THE UNIVERSITY
OF LAHORE
ISLAMABAD
CAMPUS

DATA STRUCTURE AND ALOGRITHUM

Lab Report

Name:	SAMI ULLAH
Registration #:	SEU-S17-030
Lab Report #:	08
Dated:	5-28-2018
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 1
GRAPH

Objective
To understand the implementation of adjancy matrix.

Software Tool
1.
DEV C++

1 Theory

There are 3 function creted which perfom differents tasks 1. DIRECTED GRAPH

2. UNDIRECTED GRAPH

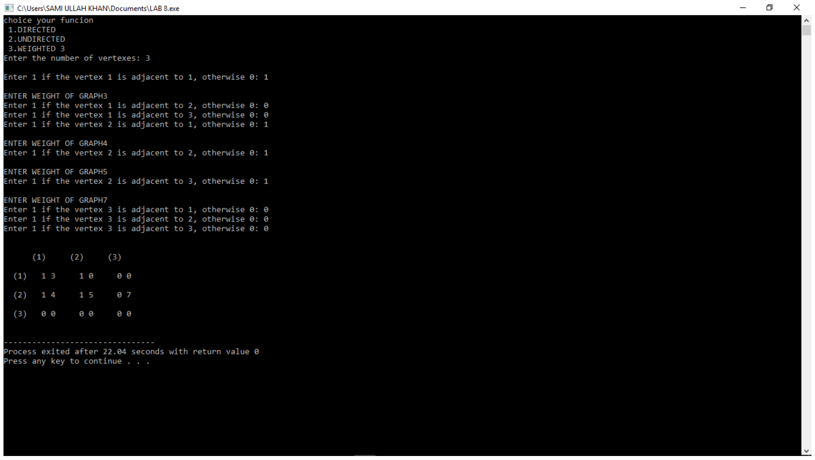
3. WEIGHTED GRAPH

2 Task

2.1 Procedure: Task 1

```
#include<iostream>
#include<iomanip>
using namespace std;
void PrintMat(int mat[][20], int n,int weight[20][20])
{
        int i, j;

        cout<<”\n\n”<<setw(4)<<”␣”;
        for(i = 0; i < n; i++)
                cout<<setw(3)<<”(”<<i+1<<”)␣␣␣”;
        cout<<”\n\n”;
```



```
choice your funcion
 1.DIRECTED
 2.UNDIRECTED
 3.WEIGHTED 3
Enter the number of vertexes: 3

Enter 1 if the vertex 1 is adjacent to 1, otherwise 0: 1

ENTER WEIGHT OF GRAPH3
Enter 1 if the vertex 1 is adjacent to 2, otherwise 0: 0
Enter 1 if the vertex 1 is adjacent to 3, otherwise 0: 0
Enter 1 if the vertex 2 is adjacent to 1, otherwise 0: 1

ENTER WEIGHT OF GRAPH4
Enter 1 if the vertex 2 is adjacent to 2, otherwise 0: 1

ENTER WEIGHT OF GRAPH5
Enter 1 if the vertex 2 is adjacent to 3, otherwise 0: 1

ENTER WEIGHT OF GRAPH7
Enter 1 if the vertex 3 is adjacent to 1, otherwise 0: 0
Enter 1 if the vertex 3 is adjacent to 2, otherwise 0: 0
Enter 1 if the vertex 3 is adjacent to 3, otherwise 0: 0


      (1)     (2)     (3)

  (1)   1 3     1 0     0 0

  (2)   1 4     1 5     0 7

  (3)   0 0     0 0     0 0


--------------------------------
Process exited after 22.04 seconds with return value 0
Press any key to continue . . .
```

Figure 1: Time Independent Feature Set

```
        // Print 1 if the corresponding vertexes are connected otherwise 0
        for(i = 0; i < n; i++)
        {
                cout<<setw(3)<<"("<<i+1<<")";
                for(j = 0; j < n; j++)
                {
                        cout<<setw(4)<<mat[i][j]<<"␣" <<weight[i][j]<<"␣␣"
                }
                cout<<"\n\n";
        }
}
void PrintMat(int mat[][20], int n)
{
        int i, j;

        cout<<"\n\n"<<setw(4)<<"";
        for(i = 0; i < n; i++)
                cout<<setw(3)<<"("<<i+1<<")";
        cout<<"\n\n";

        // Print 1 if the corresponding vertexes are connected otherwise 0
        for(i = 0; i < n; i++)
        {
                cout<<setw(3)<<"("<<i+1<<")";
                for(j = 0; j < n; j++)
                {
                        cout<<setw(4)<<mat[i][j];
                }
                cout<<"\n\n";
        }
}

int main()
{
        int n;
        int i, j, v;
        int  mat[20][20];
        int weight[20][20];
        cout<<"choice␣your␣funcion␣\n␣1.DIRECTED␣\n␣2.UNDIRECTED␣\n␣3.WEIGH
        cin>>n;
        switch(n)
        {
                case 1:


        cout<<"Enter␣the␣number␣of␣vertexes:␣";
        cin>>v;

        int  mat[20][20];

        cout<<"\n";
        // Take input of the adjacency of each pair of vertexes.
        for(i = 0; i < v; i++)
        {
                for(j = 0; j < v; j++)
                {

                        {
                                cout<<"Enter␣1␣if␣the␣vertex␣"<<i+1<<"␣is␣
                                cin>>mat[i][j];

                        //      mat[j][i] = mat[i][j];

                        }
                }
        }

        PrintMat(mat, v);
        break;
            case 2:


        cout<<"Enter␣the␣number␣of␣vertexes:␣";
        cin>>v;




        cout<<"\n";
        // Take input of the adjacency of each pair of vertexes.
        for(i = 0; i < v; i++)
        {
                for(j = i; j < v; j++)
                {
                        if(i != j)
                        {
                                cout<<"Enter␣1␣if␣the␣vertex␣"<<i+1<<"␣is␣
                                cin>>mat[i][j];

                                mat[j][i] = mat[i][j];

                        }
                        else
                                mat[i][j] = 0;
                }
        }

        PrintMat(mat, v);
        break;

            case 3:
                        int i, j, v;

        cout<<"Enter␣the␣number␣of␣vertexes:␣";
```

```
        cin>>v;




        cout<<"\n";
        // Take input of the adjacency of each pair of vertexes.
        for(i = 0; i < v; i++)
        {
                for(j = 0; j < v; j++)
                {

                        {
                                cout<<"Enter␣1␣if␣the␣vertex␣"<<i+1<<"␣is␣
                                cin>>mat[i][j];

                                mat[j][i] = mat[i][j];
                                if(mat[i][j]==1)
                                {

                                    cout<<endl<<"ENTER␣WEIGHT␣OF␣GRAPH";
                                    cin>>weight[i][j];

                            }
                            else
                               weight[i][j]=0;
                        }

                }
        }

        PrintMat(mat, v, weight);
        break;
        default:
                 cout<<"INVALID␣:";

        }
        return 0;
}
```

3 Conclusion

in this lab we perform 3 differents task of graph and well understand them in the lab.