Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

COMP-202

LAB WORK-6

Submitted By:

Aawishkar Tiwari (59)

Samir Wagle (60)

Group: Computer Engineering

Level: 2nd Year First Semester

Submitted to:

Dr. Rajani Chulyadyo

Department of Computer Science and Engineering

Submission Date:07/12/2022

Github repository link :

SamirWagle/CE2020_LAB6_59_60 (github.com)

Main Function

```cpp
#include <iostream>
#include "graph.cpp"

using namespace std;

int main(){
    cout<<"Graph Adjacency List\n\n";
    graph g1;

    cout<<"Is our graph empty??? \n";
    if(g1.isEmpty()){
        cout<<"The graph is empty\n\n";
    }else{
        cout<<"The graph is not empty\n\n";
    }

    cout<<"Adding vertex to the graph \n\n And checking isEmpty\n";
    g1.addVertex(5);
    if(g1.isEmpty()){
        cout<<"Graph is empty\n\n";
    }else{
        cout<<"Graph is not empty\n\n";
    }

    cout<<"Is the graph Directed???\nLet's check\n\n";
    if(g1.isDirected()){
        cout<<"The graph is directed\n\n";
    }else{
        cout<<"The graph is not directed\n\n";
    }

    cout<<"Adding more vertex and checking no of vertex in the graph\n";
    g1.addVertex(100);
    g1.addVertex(50);
    g1.addVertex(40);
    g1.addVertex(30);
    g1.addVertex(5);
```

