prims.c++

```cpp
#include <bits/stdc++.h>
#define INFI 99
using namespace std;

int matrix[10][10];

//Function to find sum of weights of edges of the Minimum Spanning Tree.
void prims(int V)
{
    priority_queue<
        pair<pair<int, int>,int>,
        vector<pair<pair<int, int>,int>>,
        greater<pair<pair<int, int>,int>>
    > pq;

    int vis[V+1] {0};
    vector<pair<int,int>> mst;
    int cost = 0;
    // {{wt, node}, parent}
    pq.push({{0, 1}, -1});

    while (!pq.empty()) {
        auto it = pq.top();
        pq.pop();
        int node = it.first.second;
        int wt = it.first.first;
        int parent = it.second;

        if (vis[node]) continue;
        mst.push_back({parent,node});
        vis[node] = 1;
        cost += wt;

        for (int i = 1; i <= V; i++) {
            int edW = matrix[node][i];
            if (edW != 0 && edW != INFI && !vis[i]) {
                pq.push({{edW, i},node});
            }
        }
    }
    mst.erase(mst.begin());

    cout << "The edges of minimum spanning tree:\n";
    for (auto it : mst)
        cout << "(" << it.first << ", " << it.second << ")\n";
    cout << "The cost of minimum spanning tree:\n" << cost << endl;
}

int main() {
    cout << "Enter the number of vertices:\n";
    int V;
```

```
    cin >> V;

    cout << "Enter the adjacency matrix:\n";
    for(int i = 1; i <= V; i++)
        for(int j = 1; j <= V; j++)
            cin >> matrix[i][j];

    prims(V);
    return 0;
}
```