PA3 report

系級:電機三　姓名:金家逸 學號: B10502076

Algorithm:

Undirected graph:

1. Find the maximum spanning tree of the graph, then remove the edges not in the maximum spanning tree. To find the maximum tree, multiply each edge's weight in the graph by -1. The problem is then converted to finding the minimum spanning tree.
2. Kruskal's algorithm is employed for this purpose, given that the edge weights are in the range of -100 to 100. Counting sort is used to sort the edge weights in linear time at the beginning of the Kruskal algorithm.
3. Running time complexity:
   $O(E+k)+O\big((V+E)\alpha(V+E)\big)\approx O(E+201)$, which is polynomial time. This is because the weight of each edge is within the range -100 to 100, where k = 201 and $\alpha(V+E)$ is small, considered as constant time.

Directed graph:

1. First, treat the graph as an undirected graph and find the maximum spanning tree.
2. For the removal of edges from the maximum spanning tree, if the edge weight is positive or equal to zero, add the edge back to the maximum spanning tree in non-decreasing order. Then, check for the existence of a cycle. If a cycle is found, remove the edge that was added back to the graph. If the edge weight is negative, always remove the edge to achieve the minimum total removal weight since the maximum spanning tree ensures the new graph is connected.

3. The method to check for the existence of a cycle after adding an edge back to the subgraph is DFS. If the next vertex is the visited vertex (with the color of the vertex being grey), then a cycle exists.

Data structure:

1. Disjoint set: Used to find the maximum spanning tree with union by rank and path compression for find-set.
2. Adjacency list: Implemented using vectors to check for the existence of a cycle through DFS.
3. Dynamic array: Used to store information about each edge, including the start vertex, end vertex, and weight.
4. The final removal edges are stored in a vector.