IT251 Assignment-6

NAME: SUYASH CHINTAWAR

ROLL NO.: 191IT109

TOPIC: KRUSKAL’S ALGORITHM

VARIABLES:

- **n** – number of vertices in the graph
- **edges** – number of edges in the graph
- **v1, v2, w** – two vertices in an edge and the corresponding edge weight
- **q** – priority queue of pair of int and pair to store weight and vertices in an edge
- **n_mst** – number of edges currently in MST
- **mst_cost** – cost of MST
- **parent** – vector containing root of vertices in a tree in the forest
- **size** – vector having size of tree in a forest
- **mst** – to store mst edges and their weights

FUNCTIONS:

- **kruskal()** – implement Kruskal's Algorithm
- **make_set()** – Make a tree with one vertex and initializing the root as the vertex itself and size of tree as 1.
- **find_set()** – Find the root of tree of which the vertex belongs to
- **union_sets()** – Perform union operation if two vertices belongs to different trees by joining smaller tree to larger one

README and Assumptions:

The program implements Kruskal's Algorithm on undirected graphs by using priority queue. Union-find(DSU) data structure has been used to find whether two vertices belong to the same component in the forest at a time instance.

- The undirected graph edges are stored in a priority queue **q** as {weight, {v1, v2}} where there is an edge between v1 and v2.
- **kruskal()** is called which then initializes the trees with one vertex in each tree forming a forest. The trees are made by calling **make_set()** function which assigns the root as the vertex itself and size of tree as 1.
- We then pop elements one by one from the top of the priority queue (edges which will have smallest weights are in the top), and then check whether this chosen edge forms a cycle in the forest. If adding this edge forms cycle, then we discard this edge else we add it to our MST and add the corresponding edge weight to total cost of MST.
- To check whether adding an edge form a cycle or not, we call **find_set(v)** to get the root of the tree which contains vertex v. If both the vertices of the edge belong to the same tree then its discarded.
- If the vertices belong to different trees, then we combine those trees by calling **union_sets(v1,v2)** which joins the smaller tree to the bigger one(to eliminate the possibility of chaining after union).

- Now, if the number of edges in out MST gets equal to n-1, then we break through the loop and print our output to console. Otherwise, if the queue becomes empty and we still haven't accepted n-1 edges, then it is clear that the original graph is disconnected. Hence, MST cannot be formed.
- The optimized version of Union-find data structure has been used which gives us complexity near to O(1), by using **union by size** and modifying traditional **find_set()** methods.

Time complexity of the whole algorithm is O(ElogV), where E is number of edges and V, number of vertices.

THANK YOU