# Abstract

In this phase, we solve the TSP given in the assignment manual with the DFS algorithm. The algorithm uses the distance table reference and the problem branches pruning to reduce the search time. The proposed algorithm is implemented in C++.

# Algorithm

At first, we used the brute-force algorithm. The permutation was applied to the array of the cities, the path was determined for all cases, and the minimum distance was determined. This was quite inefficient. If the minimum distance has already been exceeded by current path, the arrangement of the city is not required to be inspected. The search was performed only when the conditions were satisfied. This is called backtracking algorithm. By applying this algorithm, the time could be further reduced.

There was also one drawback in this algorithm. Even though backtracking was done, it was necessary to calculate the corresponding length each time. The length calculation was composed of a fairly complex expression such as root and square. So at the end of the meeting, we came up with one measure. If we created a matrix that stores distances between cities before we start the search, and then calculate the path using only the constant values already stored, we do not have to repeat the complex operations. As a result of applying this algorithm, we could find the solution within the most optimized time.

The final algorithm we used is as follows. First, use the double for statement to store the distance between cities. In this case, it is stored in the form of triangular matrix to prevent unnecessary calculation. Then start DFS(Depth First Search). If the number of visited cities reaches seven and the distance is less than the current minimum value, the distance is stored as the minimum value and the order of the searched city is also stored. When searching, check the backtracking condition before going to the next city, and search the next city only when the condition is satisfied. When all the searches are completed, the final path and length are printed and the program is terminated.

# Flow Chart

# Source Code

# Output

(present the path and the distance traveled)

# Conclusion