| | Department of Computer Science and Engineering<br>Faculty of Engineering, South Eastern University of Sri Lanka | | |
|---|---|---|---|
| | **Subject** | **CS53003: Data Structure and Algorithms** | |
| | **Batch** | **E18** | **Semester** | **5** |

**Lab no and title : Hands-on Lab 9 – Stack and Queue**
**Name : G.W.P.R.R. Wijesinghe**
**Reg No : SEU-IS-18-EG-013**
**Submission Date : 08-03-2023**

---

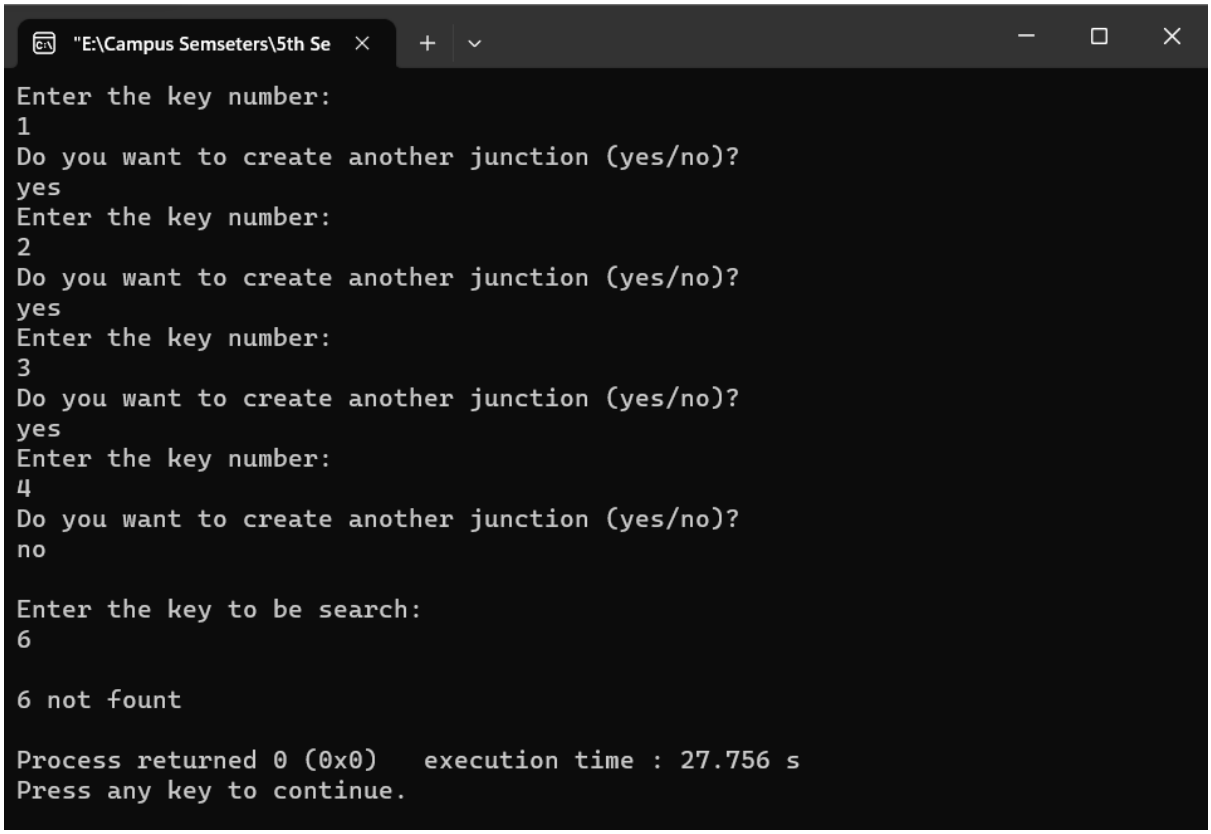
## Lab Exercise 1

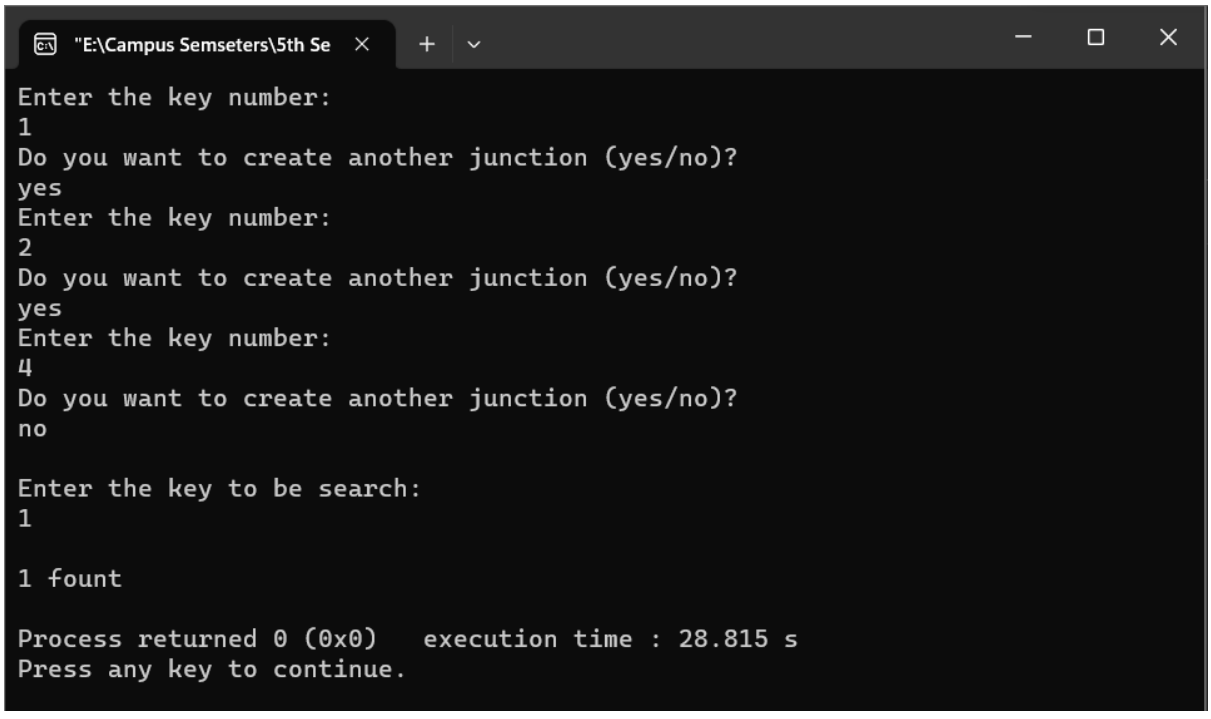
Start here | *CodeTemplate_Ex1.cpp

```cpp
//DO NOT CHANGE THE GIVEN CODE TEMPLATE. WRITE YOUR CODE IN THE PROVIDED PLACES ALONE.

#include <iostream>
using namespace std;

class Node
{
    public:
      int data;
      Node *left, *right;
};

class BST
{
    public :
    Node *root;
    BST() {
        root = NULL;
    }
    Node* insert(Node*, int);
    int search(int);
};

Node* BST :: insert(Node* root, int value){
    if (root == NULL) {
        Node* newNode = new Node();
        newNode-> data = value;
        newNode-> left = NULL;
        newNode-> right = NULL;
        return newNode;
    }
    if (value < root -> data){
        root->left = insert(root->left, value);
    }
    else if (value > root -> data) {
        root->right = insert(root->right, value);
    }
    return root;
}

```

```
41  int BST :: search(int searchKey) {
42      Node* curr = root;
43      while (curr !=NULL){
44          if (searchKey == curr->data){
45              return 1;
46          }
47              else if (searchKey < curr-> data){
