Report for PA3 (Cycle Breaking Problem)

To reach minimum removed edges, I switch the sign of weight for each edges. For example, 1 -> -1 , -2 -> 2, and then switch back before output . Therefore, I can solve the problem like normal minimum spanning tree problem. For undirected case, I simply use Kruskal's Algorithm. First, I use make_set function to make each vertex in the graph an independent set. I also sort the edges according to their weights by heap sort. The start points and end points of each edges will move together with the weights because it will be easier if I want to get the start points and end points of each edges after heap sort. What's more, the idea of disjoint set is implemented by a single vector. Vertices are the indexes of the vector. The value of the vector is the "head" of the corresponding set. All the value of the vertices which are in the same set are the same, it is the "head" of the set. If I want to union two sets, I just have to change the value in the vector. After that , for each edge taken in nondecreasing order by weight, if the set of the start point of the edge is different from the set of the end point of the edge. I union them together and pick the edge as a part of minimum spanning tree. Otherwise, I won't pick the edge (i.e. the edge is removed).

For directed case, I use Kruskal first. After that, for all removed edges, if the weight <= 0, we do nothing. If the weight is bigger than 0, we add it back to the MST decreasingly and use DFS to check whether there is a cycle. If no, the edge is added successfully into MST, so the total weights of the removed edges decrease. If yes, the edge is still removed.

I found out that fout seems to cost a lot of time. When I use fout to debug , even with a simple loop, it will significantly increase the execution time. I also found out that Kruskal is a great algorithm because it is quite easy to implement. The idea of disjoint set can be simply implemented with a vector. That's the reason why I choose Kruskal rather than Prim to deal with minimum spanning tree.