```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;
struct Edge {
    int from;
    int to;
    int weight;
};


vector<Edge> boruvkaMST(vector<Edge>& edges, int n) {
    vector<Edge> mst;
    vector<int> parent(n);
    vector<int> cheapest(n, -1);
    vector<int> component(n);
    for (int i = 0; i < n; i++) {
        parent[i] = i;
        component[i] = i;
    }
    int numComponents = n;
    while (numComponents > 1) {
        for (int i = 0; i < n; i++) {
            cheapest[i] = -1;
        }
        for (auto& edge : edges) {
            int u = edge.from;
            int v = edge.to;
            int parentU = parent[u];
            int parentV = parent[v];
            if (parentU != parentV) {
                if (cheapest[parentU] == -1 || edge.weight < edges[cheapest[parentU]].weight) {
                    cheapest[parentU] = edge.weight;
                }
                if (cheapest[parentV] == -1 || edge.weight < edges[cheapest[parentV]].weight) {
                    cheapest[parentV] = edge.weight;
                }
            }
```

```
40          }
41
42
43          for (int i = 0; i < n; i++) {
44              if (cheapest[i] != -1) {
45                  int u = i;
46                  int v = component[edges[cheapest[i]].
   from] == i ? edges[cheapest[i]].to : edges[cheapest[i
   ]].from;
47                  int parentU = parent[u];
48                  int parentV = parent[v];
49                  if (parentU != parentV) {
50                      mst.push_back(edges[cheapest[i
   ]]);
51                      numComponents--;
52                      for (int j = 0; j < n; j++) {
53                          if (parent[j] == parentV) {
54                              parent[j] = parentU;
55                              component[j] = i;
56                          }}
57                  }
58              }}}
59      return mst;
60  }
61
62  void printEdges(vector<Edge>& edges) {
63      for (auto& edge : edges) {
64          cout<<edge.from <<" - "<< edge.weight <<" - "
   << edge.to << endl;
65      }  }
66
67  int main() {
68      vector<Edge> edges = {
69              {0, 1, 4},
70              {0, 7, 8},
71              {1, 2, 8},
72              {1, 7, 11},
73              {2, 3, 7},
74              {2, 5, 4},
75              {2, 8, 2},
76              {3, 4, 9},
```

```
77              {3, 5, 14},
78              {4, 5, 10},
79              {5, 6, 2},
80              {6, 7, 1},
81              {6, 8, 6},
82              {7, 8, 7}
83      };
84      vector<Edge> mst = boruvkaMST(edges, 9);
85      cout<<"minimum spanning tree is:"<<endl;
86      printEdges(mst);
87
88      return 0;
89  }
90
```