Ministerul Educaţiei, Culturii și Cercetării

al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Ingineria Software şi Automatică

Lucrarea de laborator Nr.1

la Matematica specială

Tema : Păstrarea grafurilor în memoria calculatorului

A efectuat : gr. SI – 201 , Ivanova Evghenia

A verificat : asis. univ. Popovici Nadejda

Chişinău – 2020

**Scopul Lucrării :**

Studierea metodelor de definire a unui graf : Matricea de incidență , Matricea de adiacență , Listă de adiacenţă .

Elaborarea unor proceduri de introducere, extragere și transformare a diferitelor forme de reprezentare internă a grafurilor cu scoaterea rezultatelor la display și imprimantă .

**Sarcina Lucrării :**

1. De elaborate procedura întroducerii unui graf în memoria calculatorului în formă de matrice de incidenţă, matrice de adiacenţă şi listă de adiacenţă cu posibilităţi de analiză a corectitudinii .

2. Elaboraţi procedura de transformare dintr-o formă de reprezentare în alta .

3. Folosind procedurile mentionate, elaboraţi porogramul care va permite :

 a) întroducerea grafului reprezentat sub oricare din cele trei forme cu posibilităţi de corecţie a datelor;

 b) păstrarea grafului în memoria externă în forma de listă de adiacenţă;

 c) extragerea informaţiei în una din cele trei forme la imprimantă si display.

**Codul programului :**

#include <stdio.h>

#include <stdlib.h>

typedef struct arc{

int vi;

int vf;

}arc;

int\*\* aloc(int n,int m)

{

int \*\*vect=NULL;

vect=(int\*\*)malloc(n\*sizeof(int\*));

if(vect==NULL) return vect;

for(int i=0;i<n;i++)

{

vect[i]=(int\*)malloc(m\*sizeof(int));

if(vect[i]==NULL) return NULL;

}

return vect;

}

void mat\_in(arc \*a,int \*\*vect,int u,int x)

{

for(int i=0;i<u;i++)

for(int j=0;j<x;j++)

{

if(a[i].vi==a[i].vf && a[i].vi==j+1) vect[i][j]=2;

else if(a[i].vi==j+1) vect[i][j]=-1;

else if(a[i].vf==j+1) vect[i][j]=1;

else vect[i][j]=0;

}

}

void mat\_ad(arc \*a,int \*\*vect,int u,int x)

{

int k;

for(int i=0;i<x;i++)

for(int j=0;j<x;j++)

{

for(int l=0;l<u;l++)

if(a[l].vi==i+1 && a[l].vf==j+1)

{

vect[i][j]=1;

k++;

}

if(!k) vect[i][j]=0;

k=0;

}

}

void li(arc \*a,int \*\*LA,int u,int x)

{

for(int i=0;i<x;i++)

{

LA[i][0]=i+1;

int k=1;

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

{

for(int l=0;l<u;l++)

if(a[l].vi==i+1 && a[l].vf==j)

{

LA[i][k]=j;

k++;

}

}

LA[i][k]=0;

}

}

void outputm(int \*\*vect,int n, int m)

{

for (int i=0; i<m; i++)

printf("\tx%d",i+1 );

printf("\n\n");

for(int i=0;i<n;i++)

{

printf("u%d", i+1);

for(int j=0;j<m;j++)

printf("\t%d",vect[i][j]);

printf("\n\n");

}

}

void outputl(int \*\*vect,int n,int m)

{

for(int i=0;i<n;i++)

{

printf("%d -> ",vect[i][0]);

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

{

printf(" %d, ",vect[i][j]);

if(vect[i][j]==0) break;

}

printf("\n");

}

}

void main(void)

{

arc \*a=NULL;

int \*\*MI=NULL, \*\*MA=NULL, \*\*LA=NULL, i, u=0, x=0, ar, op;

while (1)

{

printf("\t\tMENIU");

printf("\n\t1.Introdu datele de intrare");

printf("\n\t2.Corecția datelor");

printf("\n\t3.Afisarea matricei incidente");

printf("\n\t4.Afisarea matricei adiacente");

printf("\n\t5.Afisarea listei adiacente");

printf("\n\t0.Iesire");

printf("\n\n\tAlege optiunea:"); scanf("%d", &op);

switch (op)

{

case 1: printf("\n\tIntrodu nr. de arce u ="); scanf("%d",&u);

printf("\tIntrodu nr. de varfuri x ="); scanf("%d",&x);

a=(arc\*)malloc(u\*sizeof(arc));

for(i=0;i<u;i++)

{

p: printf("Introdu datele arcului %d\n",i+1);

printf("Introdu X initial:"); scanf("%d",&a[i].vi);

printf("Introdu X final:"); scanf("%d",&a[i].vf);

if((a[i].vi<1) || (a[i].vi>x) || (a[i].vf<1) || (a[i].vf>x))

{

printf(" ! VALOARE INCORECTA !");

goto p;

}

}

break;

case 2: for(int i=0;i<u;i++)

{

printf("Datele arcului%d\n",i+1);

printf("%d -> %d\n",a[i].vi,a[i].vf);