48                  curr = curr->left;
49              }
50              else{
51                  curr = curr->right;
52              }
53
54      }
55
56      return 0; //Change the return value as per the problem statement
57  }
58
59  int main()
60  {
61      BST tree;
62      string ch="yes";
63      int num,searchKey;
64      cout<<"Enter the key number:\n";
65      cin>>num;
66
67      tree.root=tree.insert(tree.root, num);
68      do  {
69          cout<<"Do you want to create another junction (yes/no)?\n";
70          cin>>ch;
71          if (ch.compare("yes")==0){
72              cout<<"Enter the key number:\n";
73              cin>>num;
74              tree.root=tree.insert(tree.root, num);
75
76          }
77          else{
78              break;
79          }
80      }while(true);
81
82      cout<<"\nEnter the key to be search: \n";
83      cin>>searchKey;
```

```
84      if (tree.search(searchKey)){
85          cout<<endl<<searchKey<<" fount\n";
86      }
87      else{
88          cout<<endl<<searchKey<<" not fount\n";
89      }
90      return 0;
91  }
92
```


"E:\Campus Semseters\5th Se
Enter the key number:
1
Do you want to create another junction (yes/no)?
yes
Enter the key number:
2
Do you want to create another junction (yes/no)?
yes
Enter the key number:
3
Do you want to create another junction (yes/no)?
yes
Enter the key number:
4
Do you want to create another junction (yes/no)?
no
Enter the key to be search:
6
6 not fount
Process returned 0 (0x0) execution time : 27.756 s
Press any key to continue.