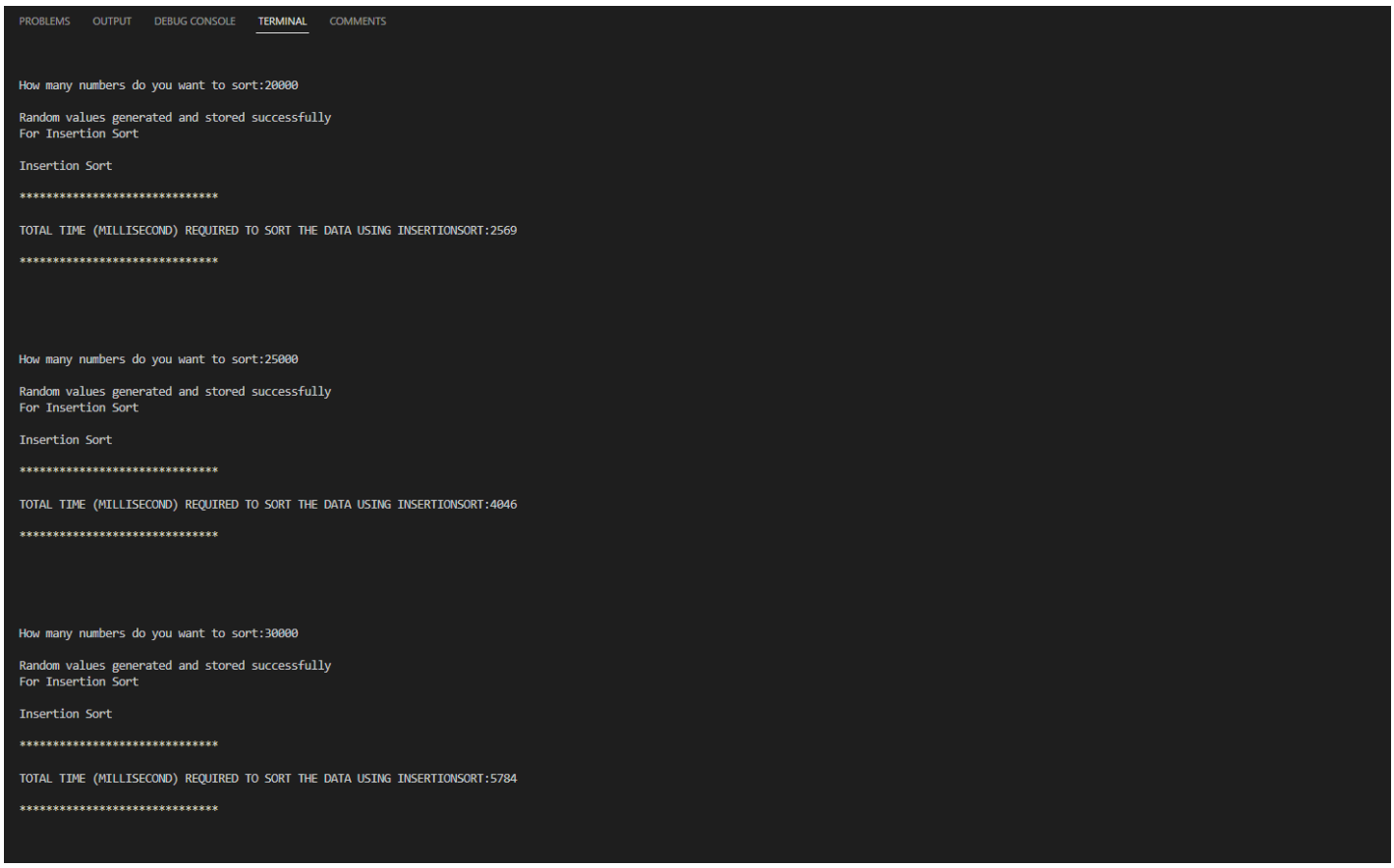
```
    cout<<"The number of vertex in the graph is "<<g1.numVertex()<<endl<<endl;

    cout<<"Adding edges between vertices\n\n Checking if they are neighbours or
not\n";
    g1.addEdges(50,40);
    g1.addEdges(30,100);
    g1.addEdges(5,100);
    g1.addEdges(100,40);
    g1.addEdges(5,50);
    if(g1.isNeighbour(30,100)){
        cout<<"The numbers are neighbours of each other\n\n";
    }else{
        cout<<"The numbers are not neighbours\n\n";
    }

    cout<<"Checking the Degree of a Vertex\n";
    cout<<"Indegree of vertex 100 is "<<g1.inDegree(5)<<endl;
    cout<<"Outdegree of vertex 5 is "<<g1.outDegree(3)<<endl;
    cout<<"Degree of vertex 40 is "<<g1.degree(9)<<endl<<endl;

    cout<<"The total number of edges in the graph "<<g1.numEdges()<<endl<<endl;

    if(g1.isNeighbour(30,100)){
        cout<<"30 and 100 are neighbours\n";
    }else{
        cout<<"30 and 100 are not neighbours\n";
    }
    cout<<"Removing an edge from the vertex and checking if they are
neighbours\n";
    g1.removeEdges(30,100);
    if(g1.isNeighbour(30,100)){
        cout<<"30 and 100 are neighbours\n\n";
    }else{
        cout<<"30 and 100 are not neighbours\n\n";
    }

    cout<<"Removing a vertex from the graph and showing the total no of
vertices\n";
    cout<<"The number before removing is "<<g1.numVertex()<<endl;

    cout<<"The number after removing is "<<endl;
 g1.removeVertex(3);
    return 0;
}
```

