**Sandesh Ghimire**

**CSCI 301 section 3**

**Computer Science 2**

**Project 11**

**1st December**

**Introduction**

Prim’s Algorithm is one of the algorithms that finds the minimum spanning tree (MST) of an undirected weighted graph. It starts at any vertex that is closest to the vertex in the tree which joins the tree afterward. We basically use three arrays to do the prim’s algorithm.

**Data Structures:**

The data structures used in this project is Array. We used 1D as well as 2D array to do this program. 2D array is used to make matrix that holds the weight of the graph. We used file stream to open a .txt file and extract the dates from there. We used adjacency list to represent the undirected graph. We also used matrix to keep the weights and vertex of the graph.

**Functions:**

The program contains only one file i.e. main.cpp

*Void Prim()* : This function is a heart of prim’s algorithm. It carries out the whole algorithm. It takes three parameters i.e. an integer, a matrix and an integer list. It initializes all the lists and matrix first and starts finding the minimum rout. After calculating the MST it displays the graph with the weight followed by the total weight.

**The Main Program:**

The mainfunction asks the input to the user first. It asks for the number of vertices they want. After that the program reads the file name “vertices.txt”. It takes out the vertex and their weight and store them to a variable. After doing all this things the main function calls the prim() function to perform Prim’s algorithm.

**Code**

**Main.cpp**

/\*Prim’s Algorithm is one of the algorithms that finds the minimum spanning tree (MST) of an undirected weighted graph.

It starts at any vertex that is closest to the vertex in the tree which joins the tree afterward.

 We basically use three arrays to do the prim’s algorithm. \*/

#include <iostream>

#include<iomanip>

#include<cstdlib>

#include <cmath>

#include <fstream>

using namespace std;

void prim ( int n, int g[][100],int []);//initalizing the function

void prim ( int n, int g[][100],int lowcost[]) //prim's algorithm

{

  int closest[n];

  bool connected[n];

  int k, min, Cost = 0;

  int i,j;

  connected[0] = true;         // v0 is in the spanning tree.

  for (int i=1; i<n; ++i)

  {

    connected[i] = false;     // for false the vertex is not connected

    closest[i] = 0;       // unconnected vertex i

  }

  // Finding the MST the MST.

  cout << "Minimum cost spanning tree:" << endl;

  for (i=1; i<n; ++i)

  {

    // Finding the value of k not closest.

    min = INFINITY;

    for (j=1; j<n; ++j)

      if ( !connected[j] && lowcost[j] < min)

      {

        min = lowcost[j];

        k = j;

      }

    // Displaying the edges

    cout << "  edge = (v" << k << ",v" << closest[k] << "); cost = "

         << lowcost[k] << endl;

    Cost = Cost + lowcost[k];

    connected[k] = true;

    // updating the values

    for (j=1; j<n; ++j)

      if ( !connected[j] && g[k][j] < lowcost[j] )

      {

        lowcost[j] = g[k][j];

        closest[j] = k;

      }

  }

  cout << "Total cost = " << Cost << endl;

}

int main(){//The main function

   int i, j;

   int inp;

   int cost[7][100];//matrix

   int lowcost[7];

   int  data;

    ifstream myfile;

    cout<<"ENter the number of vertices you want: ";//asking input to the user

    cin>>inp;

 //initializing

      for (i = 0; i<inp; i++) { //starting the loop

              for (j = 0; j<inp; j++) {

                      cost[i][j] = INT\_MAX;

              }

       }

    myfile.open ("vertex.txt");//opening the file

   if (!myfile) {//display error message

        cout << "Unable to open file";

        exit(1); // terminate with error

    }

 while (!myfile.eof())//starting loop if there is no error

       {

              myfile >> i;//loading data from txt file

              myfile >> j;

              myfile >> data;

              cost[i][j] = data;//keeping the weight in matrix

              cost[j][i] = data;

              lowcost[i]=data;//Keeping the weight

       }

    myfile.close();//closing the file

prim(inp,cost,lowcost);//calling the function to evaulate prime

return 0;

}

**User Document**

The program can be used by the user by entering the number of vertices they want.