"E:\Campus Semseters\5th Se
Enter the key number:
1
Do you want to create another junction (yes/no)?
yes
Enter the key number:
2
Do you want to create another junction (yes/no)?
yes
Enter the key number:
4
Do you want to create another junction (yes/no)?
no
Enter the key to be search:
1
1 fount
Process returned 0 (0x0) execution time : 28.815 s
Press any key to continue.

## Lab Exercise 2

Start here | CodeTemplate_Ex1.cpp | CodeTemplate_Ex2.cpp

```cpp
//DO NOT CHANGE THE GIVEN CODE TEMPLATE. WRITE YOUR CODE IN THE PROVIDED PLACES ALONE.


#include <iostream>
using namespace std;
class Node {
    public:
    int data;
    Node *left, *right;
};

class BST {
    public :
    Node *root;
    BST() {
        root = NULL;
    }
    Node* insert(Node*, int);
    void preOrder(Node*);
    void inOrder(Node*);
    void postOrder(Node*);
};

Node* BST :: insert(Node* root, int value) {
    if (root == NULL) {
        Node* newNode = new Node();
        newNode-> data = value;
        newNode-> left = newNode -> right = NULL;
        root = newNode;
        return root;
    }
    if (value < root -> data){
        root->left = insert(root->left, value);
    }
    else {
        root->right = insert(root->right, value);
    }


    return root;
}
```

Start here | CodeTemplate_Ex1.cpp | CodeTemplate_Ex2.cpp

```cpp
void BST :: preOrder(Node* root){
  if (root == NULL) {
    return;
  }
  cout<< root-> data << " ";
  preOrder(root->left);
  preOrder(root->right);
}

void BST :: inOrder(Node* root) {
    if (root == NULL) {
    return;
  }
  inOrder(root->left);
  cout<< root-> data << " ";
  inOrder(root->right);
}


void BST :: postOrder(Node* root) {
  if (root == NULL) {
    return;
  }
  postOrder(root->left);
  postOrder(root->right);
  cout<< root-> data << " ";
}

```

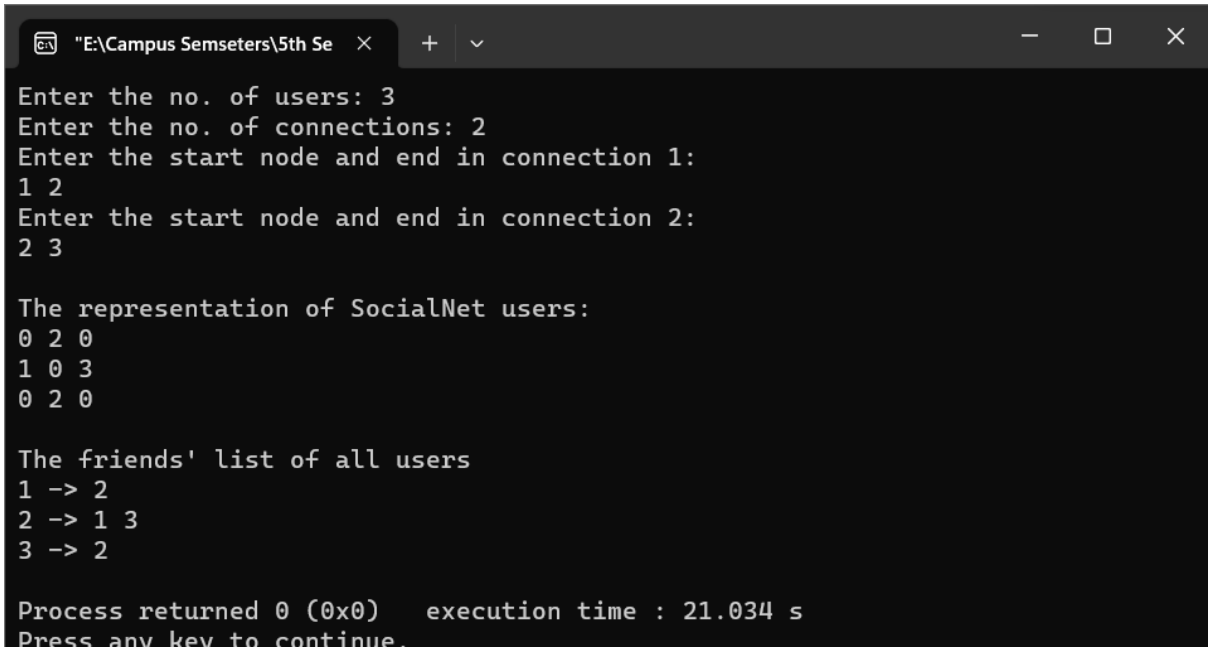
```
70
71  int main() {
72      BST tree;
73      string ch="yes";
74      int num;
75      do  {
76          cout<<"Enter the element to be inserted in the tree\n";
77          cin>>num;
78          tree.root=tree.insert(tree.root, num);
79          cout<<"Do you want to insert another element?\n";
80          cin>>ch;
81      }while(ch.compare("yes")==0);
82
83      cout<<"Preorder Traversal : The elements in the tree are:\n";
84      tree.preOrder(tree.root);
85      cout<<"\n";
86      cout<<"Inorder Traversal : The elements in the tree are:\n";
87      tree.inOrder(tree.root);
88      cout<<"\n";
89      cout<<"Postorder Traversal : The elements in the tree are:\n";
90      tree.postOrder(tree.root);
91      cout<<"\n";
92      return 0;
93  }
94
95
96
97
```

