ARQUIVO TXT: “oi.txt”

5 1

0 4

0 1 1

0 2 3

1 0 1

1 2 5

1 3 10

2 0 3

2 1 5

2 3 9

2 4 7

3 1 10

3 2 9

3 4 4

4 2 7

4 3 4

==================================

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int \*\*matriz;

int \*dist;

int \*marcados;

int \*precedentes;

int minimo;

int novoMinimo;

int destino;

int d;

int inicio;

int vertices;

int menor(vertices){

int i,menor=-1,d=10000000;

for(i=0;i<vertices;i++){

if(dist[i]<d && !marcados[i]){

menor=i;

break;

}

}

if(menor!=-1){

for(i=0;i<vertices;i++){

if(dist[i]<dist[menor] && !marcados[i]) menor=i;

}

}

return menor;

}

void Dijkstra(int inicio, int vertices){

int i,j,atual;

//definindo destino como ultimo vertice

destino=vertices-1;

dist=malloc(sizeof(int)\*vertices);

marcados=malloc(sizeof(int)\*vertices);

precedentes= malloc(sizeof(int)\*vertices);

dist[inicio]=0;

marcados[inicio]=1;

precedentes[inicio]=0;

for(i=1;i<vertices;i++){

marcados[i]=0;

precedentes[i]=-1;

if(matriz[inicio][i]==0){

dist[i]=1000; //INF

}

else{

dist[i]=matriz[inicio][i];

precedentes[i]=0; // se inicializados com valor da matriz, mostrar o zero

}

}

/\*

for(i=0;i<vertices;i++){

printf("DIST[%d]= %d\n",i,dist[i]);

}

\*/

while(menor(vertices)!=-1){

atual=menor(vertices);

//printf("EM %d:\n",atual);

marcados[atual]=1;

for(i=0;i<vertices;i++){

//printf("DIST[%d]: %d\n DITANCIA RELATIVA[%d]: %d\n",i,dist[i],i,dist[atual]+matriz[atual][i]);

if(matriz[atual][i]!=0 && marcados[i]==0){

if(dist[atual]+matriz[atual][i]<dist[i]) {

dist[i]=dist[atual]+matriz[atual][i];

precedentes[i]=atual;

}

}

}

for(i=0;i<vertices;i++){

printf("DIST[%d]= %d\n",i,dist[i]);

}

}

//recebe o destino anterior

d=destino;

while((destino)!=inicio){

matriz[precedentes[destino]][destino]=0;

//matriz[destino][precedentes[destino]]=0;

destino=precedentes[destino];

}

minimo=dist[d];

}

void DijkstraDois(int inicio, int vertices){

//printf(">>>>>>>> O D EH: %i\n", d);

int i,j,atual;

destino=vertices-1;

dist=malloc(sizeof(int)\*vertices);

marcados=malloc(sizeof(int)\*vertices);

precedentes= malloc(sizeof(int)\*vertices);

dist[inicio]=0;

marcados[inicio]=1;

precedentes[inicio]=0;

for(i=1;i<vertices;i++){

marcados[i]=0;

precedentes[i]=-1;

if(matriz[inicio][i]==0){

dist[i]=1000; //INF

}

else{

dist[i]=matriz[inicio][i];

precedentes[i]=0; // se inicializados com valor da matriz, mostrar o zero

}

}

/\*

for(i=0;i<vertices;i++){

printf("DIST[%d]= %d\n",i,dist[i]);

}

\*/

while(menor(vertices)!=-1){

atual=menor(vertices);

//printf("EM %d:\n",atual);

marcados[atual]=1;

for(i=0;i<vertices;i++){

//printf("DIST[%d]: %d\n DITANCIA RELATIVA[%d]: %d\n",i,dist[i],i,dist[atual]+matriz[atual][i]);

if(matriz[atual][i]!=0 && marcados[i]==0){

if(dist[atual]+matriz[atual][i]<dist[i]) {

dist[i]=dist[atual]+matriz[atual][i];

precedentes[i]=atual;

}

}

}

for(i=0;i<vertices;i++){

printf("DIST[%d]= %d\n",i,dist[i]);

}

}

d=destino;

while((destino)!=inicio){

matriz[precedentes[destino]][destino]=0;

//matriz[destino][precedentes[destino]]=0;

destino=precedentes[destino];

}

novoMinimo=dist[d];

}

int main(){

int i,j, valor;

int inicio=0, prox=0, vertices, linha, coluna,peso, x;

FILE \*p=fopen("oi.txt", "r");

if( !p ) {

printf("Erro com o arquivo\n");

return 1;

}

//printf("Digite a quantidade de vertices?\n");

//scanf("%d",&vertices);

fscanf(p, "%i %i", &vertices, &x);

matriz = malloc(sizeof(int\*)\*vertices);

for(i=0;i<vertices;i++){

matriz[i]=malloc(sizeof(int)\*vertices);

}

//printf("Digite os valores da matriz para cada posicao (0 se nao existir): \n");

for(i=0;i<vertices;i++){

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

(matriz)[i][j]=0;

//scanf("%d",&matriz[i][j]);

}

}

//INICIALIZANDO A MATRIZ COM ARQUIVO TXT

fscanf(p, "%i %i", &inicio, &destino);

while(fscanf(p, "%i %i %i", &linha, &coluna, &peso)!=EOF){ //mesmo que a anterior porem quando o grafo for direcionado

matriz[linha][coluna]= peso;

}

printf(">>>>>>>>> MATRIZ: \n");

for(i=0;i<vertices;i++){

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

printf(" %i", matriz[i][j]);

}

printf("\n");

}

Dijkstra(inicio, vertices);

//printf("IMPRIMINDO O VETOR DE DISTANCIAS A PARTIR DE %d:\n",inicio);

for(i=0;i<vertices;i++){

printf("Distancia[%d]: %d\n",i,dist[i]);

}

printf("\n\n");

for(i=0;i<vertices;i++){

printf("Precedente[%d]: %d\n",i,precedentes[i]);

}

printf("\n\n");

/\*

printf("MATRIZ BONITINHA DEPOIS DO DIJKSTRA\n");

for(i=0;i<vertices;i++){

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

printf(" %i", matriz[i][j]);

}

printf("\n");

}\*/

DijkstraDois(inicio, vertices);

if(minimo==novoMinimo){

DijkstraDois(inicio, vertices);

}

if(dist[d]==0){

printf(" -1 \n") ;

}else{

printf("O quase menor caminho eh: %i", dist[d]);

}

return 0;

}