Batch:SY-IT(B3) **Experiment Number:7**

Roll Number: 16010423076 **Name:Ritesh Jha**

Aim of the Experiment: To study implementation of Spanning tree

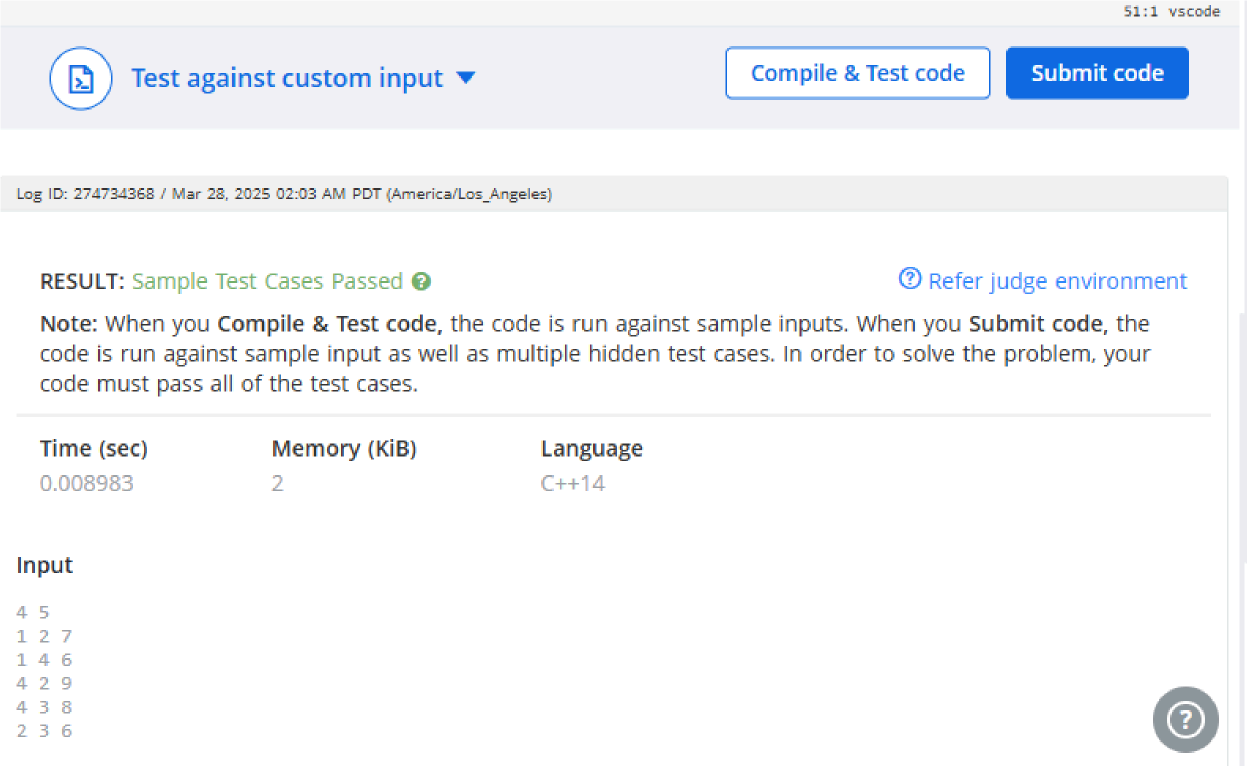
Program/ Steps:

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

struct Edge {
   int u, v, w;
};
int findParent(int node, vector<int>& parent) {
   if (parent[node] != node)
       parent[node] = findParent(parent[node], parent);
   return parent[node];
}
int main() {
   int N, M;
   cin >> N >> M;

   vector<Edge> edges(M);
   for (int i = 0; i < M; i++)
       cin >> edges[i].u >> edges[i].v >> edges[i].w;
```

```
// Sort edges by weight
sort(edges.begin(), edges.end(), [](Edge a, Edge b) {
    return a.w < b.w;
});
// Disjoint Set Initialization
vector<int> parent(N + 1);
for (int i = 1; i <= N; i++) parent[i] = i;

int mstWeight = 0, edgesUsed = 0;

for (auto edge : edges) {
    int pu = findParent(edge.u, parent);
    int pv = findParent(edge.v, parent);
    if (pu != pv) { // If not in the same component, include in MST
        parent[pu] = pv;
        mstWeight += edge.w;
        edgesUsed++;
        if (edgesUsed == N - 1) break; // MST complete
    }
}