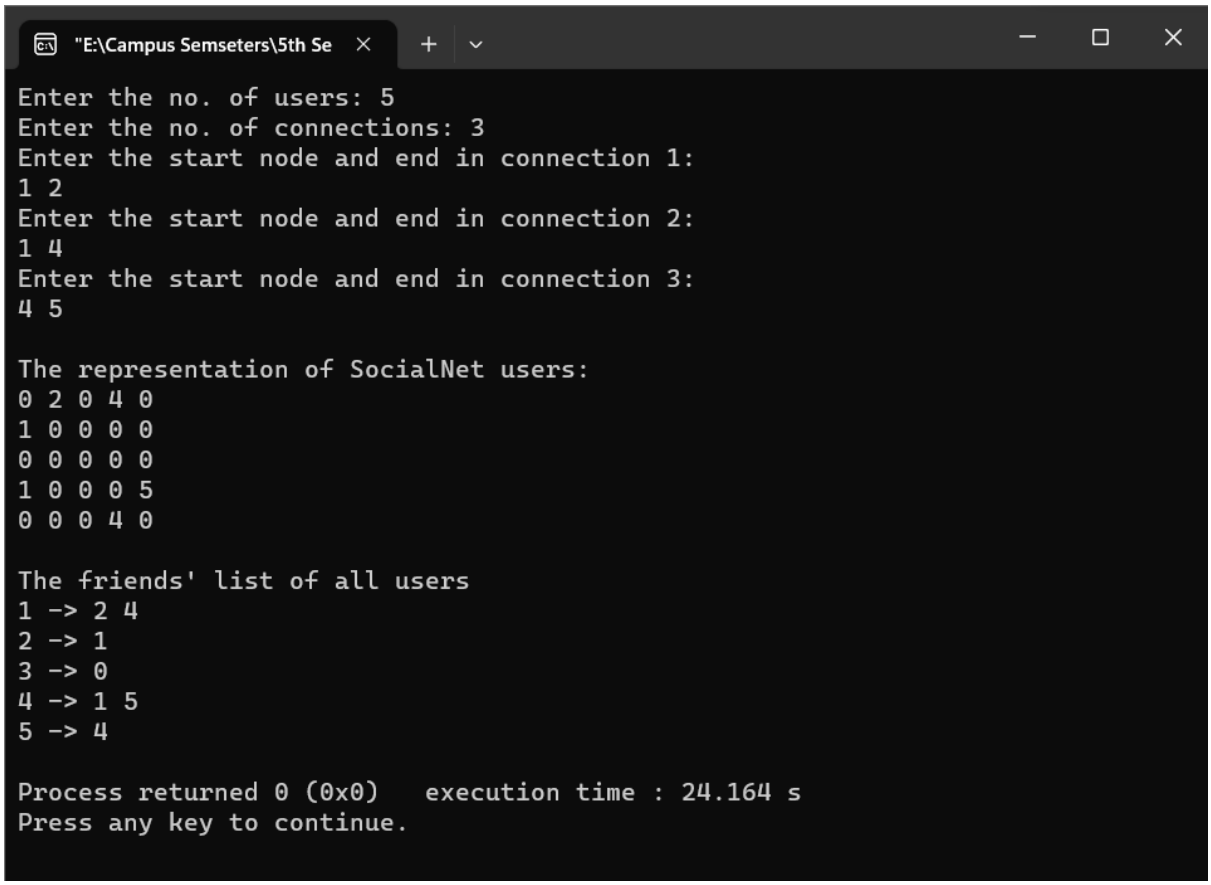
```
"E:\Campus Semseters\5th Se
Enter the element to be inserted in the tree
5
Do you want to insert another element?
yes
Enter the element to be inserted in the tree
3
Do you want to insert another element?
yes
Enter the element to be inserted in the tree
6
Do you want to insert another element?
yes
Enter the element to be inserted in the tree
8
Do you want to insert another element?
no
Preorder Traversal : The elements in the tree are:
5 3 6 8
Inorder Traversal : The elements in the tree are:
3 5 6 8
Postorder Traversal : The elements in the tree are:
3 8 6 5

Process returned 0 (0x0)   execution time : 26.101 s
Press any key to continue.
```

