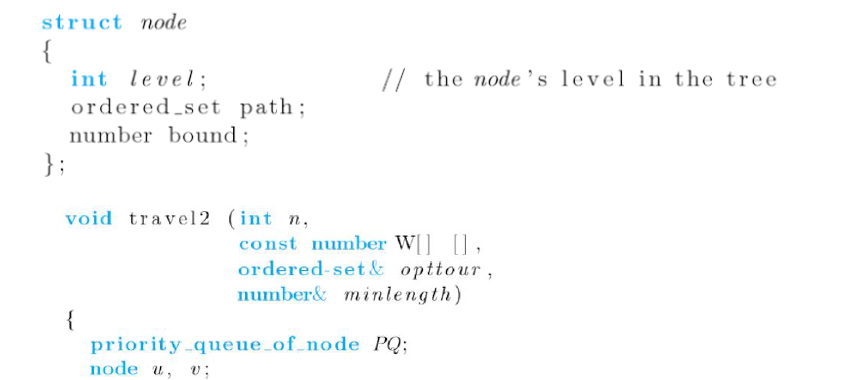
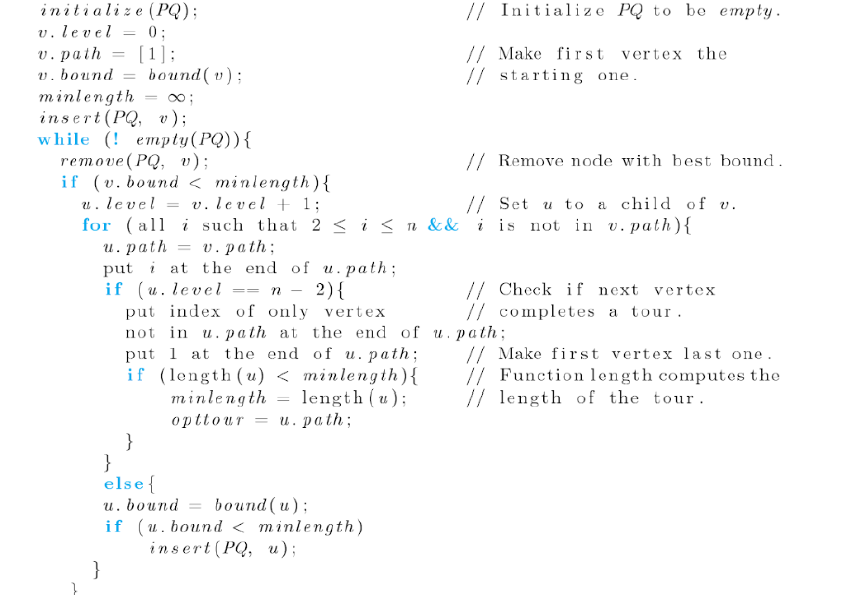
1. Pseudo Code

My pseudo code is the same as the professor’s pseudo code, I just followed what the lecture told and tried to implement it.



1. Flow chart
2. Initialize an array for the tour, with all of the data being 0;
3. Initialize a visited array to store the visited, array[i] ‘s value will be the position in the tour.
4. Calculate the bound for the root
5. Push the root into the priority queue
6. Starting from the priority queue, visit all nodes using the adjacency matrix
7. Calculate the bound for each node we visit
8. Starting with the node with the smallest bound, we go deeper and repetitively calculate the bound.
9. Update the Min\_path\_cost when hitting a leaf
10. If another branch has higher bound than current Min\_path\_cost, we prune it
11. Repeat the process until obtaining the best tour.
12. Time complexity analysis

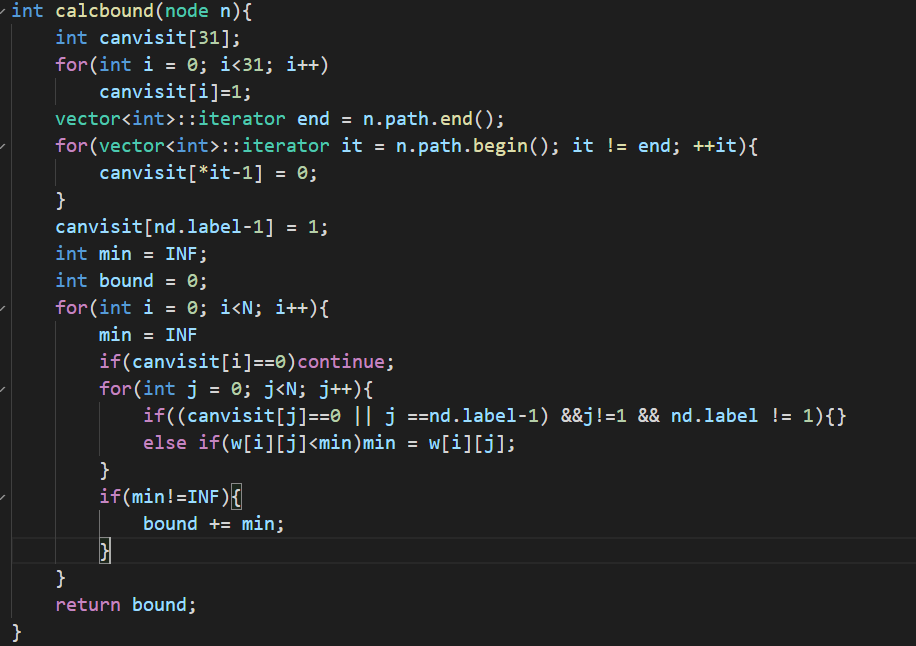
Suppose we have N cities left unvisited (including the root), we have to create all possible extensions for the unvisited cites which are N-1 (excluding the root).

Following this thought, the complexity for generating the permutation is

,

which is equal to

However, for each node, we have to calculate the cost, following the method below.



Since there is a double for loop inside the function, which iterates through the N cities. This generates a time complexity of

So, combining the above, the total time complexity should be