LAB TASK:14

NAME: HUZAIFA MUSTAFA

ID : SP22-BSCS-0046 (AM)

CODE:

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

const int MAX = 100;

class Graph {
public:
   int adj_matrix[MAX][MAX];
   int n;

   Graph() {
      cout << "Input Vertices: ";
      cin >> n;
      // Initialize the adjacency matrix with zeros
      for (int i = 0; i < n; i++) {
         for (int j = 0; j < n; j++) {
            adj_matrix[i][j] = 0;
         }
      }
   }
```

```
// Print the adjacency matrix
void PrintMatrix() {
   cout << "   ";
   for (int i = 0; i < n; i++) {
      cout << i + 1 << " ";
   }
   cout << endl;
   for (int i = 0; i < n; i++) {
      cout << i + 1 << " : ";
      for (int j = 0; j < n; j++) {
         cout << adj_matrix[i][j] << " ";
      }
      cout << endl;
   }
}

// Add an edge to the graph
void add_edge(int source, int destination, int type) {
   if (adj_matrix[source - 1][destination - 1] == 1) {
      cout << "Edge already exists." << endl;
   } else {
      adj_matrix[source - 1][destination - 1] = 1;
      if (type == 2) {
```

```
            adj_matrix[destination - 1][source - 1] = 1;
        }
        cout << "Edge added successfully." << endl;
    }
}

// Remove an edge from the graph
void remove_edge(int source, int destination, int type) {
    if (adj_matrix[source - 1][destination - 1] == 0) {
        cout << "Edge does not exist." << endl;
    } else {
        adj_matrix[source - 1][destination - 1] = 0;
        if (type == 2) {
            adj_matrix[destination - 1][source - 1] = 0;
        }
        cout << "Edge removed successfully." << endl;
    }
}

// Add an edge to the graph using adjacency list
void addEdge(vector<int> adj[], int s, int d) {
  adj[s].push_back(d);
  adj[d].push_back(s);
}
```

```
    // Print the adjacency list
    void printGraph(vector<int> adj[], int V) {
      for (int d= 0; d < V; ++d) {
            cout << "\n Vertex " << d << ":";
            for (auto x : adj[d]) {
               cout << "-> " << x;
            }
            cout << endl;
        }
    }
};

int main() {
    Graph g1;
    int choice;
    int typegraph = 0;
    int sourceVertex;
    int destinationVertex;
    vector<int> adj[MAX];

    do {
        cout << "Enter 1 to add an edge : " << endl;
        cout << "Enter 2 to remove an edge : " << endl;
```

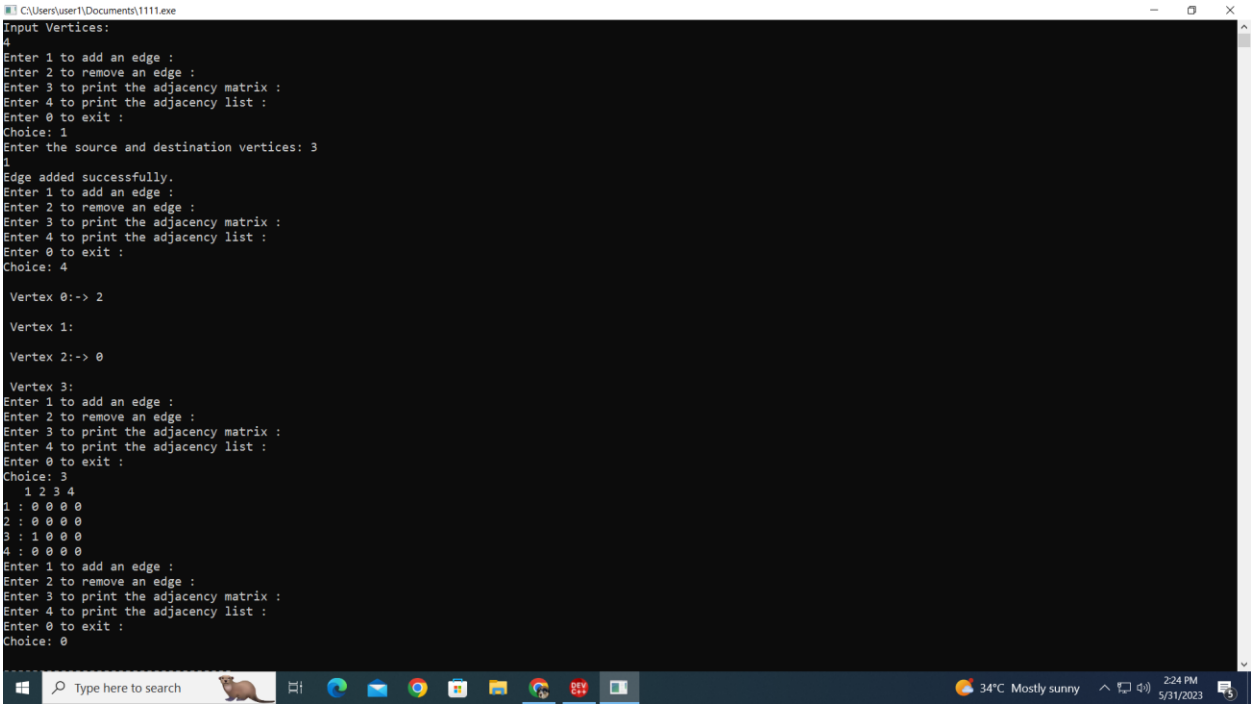
```
    cout << "Enter 3 to print the adjacency matrix : " << endl;
    cout << "Enter 4 to print the adjacency list : " << endl;
    cout << "Enter 0 to exit : " << endl;
    cout << "Choice: ";
    cin >> choice;

    if (choice == 1) {
        cout << "Enter the source and destination vertices: ";
        cin >> sourceVertex >> destinationVertex;
        g1.add_edge(sourceVertex, destinationVertex, typegraph);
        g1.addEdge(adj, sourceVertex-1, destinationVertex-1);
    } else if (choice == 2) {
        cout << "Enter the source and destination vertices: ";
        cin >> sourceVertex >> destinationVertex;
        g1.remove_edge(sourceVertex, destinationVertex, typegraph);
    } else if (choice ==3) {
        g1.PrintMatrix();
    }
    else if (choice == 4) {
        g1.printGraph(adj, g1.n);
    }
} while (choice != 0);

return 0;
```

}

SOURCE CODE:


C:\Users\user1\Documents\1111.exe
Input Vertices:
4
Enter 1 to add an edge :
Enter 2 to remove an edge :
Enter 3 to print the adjacency matrix :
Enter 4 to print the adjacency list :
Enter 0 to exit :
Choice: 1
Enter the source and destination vertices: 3
1
Edge added successfully.
Enter 1 to add an edge :
Enter 2 to remove an edge :
Enter 3 to print the adjacency matrix :
Enter 4 to print the adjacency list :
Enter 0 to exit :
Choice: 4
Vertex 0:-> 2
Vertex 1:
Vertex 2:-> 0
Vertex 3:
Enter 1 to add an edge :
Enter 2 to remove an edge :
Enter 3 to print the adjacency matrix :
Enter 4 to print the adjacency list :
Enter 0 to exit :
Choice: 3
1 2 3 4
1 : 0 0 0 0
2 : 0 0 0 0
3 : 1 0 0 0
4 : 0 0 0 0
Enter 1 to add an edge :
Enter 2 to remove an edge :
Enter 3 to print the adjacency matrix :
Enter 4 to print the adjacency list :
Enter 0 to exit :
Choice: 0
Type here to search
34°C Mostly sunny
2:24 PM
5/31/2023