Description of the problem and algorithm

[A-1]

To find unique characters in a sentence, it must be searched to find if there is overlap. I used Hash table to search and record characters and to detect duplicate characters.

Without any data structure, I used ASCII code of characters. Each character has a unique ASCII code value. Therefore, if characters are converted to each unique ASCII code, all characters can be represented in numbers. After that, I used a bit operator. I was able to figure out if bits are overlapped using shift operator as much as the ASCII value of the text. Overlapped bits will mean overlapped characters.

[b-1]

Brute-force

In brute-force algorithm, I identified all sub-matrix present in the matrix. The operation of h^2\*w^2 was used to determine all sub-matrix in existence, and the operation of h\*w was used to obtain the sum of sub-matrix.

Divide and conquer

In the Divide and Conquer algorithm, I divided the problem into four categories. It is upper left, upper right, lower left, lower right.

In the figure, the upper left square is recursively called. The upper right square can be solved by a divide and conquery method that looks for leftMax, rightMax, and crossMax, as shown in sub-array problems for sum of the upper left square and right square. The same is true of the lower left square. The lower right square is used to obtain sub-matrix containing the diagonal from the entire square.

Dynamic-programming

The dynamic programming uses the memoization technique. Prepare memory\_matrix (n\* n) to record the cumulative sum of matrix elements. Check all the elements within the initial matrix and record the cumulative sum up to that element in memory\_matrix. Sub-matrix sum can then be obtained by using the subtraction with memory\_matrix.