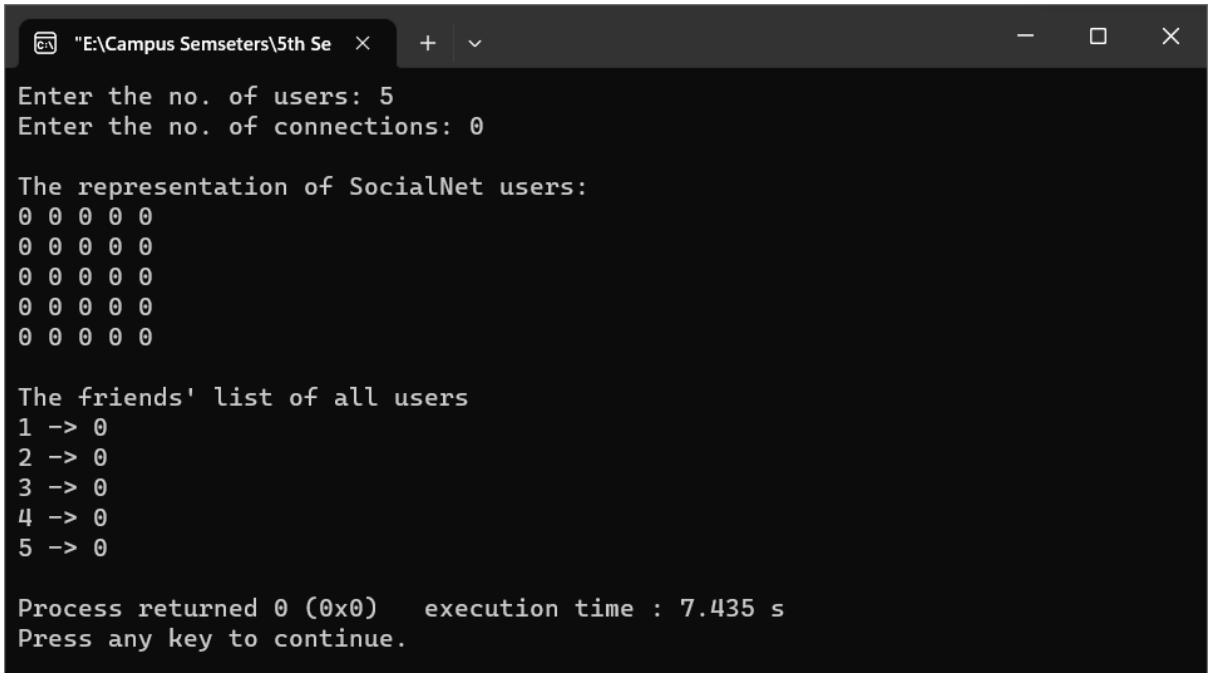
## Lab Exercise 03

Start here | CodeTemplate_Ex1.cpp | CodeTemplate_Ex2.cpp | CodeTemplate_Ex3.cpp

```cpp
//DO NOT CHANGE THE GIVEN CODE TEMPLATE. WRITE YOUR CODE IN THE PROVIDED PLACES ALONE.
//If required, you can add additional functions or header files.


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

int main(){
        int a, b;
        cout <<"Enter the no. of users: ";
        cin >> a;
        cout <<"Enter the no. of connections: ";
        cin >> b;
        vector < vector<int>> setlist(a+1);

        for(int i=0; i<b; i++){
            int x,y;
            cout<<"Enter the start node and end in connection "<<i+1<<":"<<endl;
            cin>>x>>y;
            if (x<1 || x>a ||y<1 ||y>a){
                cout <<"Invalid node number" <<endl;
                i--;
                continue;
            }
            setlist[x].push_back(y);
            setlist[y].push_back(x);
        }
        cout<<"\nThe representation of SocialNet users: "<<endl;
        for(int i = 1; i<=a;i++){
        for(int j =1; j<=a; j++){
                if (find(setlist[i].begin(), setlist[i].end(), j)!=setlist[i].end()){
                    cout<<j<<" ";
                }
                else{
                    cout<<"0 ";
                }

        }
