The purpose of this code is to find the minimum length between two sequence bands, and we declaim three vice functions to implement the result, which is "euclidean\_distance(point1, point2)", "valid\_band(P, Q, band\_length, output\_file\_path)", and "min\_band\_length(input\_file\_path, output\_file\_path)".

First, the function of "Euclidean\_distance (point1, point2)" is to calculate the distance between two points, assuming (x, y) in tuples so that we can calculate the distance by Euclidean formula: [(X1 – X2)2+(Y1 – Y2)2]1/2 . for "valid\_band(P, Q, band\_length)", this vice function is to check whether the band length between two sequence points ("P", "Q") is valid or not, and we use dynamic programming to reduce the running time by storing the band length between "P" and "Q" so that we don't have to keep calculate the same formula iteratively. we use the 2D array to store the length and check if the Euclidean distance between related points satisfies the specified band length. And for "min\_band\_length(input\_file, output\_file\_path)", we read input from file, and parse the input to extract the sequences "P" and "Q", and band length "L". then we sort the band lengths, and we use binary search to find the minimum length("valid\_band"), return to the output file.

Algorithm and Time Complexity Analysis:

In this program, we use "The Merge Sort algorithm", "Dynamic Programming" and "Binary search" to maximum reduce the time complexity. We have described the merge sort in the previous assignment, so we start with dynamic programming, the reason why we use this is that we use the nested loop to find out all the possible lengths between sequences P and Q, we want to reduce the time complexity, so we can use the dynamic programming to stop early when we know the following length is not in valid band length. We allocated a 2D array to set the boundary condition as the valid band length, we judge the valid while computing the possible length, similar to the memorization, using memory to save time. Binary also is an algorithm to calculate the minimum value, it helps search more linear, but it only works while the array is sorted, so we have to sort before we move on. However,

we have to change the binary tree every time we find a new possible length, so we have to call this function in the nested loop.

The time complexity of calculating the distance between two points is always one step, so it's O(1). to analyze the time complexity of "valid band", we use nest loop iterating through sequences "P" and "Q", and the maximum index value is "m" and "n", so it comes to O(m \* n). Be careful, we have to do the binary search for every possible tuple (m, n), so the time complexity is O(m \* n \* log t), Because we use the binary search, we have to sort the array "L", whose maximum index value is t, so the time complexity is O(t log t).

Integral these descriptions, the mathematic model of time complexity will be O(t log t + m \* n \* log t), where m is the length of sequence P, n is the length of sequence Q, and t is the number of band lengths.