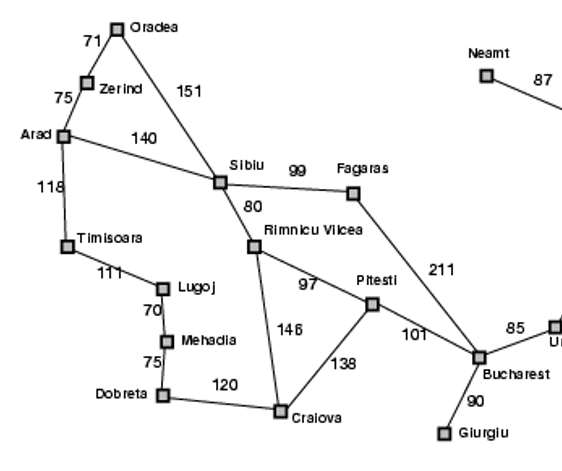
**Project 4**

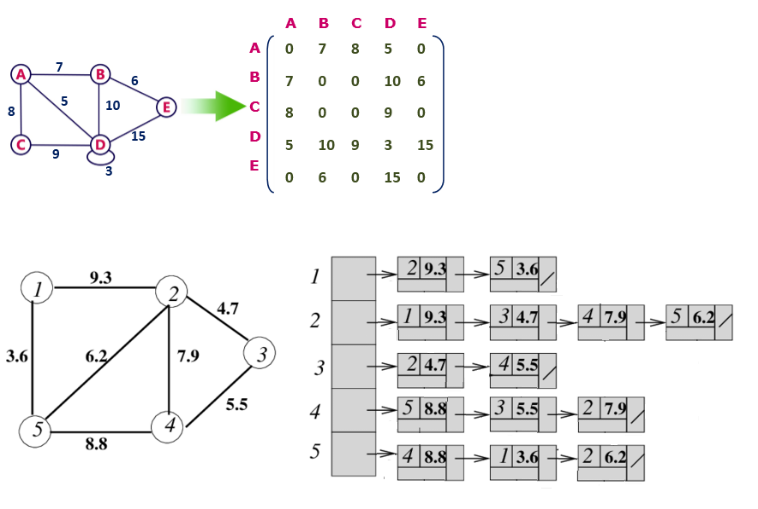
**Name: Boya Zhang Date:9/25/2020**

Write a program for finding the shortest path in a non-directed graph. The graph can be represented as a matrix or linked structure.

Apply Dijkstra’s algorithm. Implement it on the graph of Romania given in the lectures. To simplify your work, you may shorten the graph omitting some branches that are East of Bucharest.



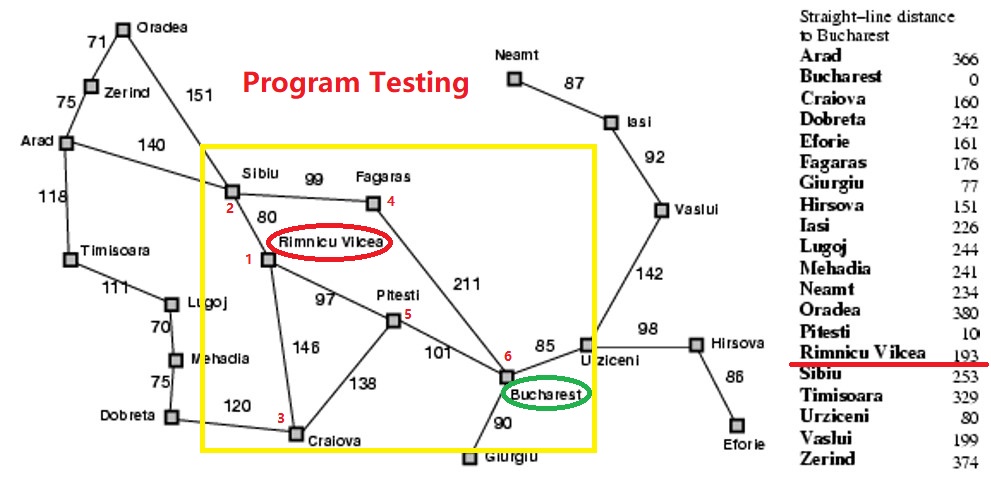
Submit the program and a screenshot of the input/output. If you want, you can submit the executable, but it is not required. If you have not passed a course on advanced programming, we will see how to do this with a system and you can use the system for this project.

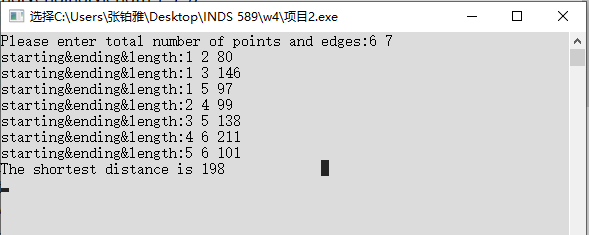
Examples of adjacency matrix (top) and adjacency list (bottom):

# Projects show

**Program test method:**

**Enter the number of points and edges from the starting point to the target point, then enter the starting point (marked as number 1) and the number of points after each step to automatically generate the shortest distance.**





#include <iostream>//You can find the code in main2.cpp file

/\* run this program using the console pauser or

add your own getch, system("pause") or input loop \*/

#include"iostream"

#include"cstring"

#include"cstdio"

using namespace std;

#define INF 0x7f7f7f7f

const int N = 105; //The upper limit of the number of points

int maze[N][N];

int dis[N];

bool vis[N];

//The number of points and the number of edges

int n,m;

void init()

{

memset(maze,INF,sizeof(maze));

memset(dis,INF,sizeof(dis));

memset(vis,false,sizeof(vis));

}

void dijkstra(int st)

{

dis[st]=0;

for(int i=1; i<=n; i++)

{

//Find the point with the shortest distance from the starting point

int minx=INF;

int minmark;

for(int j=1; j<=n; j++)

{

if(vis[j]==false&&dis[j]<=minx)

{

minx=dis[j];

minmark=j;

}

}

//And tag

vis[minmark]=true;

//Updates the distance of all points to which it is connected

for(int j=1; j<=n; j++)

{

if(vis[j]==false&&dis[j]>dis[minmark]+maze[minmark][j])

dis[j]=dis[minmark]+maze[minmark][j];

}

}

}

int main()

{ printf("Please enter total number of points and edges:");

while(scanf("%d %d",&n,&m)!=EOF)

{

if(n==0&&m==0) break;

//Each set of data is initialized

init();

for(int i=1; i<=m; i++)

{

int x,y,len;

printf("starting&ending&length:");

scanf("%d %d %d",&x,&y,&len);

if(x!=y&&maze[x][y]>len)

{

maze[y][x]=len;

maze[x][y]=len;

}

}

//Run a DIj starting at 1

dijkstra(1);

//Output distance to n

printf("The shortest distance is %d\n",dis[n]);

}}