        cout<<endl;
        }
         cout<<"\nThe friends' list of all users" <<endl;
         for(int i =1; i <= a; i++){
             cout<<i <<" -> ";
             if(setlist[i].empty()){
                 cout<<0<<endl;
             }
         else{
             for (int j = 0;j<setlist[i].size(); j++){
                 cout<<setlist[i][j]<<" ";
             }
             cout<<endl;
         }
         }
    return 0;
}
```


"E:\Campus Semseters\5th Se
Enter the no. of users: 3
Enter the no. of connections: 2
Enter the start node and end in connection 1:
1 2
Enter the start node and end in connection 2:
2 3
The representation of SocialNet users:
0 2 0
1 0 3
0 2 0
The friends' list of all users
1 -> 2
2 -> 1 3
3 -> 2
Process returned 0 (0x0) execution time : 21.034 s
Press any key to continue.

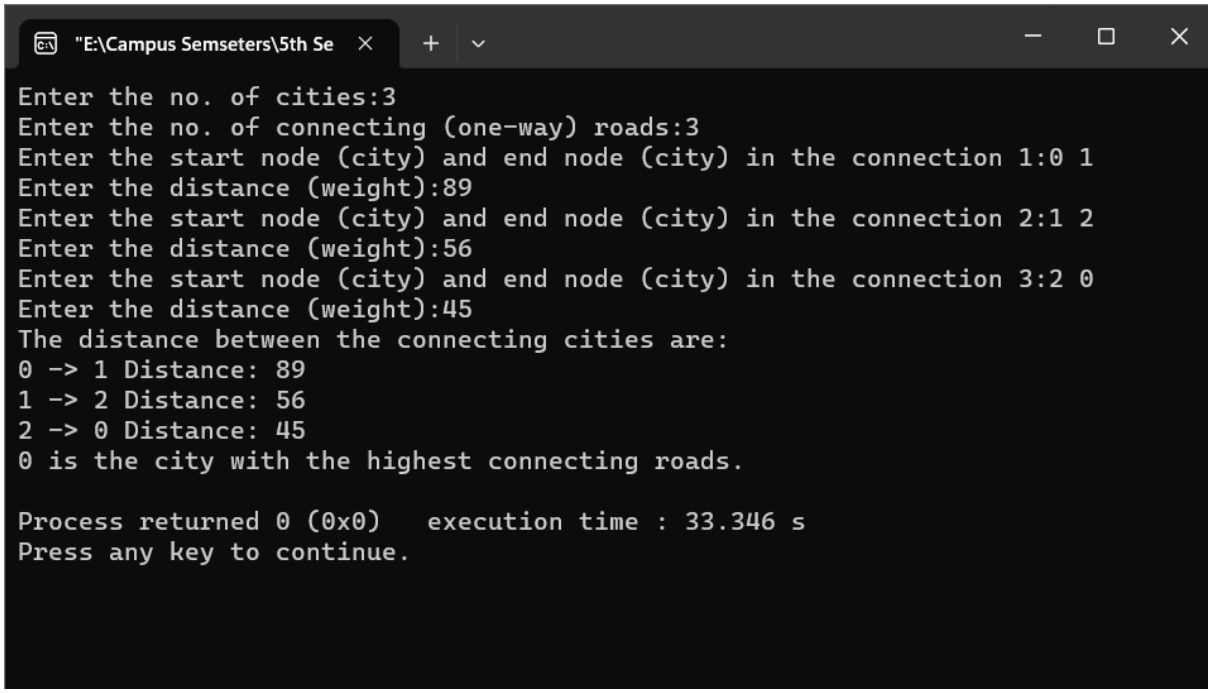

"E:\Campus Semseters\5th Se
Enter the no. of users: 5
Enter the no. of connections: 3
Enter the start node and end in connection 1:
1 2
Enter the start node and end in connection 2:
1 4
Enter the start node and end in connection 3:
4 5
The representation of SocialNet users:
0 2 0 4 0
1 0 0 0 0
0 0 0 0 0
1 0 0 0 5
0 0 0 4 0
The friends' list of all users
1 -> 2 4
2 -> 1
3 -> 0
4 -> 1 5
5 -> 4
Process returned 0 (0x0) execution time : 24.164 s
Press any key to continue.