cout << mstWeight << endl;
return 0;
```

```
}
```

Output/Result:



Log ID: 274734368 / Mar 28, 2025 02:03 AM PDT (America/Los_Angeles)

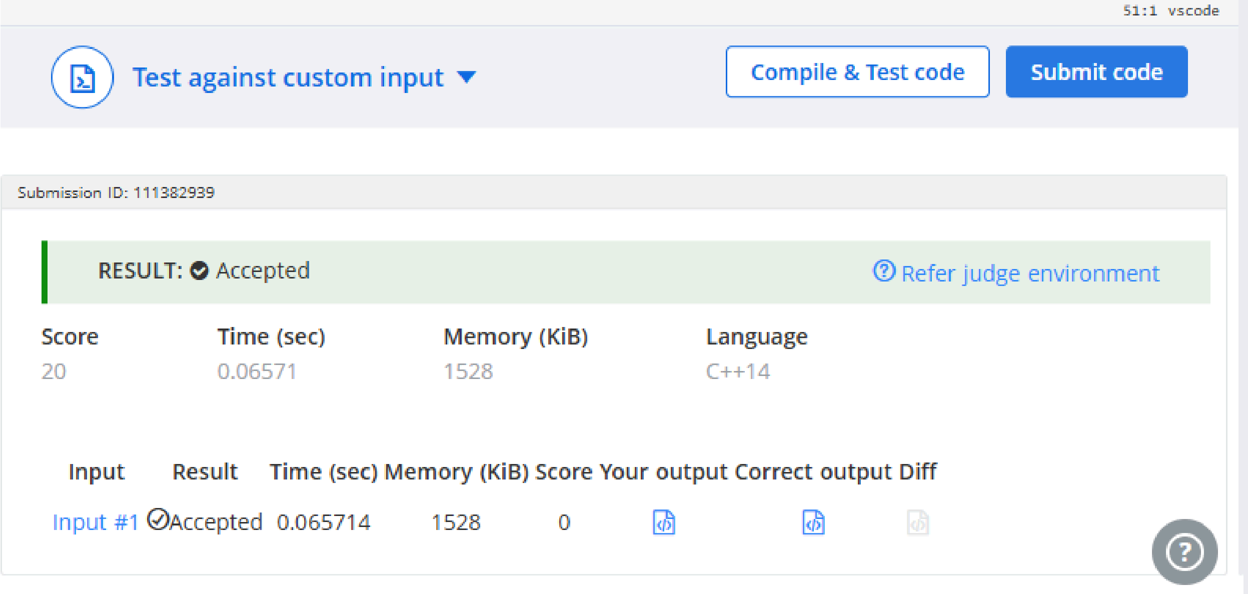
RESULT: Sample Test Cases Passed

Note: When you **Compile & Test code,** the code is run against sample inputs. When you **Submit code,** the code is run against sample input as well as multiple hidden test cases. In order to solve the problem, your code must pass all of the test cases.

Time (sec)	Memory (KiB)	Language
0.008983	2	C++14

Input

```
4 5
1 2 7
1 4 6
4 2 9
4 3 8
2 3 6
```



Submission ID: 111382939

RESULT: Accepted

Score	Time (sec)	Memory (KiB)	Language
20	0.06571	1528	C++14

Input	Result	Time (sec)	Memory (KiB)	Score	Your output	Correct output	Diff
Input #1	✓Accepted	0.065714	1528	0			

Outcomes:

CO3 . Understand the Graphs, related algorithms, efficient implementation of those algorithms and applications

Conclusion (based on the Results and outcomes achieved):

From this experiment, I learned how to implement a Minimum Spanning Tree (MST) using Kruskal’s Algorithm efficiently. By utilizing the Disjoint Set Union (DSU) data structure and sorting edges by weight, I was able to construct the MST and compute its total weight. This experiment reinforced my understanding of graph algorithms, their practical applications, and the importance of efficient implementation in competitive programming.

References:

1. https://www.hackerearth.com/practice/algorithms/graphs/minimum-spanning-tree/practice-problems/algorithm/minimum-spanning-tree-5/
2. T.H. Coreman ,C.E. Leiserson,R.L. Rivest, and C. Stein, &quot; Introduction to algorithms&quot;,
3rd Edition 2009, Prentice Hall India Publication
3. Antti Laaksonen, “Guide to Competitive Programming”,Springer,2018
4. Gayle Laakmann McDowell,” Cracking the Coding Interview”,CareerCup LLC,2015
5. Steven S. Skiena Miguel A. Revilla,”Programming challenges, The Programming Contest Training Manual”, Springer, 2006
6. Antti Laaksonen, “Competitive Programmer’s Handbook”, Hand book, 2018
7. Steven Halim and Felix Halim, “Competitive Programming 3: The Lower Bounds of Programming Contests”, Handbook for ACM ICPC