main.cpp

Share Run

```cpp
#include <iostream>
#include <unordered_map>
#include <vector>
#include <string>
using namespace std;
struct Edge {
    string destination;
    double flowRate;
    Edge(string dest, double flow) : destination(dest), flowRate
        (flow) {}
};
class WaterNetwork {
private:
    unordered_map<string, vector<Edge>> graph;
    const double FLOW_THRESHOLD = 10.0;
public:
    void addNode(const string& node) {
        graph[node];
    }
    void addPipe(const string& from, const string& to, double
        flowRate) {
        graph[from].emplace_back(to, flowRate);
    }
    void displayNetwork() {
```

Output Clear

```
Water Supply Network:
House1 ->
Junction2 -> House1 (Flow: 8.5 L/min), House2 (Flow: 17 L/min),
House2 ->
Junction1 -> Junction2 (Flow: 20 L/min),
Source -> Junction1 (Flow: 25 L/min),

Leak Detection Report:
Possible leak between Junction2 and House1 [Flow: 8.5 L/min]


=== Code Execution Successful ===
```

main.cpp

Share Run

```
23          cout << "Water Supply Network:\n";
24          for (const auto& pair : graph) {
25              cout << pair.first << " -> ";
26              for (const auto& edge : pair.second) {
27                  cout << edge.destination << " (Flow: " << edge
                        .flowRate << " L/min), ";
28              }
29              cout << endl;
30          }
31      }
32      void detectLeaks() {
33          cout << "\nLeak Detection Report:\n";
34          for (const auto& pair : graph) {
35              for (const auto& edge : pair.second) {
36                  if (edge.flowRate < FLOW_THRESHOLD) {
37                      cout << "Possible leak between " << pair.first
                            << " and "
38                           << edge.destination << " [Flow: " << edge
                                .flowRate << " L/min]\n";
39                  }
40              }
41          }
42      }
43  };
```

Output Clear

```
Water Supply Network:
House1 ->
Junction2 -> House1 (Flow: 8.5 L/min), House2 (Flow: 17 L/min),
House2 ->
Junction1 -> Junction2 (Flow: 20 L/min),
Source -> Junction1 (Flow: 25 L/min),

Leak Detection Report:
Possible leak between Junction2 and House1 [Flow: 8.5 L/min]


=== Code Execution Successful ===
```

main.cpp

Share Run

```
        cout << "\nLeak Detection Report:\n";
        for (const auto& pair : graph) {
            for (const auto& edge : pair.second) {
                if (edge.flowRate < FLOW_THRESHOLD) {
                    cout << "Possible leak between " << pair.first << " and "
                         << edge.destination << " [Flow: " << edge.flowRate <<
                            " L/min]\n";
                }
            }
        }
    }
};
int main() {
    WaterNetwork network;
    network.addNode("Source");
    network.addNode("Junction1");
    network.addNode("Junction2");
    network.addNode("House1");
    network.addNode("House2");
    network.addPipe("Source", "Junction1", 25.0);
    network.addPipe("Junction1", "Junction2", 20.0);
    network.addPipe("Junction2", "House1", 8.5);
    network.addPipe("Junction2", "House2", 17.0);
    network.displayNetwork();
    network.detectLeaks();
    return 0;
}
```

Output

Clear

```
Water Supply Network:
House1 ->
Junction2 -> House1 (Flow: 8.5 L/min), House2 (Flow: 17 L/min),
House2 ->
Junction1 -> Junction2 (Flow: 20 L/min),
Source -> Junction1 (Flow: 25 L/min),

Leak Detection Report:
Possible leak between Junction2 and House1 [Flow: 8.5 L/min]


=== Code Execution Successful ===
```