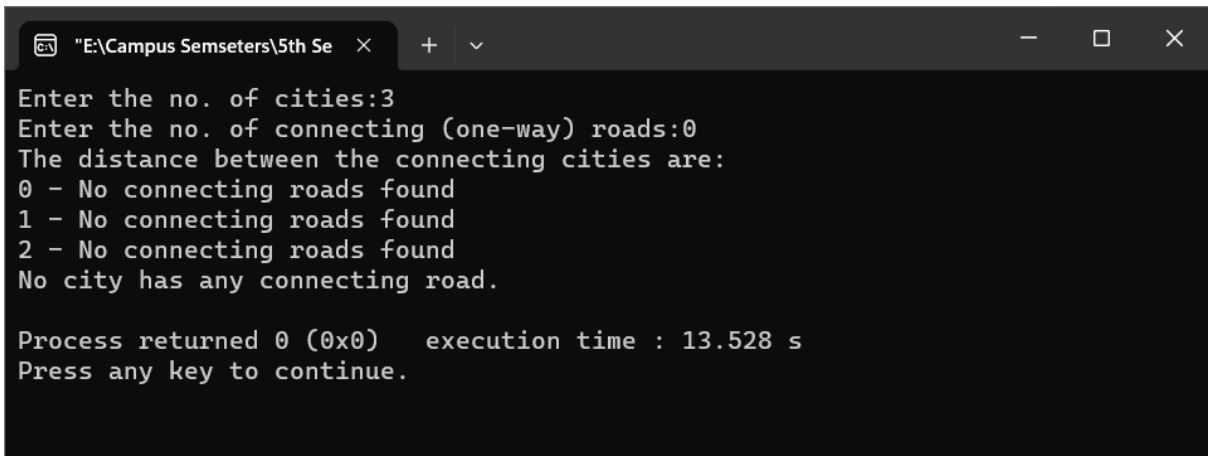

"E:\Campus Semseters\5th Se
Enter the no. of users: 5
Enter the no. of connections: 0
The representation of SocialNet users:
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
The friends' list of all users
1 -> 0
2 -> 0
3 -> 0
4 -> 0
5 -> 0
Process returned 0 (0x0) execution time : 7.435 s
Press any key to continue.