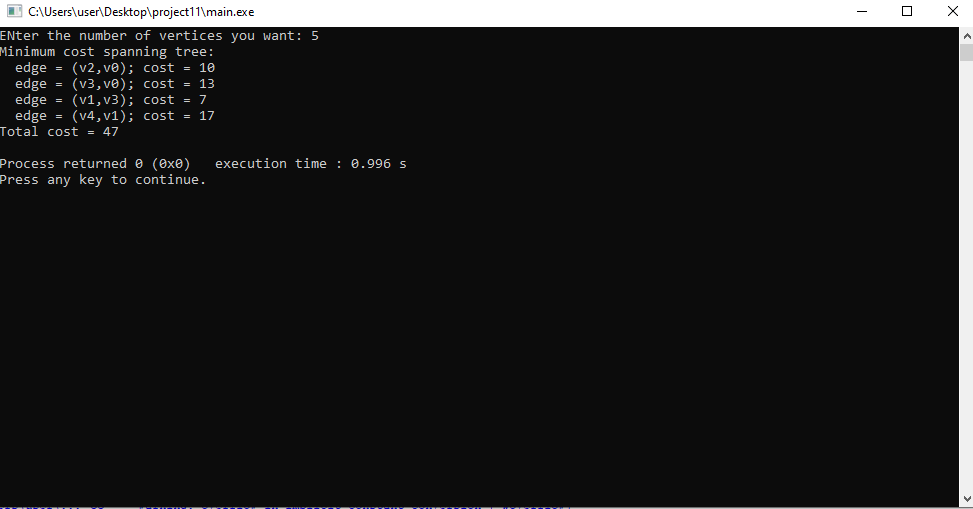
To compile the program simply enter:

*g++ -o main main.cpp*

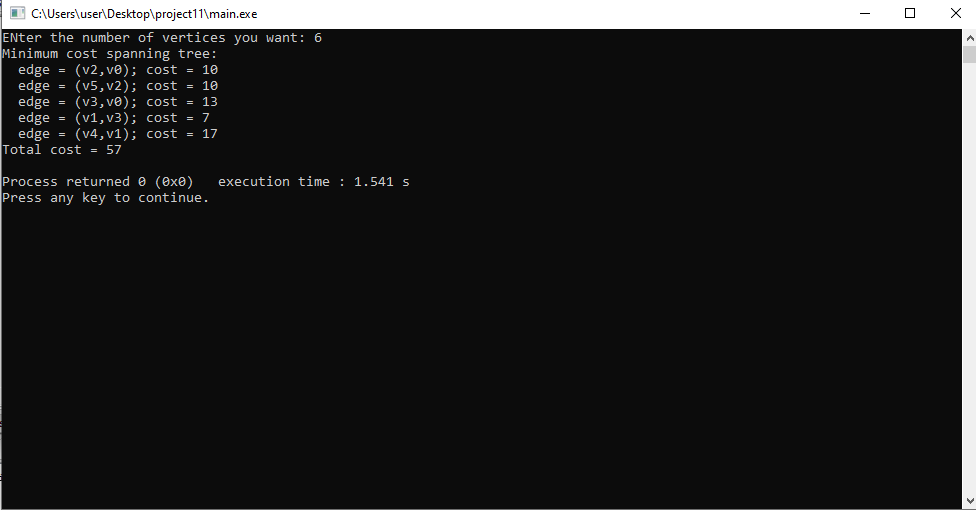
To run the program, enter. */main,* then you can input an integer. After entering an integer, the program will display the MST.

**Testing**

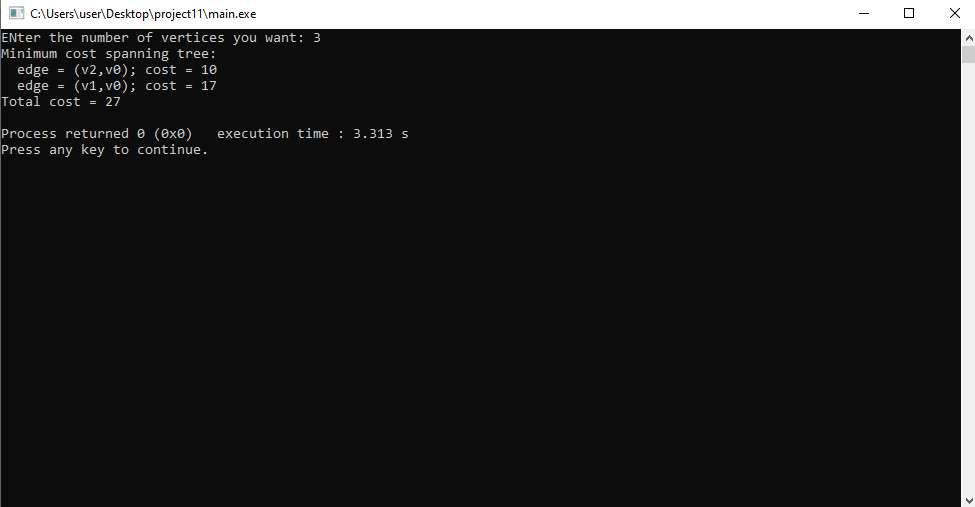
-Test 1



Test 2-



Test 3:



**Summary**

In this project, we have implemented an algorithm that finds the MST of an undirected edge-weighted graph that is called The Prim’s Algorithm. The program reads an input from the user. That determines the number of vertices the tree contain. The problems and their solutions involving Prim’s Algorithm that we learnt in class helped a to solve the given question.