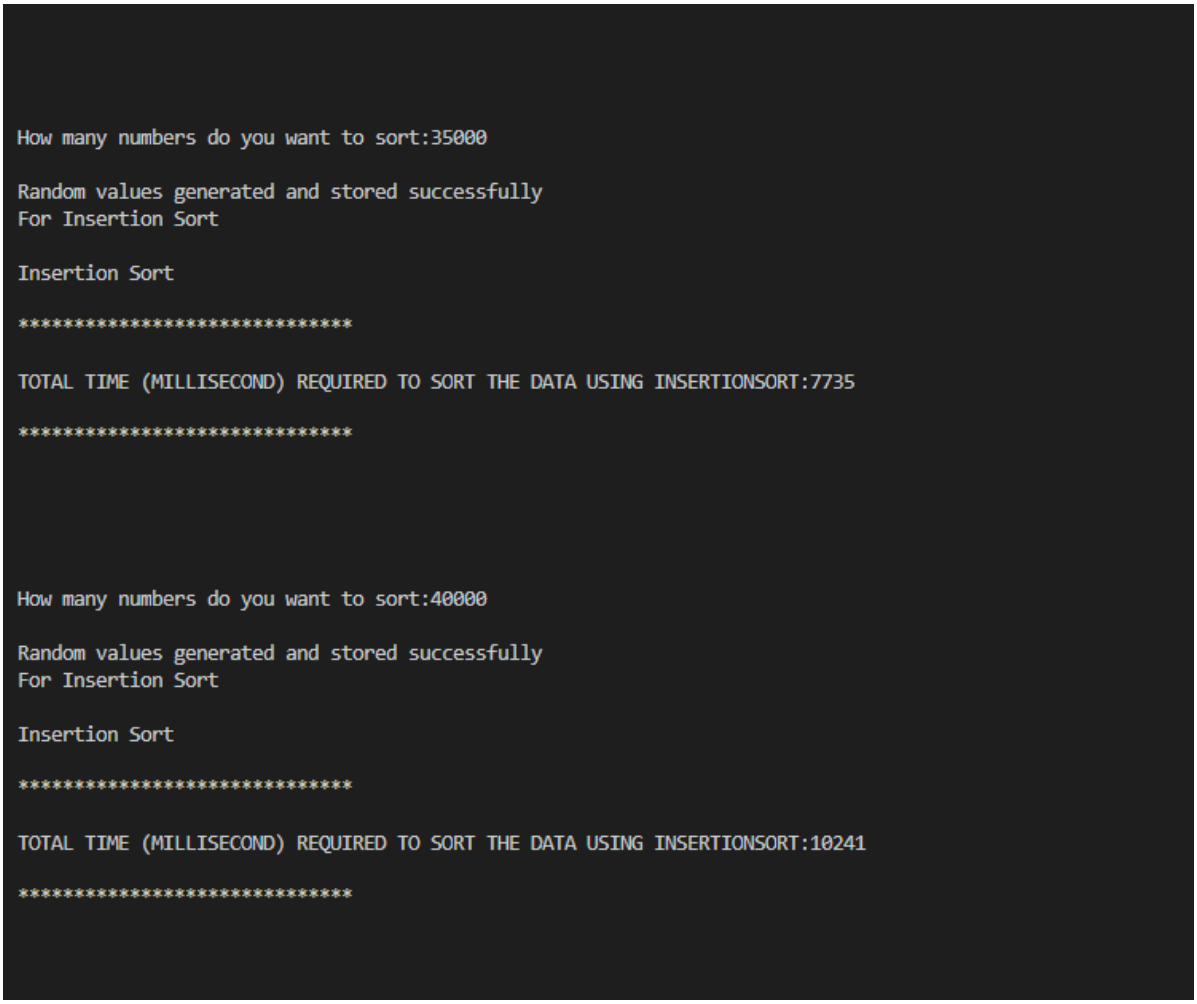
Main Function

```
                                        g++ main.cpp
PS C:\Users\samir\Desktop\CE2020_LAB6_59_60> ./a.exe
Graph Adjacency List

Is our graph empty???
The graph is empty

Adding vertex to the graph

 And checking isEmpty
Graph is not empty

Is the graph Directed???
Let's check

The graph is directed

Adding more vertex and checking no of vertex in the graph
The number of vertex in the graph is 6

Adding edges between vertices

 Checking if they are neighbours or not
The numbers are neighbours of each other

Checking the Degree of a Vertex
Indegree of vertex 100 is 0
Outdegree of vertex 5 is 0
Degree of vertex 40 is 0

The total number of edges in the graph 5

30 and 100 are neighbours
Removing an edge from the vertex and checking if they are neighbours
30 and 100 are not neighbours

Removing a vertex from the graph and showing the total no of vertices
The number before removing is 6
The number after removing is 100
PS C:\Users\samir\Desktop\CE2020_LAB6_59_60>
```

