Samuel Barker
00100768
9/20/23
CS 417

Exam 1 – Problem 1

Given: Write a class AdjListGraph in C++ that implements this representation of a class.

Source Code:

```
//Samuel Barker
//00100768
//sbarker1@my.athens.edu
//CS 417, Exam One, Problem 1

#include <iostream>
#include <vector>
#include <list>

class Node {
public:
     int value;
     Node(int val) : value(val) {}
};

class AdjListGraph {
private:
     std::vector<Node> nodeVector;
     std::vector<std::list<int>> edgeVector;

public:

     AdjListGraph() {}

     // Add new node to the graph
     void addNode(Node x) {
          nodeVector.push_back(x);
          edgeVector.emplace_back();
     }

     // Delete node from graph and remove associated edges
     void deleteNode(Node x) {
          int indexToDelete = -1;
          for (int i = 0; i < nodeVector.size(); ++i) {
               if (nodeVector[i].value == x.value) {
                    indexToDelete = i;
                    break;
               }
          }

          if (indexToDelete != -1) {
               nodeVector.erase(nodeVector.begin() + indexToDelete);
               edgeVector.erase(edgeVector.begin() + indexToDelete);
               for (auto& edges : edgeVector) {
                    edges.remove(indexToDelete);
               }
          }
     }

     // Add edge from node x to node y
```

Samuel Barker
00100768
9/20/23
CS 417

```
    void addEdge(int x, int y) {
        if (x < 0 || x >= nodeVector.size() || y < 0 || y >= nodeVector.size()) {
            std::cout << "Invalid node index" << std::endl;
            return;
        }

        edgeVector[x].push_back(y);
    }

    // Delete edge from node x to node y
    void deleteEdge(int x, int y) {
        if (x < 0 || x >= nodeVector.size() || y < 0 || y >= nodeVector.size()) {
            std::cout << "Invalid node index" << std::endl;
            return;
        }

        edgeVector[x].remove(y);
    }
};

int main() {

    AdjListGraph graph;

    Node node0(0);
    Node node1(1);
    Node node2(2);

    graph.addNode(node0);
    graph.addNode(node1);
    graph.addNode(node2);

    // Add edges between nodes
    graph.addEdge(0, 1);
    graph.addEdge(1, 2);

    // Delete edge
    graph.deleteEdge(0, 1);

    graph.deleteNode(node1);

    return 0;
}
```