1. Formularea enuntului problemei

Sa se conceapa un algoritm care sa faca conversia de la reprezentarea prin matricea de incidenta a unui graf la reprezentarea prin lista succesorilor.

2. Dezvoltarea algoritmului

Algoritmul de conversie din matricea de incidenta in lista succcesorilor presupune efectuarea urmatorilor pasi:

a. Citirea matricei de incidenta a grafului.

b. Crearea a 2 tablouri unidimensionale α si β cu semnificatiile urmatoare:

α– vector de pozitii; tabloul α are dimensiunea n+1 ; pentru fiecare varf i ϵ X ,α(i) indica adresa(indicele) din tabloul β unde sunt inregistrati succesorii sai .

β – vectorul ce contine succesorii varfului i . Acestia se afla intre in β intre pozitiile α(i) si α(i+1) - i .

3. Descrierea algoritmului

In primul rand se citeste matricea de incidenta . De asemenea avem grija ca α(1) sa ia valoarea 1.

Apoi se parcurge aceasta matrice pe linii si unde gaseste valoarea 1 , trece la coloanele corespunzatoare pentru a cauta valoarea -1 si pentru fiecare valoare de -1 intalnita se va adauga indicele coloanei in vectorul β.

La schimbarea liniei se va retine in vetorul α pozitia ultimului element

memorat in β plus 1.

*citeste n,m ((bij, i=1,n) , j=1,m);*

*k:=1;*

*pentru i:=1,n executa*

*α(i):=k;*

*pentru u:=1,m executa*

*daca biu=1 atunci*

*pentru j:=1,m executa*

*daca biu=-1 atunci q:=j sfdaca;*

*sfpentru;*

*β(k):=q;*

*k:=k+1;*

*sfdaca;*

*sfpentru;*

*sfpentru ;*

*α(n+1):=m+1;*

4. Demonstrarea corectitudinii algoritmului

Tot ceea ce am folosit se gaseste in manual .

5. Cod sursa

#include <iostream>

#include <fstream>

using namespace std;

int in[10][20],n,m,alfa[10],beta[10];

void citire()

{

ifstream f("incidenta.in");

f>>n>>m;

int i,j;

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

for(j=1;j<=m;j++)f>>in[i][j];

f.close();

}

void transf()

{

int k=0;

int vf,i,j;

for(vf=1;vf<=n;vf++)

{alfa[vf]=k+1;

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

if(in[vf][i]==1)

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

if(in[j][i]==-1)

{

k++;beta[k]=j;

}

}

}

void afissir(int a[20],int n)

{

int i;

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

cout<<a[i]<<" ";

cout<<endl;

}

void afissucc(int vf)

{int i,c=0;

cout<<"succesorii nodului "<<vf<<" sunt:";

for(i=alfa[vf];i<alfa[vf+1];i++)

{cout<<beta[i]<<" ";c++;}

if(c==0)cout<<"Nu exista";

cout<<endl;

}

int main()

{int i;

citire();

transf();

cout<<"alfa:";afissir(alfa,n);

cout<<"beta:";afissir(beta,m);

cout<<"Lista succesorilor este:";

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

afissucc(i);

return 0;

}

6. Date de test

|  |  |
| --- | --- |
| Date de intrare | Date de iesire |
| 1. 3  3  0 1 -1  -1 0 1  1 -1 0 | alfa: 1 2 3  beta : 3 1 2  Lista succesorilor este :  Succesorii nodului 1 sunt: 3  Succesorii nodului 2 sunt: 1  Succesorii nodului 3 sunt: nu exista |
| 2. 5  6  1 1 -1 0 0 0  -1 0 1 1 0 -1  0 -1 0 -1 0 0  0 0 0 0 1 1  0 0 0 0 -1 0 | alfa: 1 3 5 5 7  beta : 3 1 3 5 2  Lista succesorilor este :  Succesorii nodului 1 sunt: 2 3  Succesorii nodului 2 sunt: 1 3  Succesorii nodului 3 sunt: nu exista  Succesorii nodului 4 sunt: 5 2  Succesorii nodului 5 sunt: nu exista |
| 3. 4  6  -1 1 0 0 1 0  1 0 0 0 -1 1  0 0 1 -1 0 0  0 -1 -1 1 0 -1 | alfa: 1 3 5 6  beta : 4 2 1 4 4 3  Lista succesorilor este :  Succesorii nodului 1 sunt: 4 2  Succesorii nodului 2 sunt: 1 4  Succesorii nodului 3 sunt: 4  Succesorii nodului 4 sunt: nu exista |