Output

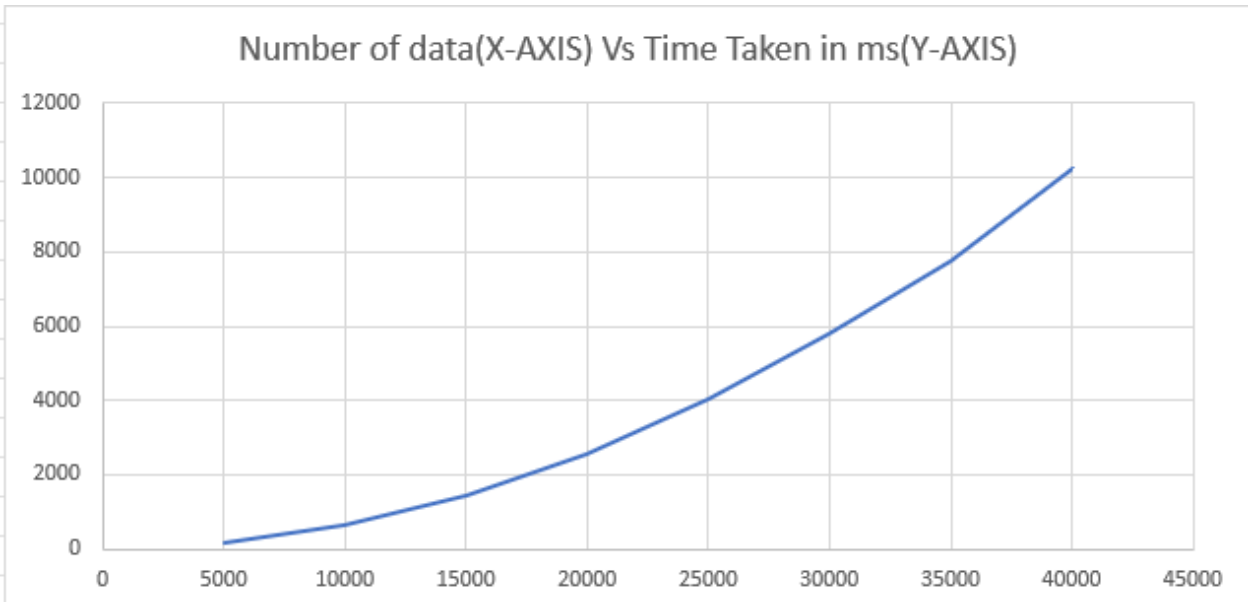
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\samir\Desktop\CE2020_Lab5_59_60> g++ main2.cpp
PS C:\Users\samir\Desktop\CE2020_Lab5_59_60> ./a.exe

How many times do you want to sort: 8

How many numbers do you want to sort:5000

Random values generated and stored successfully
For Insertion Sort

Insertion Sort

*****************************

TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:172

*****************************




How many numbers do you want to sort:10000

Random values generated and stored successfully
For Insertion Sort

Insertion Sort

*****************************

TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:646

*****************************




How many numbers do you want to sort:15000

Random values generated and stored successfully
For Insertion Sort

Insertion Sort

*****************************

TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:1460
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
COMMENTS
How many numbers do you want to sort:20000
Random values generated and stored successfully
For Insertion Sort
Insertion Sort
****
TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:2569
****
How many numbers do you want to sort:25000
Random values generated and stored successfully
For Insertion Sort
Insertion Sort
****
TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:4046
****
How many numbers do you want to sort:30000
Random values generated and stored successfully
For Insertion Sort
Insertion Sort
****
TOTAL TIME (MILLISECOND) REQUIRED TO SORT THE DATA USING INSERTIONSORT:5784
****



Here data is sort using insertion sort and time taken to sort is being recorded. The time and number of data sort graph is drawn to analyze the time difference due to increase in amount of data.

Graph


Number of data(X-AXIS) Vs Time Taken in ms(Y-AXIS)
12000
10000
8000
6000
4000
2000
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000