PA3 Report　　B11901040 項達均

The graph is represented in 2 data structures:

- An adjacency list implemented with the std::map data structure. This is used during DFS to find neighbors of a vertex in $O(1)$ time.
- A vector of a defined struct Edge, which stores the two vertices and the weight. This is used in sorting edge weights.

Finding an connected acyclic graph with least cost on removed edges is equivalent to finding a acyclic connected graph with the max weight cost. Therefore for undirected graph, I used Kruskal's algorithm to find the maximum spanning tree. The maximum spanning tree is implemented using the disjoint set data structure with union by rank and path compression.

For directed graphs, I first created a MST using Kruskal's algorithm, then for all edges with positive weight, put them back if they do not create a cycle. We can check if a cycle exists by running DFS from v to u.