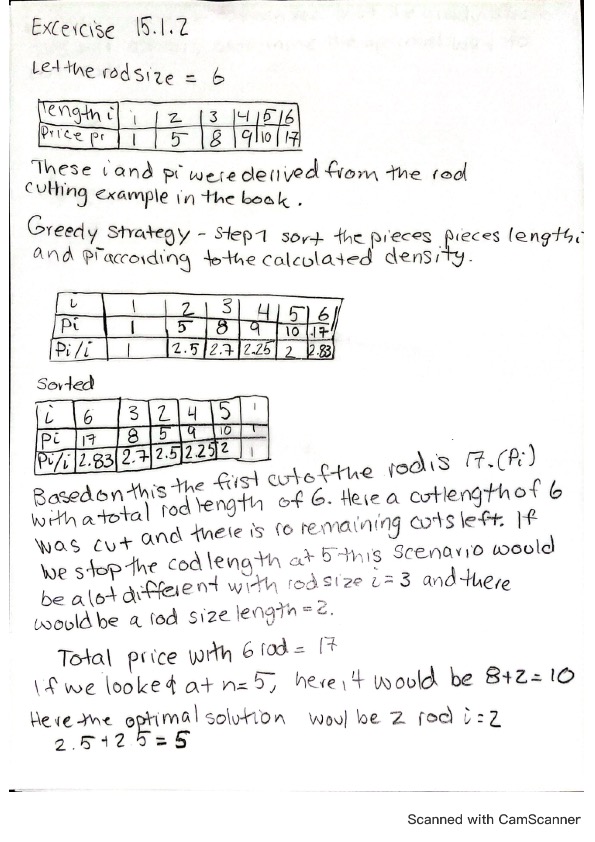
Description automatically generated

The following code in the main file was added to be able to add the tables, this needs to be done by commented out the generator first.

Text

Description automatically generated

Exercises



Text, letter

Description automatically generated

Text, letter

Description automatically generated

A close-up of a key chain

Description automatically generated with low confidence

A picture containing diagram

Description automatically generated