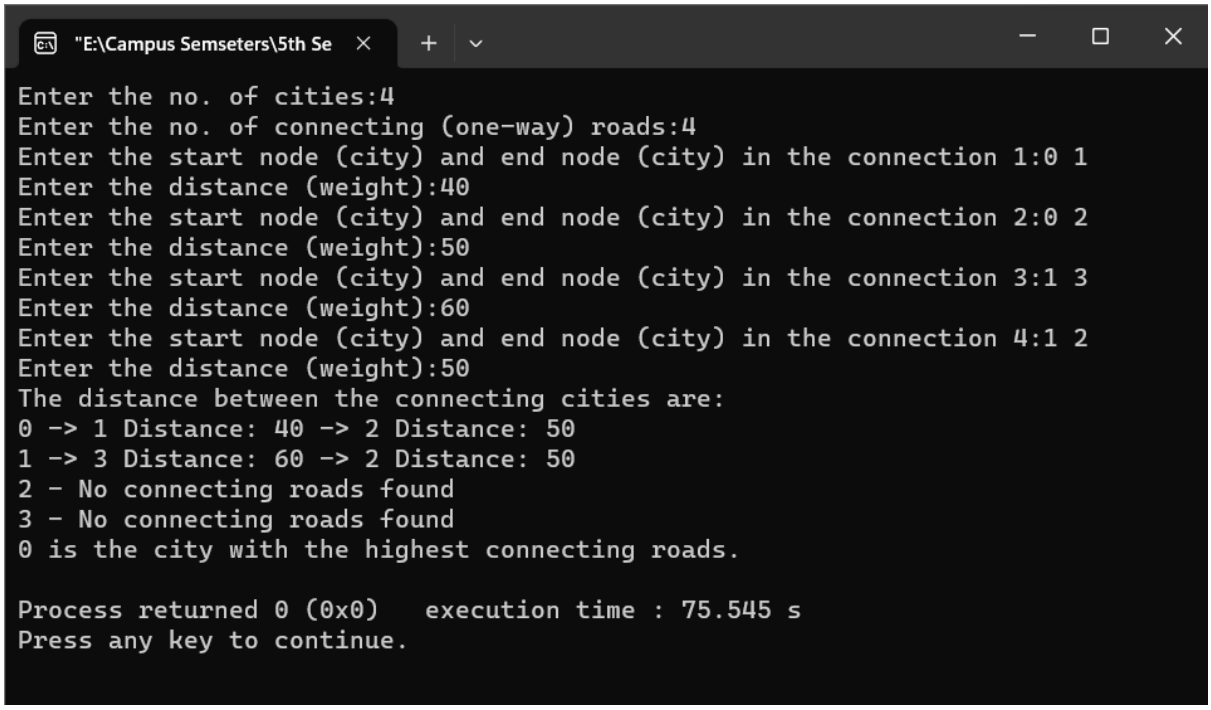
## Lab Exercise 4

Start here | CodeTemplate_Ex1.cpp | CodeTemplate_Ex2.cpp | CodeTemplate_Ex3.cpp | CodeTemplate_Ex4.cpp

```cpp
//DO NOT CHANGE THE GIVEN CODE TEMPLATE. WRITE YOUR CODE IN THE PROVIDED PLACES ALONE.
//If required, you can add additional functions or header files.

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

struct Road {
    int start, end, distance;
};

int main() {
    int n, m;
    cout << "Enter the no. of cities:";
    cin >> n;
    cout << "Enter the no. of connecting (one-way) roads:";
    cin >> m;
    vector<vector<Road>> graph(n);
    for (int i = 0; i < m; i++) {
        int start, end, distance;
        cout << "Enter the start node (city) and end node (city) in the connection " << i+1 << ":";
        cin >> start >> end;
        cout << "Enter the distance (weight):";
        cin >> distance;
        graph[start].push_back({start, end, distance});
    }
    cout << "The distance between the connecting cities are:\n";
    for (int i = 0; i < n; i++) {
        cout << i;
        if (graph[i].empty()) {
            cout << " - No connecting roads found\n";
        } else {
            for (Road road : graph[i]) {
                cout << " -> " << road.end << " Distance: " << road.distance;
            }
            cout << '\n';
        }
    }
    int max_connections = 0;
    int city_with_max_connections = -1;
    for (int i = 0; i < n; i++) {
        if (graph[i].size() > max_connections) {
            max_connections = graph[i].size();
            city_with_max_connections = i;
        }
    }
    if (city_with_max_connections == -1) {
        cout << "No city has any connecting road.\n";
    } else {
        cout << city_with_max_connections << " is the city with the highest connecting roads.\n";
    }
    return 0;
}
```


"E:\Campus Semseters\5th Se
Enter the no. of cities:3
Enter the no. of connecting (one-way) roads:3
Enter the start node (city) and end node (city) in the connection 1:0 1
Enter the distance (weight):89
Enter the start node (city) and end node (city) in the connection 2:1 2
Enter the distance (weight):56
Enter the start node (city) and end node (city) in the connection 3:2 0
Enter the distance (weight):45
The distance between the connecting cities are:
0 -> 1 Distance: 89
1 -> 2 Distance: 56
2 -> 0 Distance: 45
0 is the city with the highest connecting roads.
Process returned 0 (0x0) execution time : 33.346 s
Press any key to continue.


"E:\Campus Semseters\5th Se
Enter the no. of cities:3
Enter the no. of connecting (one-way) roads:0
The distance between the connecting cities are:
0 - No connecting roads found
1 - No connecting roads found
2 - No connecting roads found
No city has any connecting road.
Process returned 0 (0x0) execution time : 13.528 s
Press any key to continue.


"E:\Campus Semseters\5th Se
Enter the no. of cities:4
Enter the no. of connecting (one-way) roads:4
Enter the start node (city) and end node (city) in the connection 1:0 1
Enter the distance (weight):40
Enter the start node (city) and end node (city) in the connection 2:0 2
Enter the distance (weight):50
Enter the start node (city) and end node (city) in the connection 3:1 3
Enter the distance (weight):60
Enter the start node (city) and end node (city) in the connection 4:1 2
Enter the distance (weight):50
The distance between the connecting cities are:
0 -> 1 Distance: 40 -> 2 Distance: 50
1 -> 3 Distance: 60 -> 2 Distance: 50
2 - No connecting roads found
3 - No connecting roads found
0 is the city with the highest connecting roads.
Process returned 0 (0x0) execution time : 75.545 s
Press any key to continue.

***