}

printf("\nIntrodu nr. arcului pentru modificare:"); scanf("%d",&ar);

if(!(ar >= 1 && ar <= u)) printf("Arcul nu exista!\n");

printf("Datele arcului%d\n%d -> %d\n\n",ar,a[ar-1].vi,a[ar-1].vf);

printf("Introdu X initial:"); scanf("%d",&a[ar-1].vi);

printf("Introdu X final:"); scanf("%d",&a[ar-1].vf);

break;

case 3: printf("\n\t\tMATRICE DE INCIDENTA\n\n");

if(MI) outputm(MI,u,x);

else{

MI=aloc(u,x);

mat\_in(a,MI,u,x);

outputm(MI,u,x);

}

break;

case 4: printf("\n\t\tMATRICE DE ADIACENTA\n\n");

if(MA) outputm(MA,u,x);

else{

MA=aloc(x,x);

mat\_ad(a,MA,u,x);

outputm(MA,x,x);

}

break;

case 5: printf("\n\t\tLISTA DE ADIACENTA\n\n");

if(LA!=NULL) outputl(LA,x,x+2);

else{

LA=aloc(x,x+2);

li(a,LA,u,x);

outputl(LA,x,x+2);

}

break;

case 0: exit(0);

default:printf("\tAti introdus o comanda inexistenta!\n\n");

break;

}

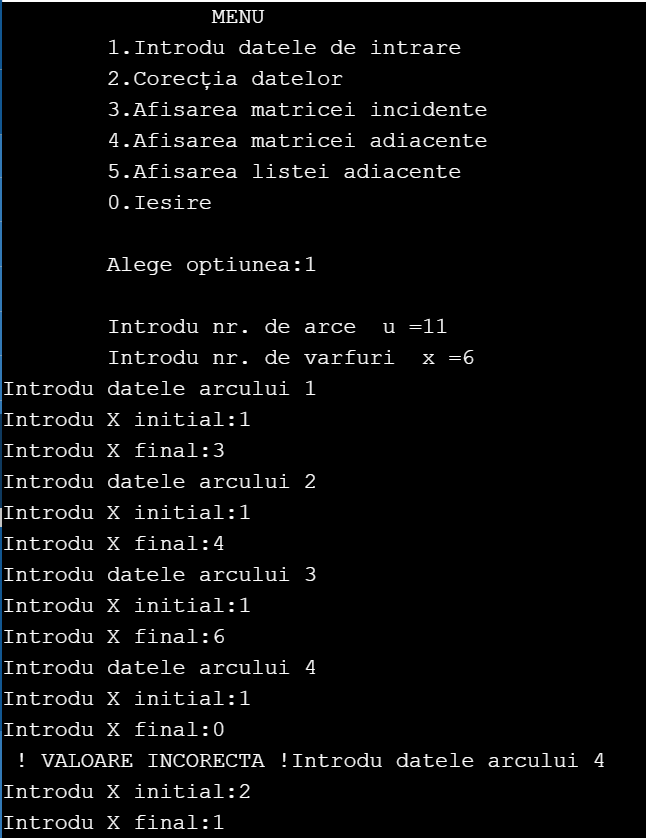
}

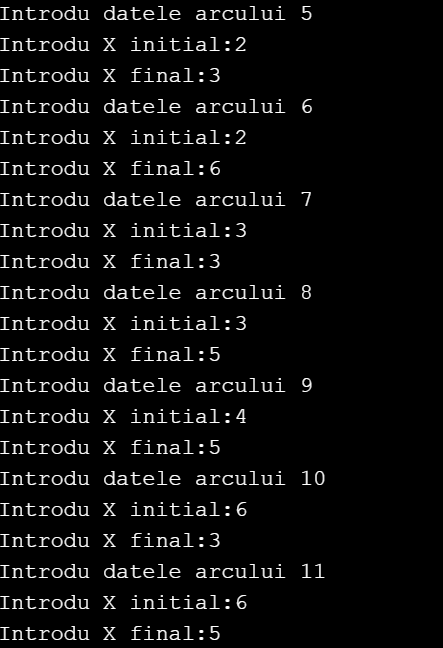
}

**Execuția Programului :**

**Afișarea meniului și executarea primei opțiuni de introducere a datelor necesare**

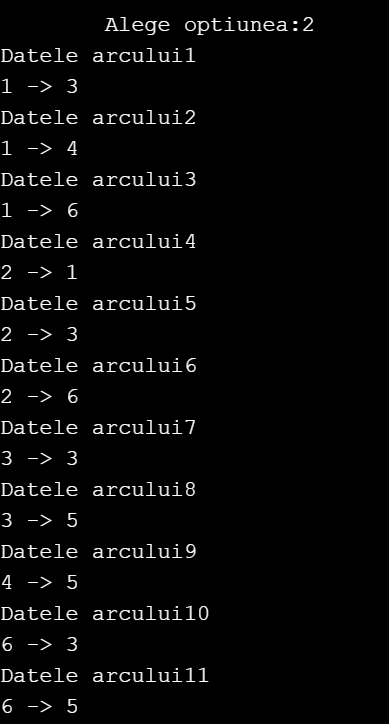
**În caz că ați introdus un vârf inexistent – apare un mesaj de alarmă și ai posibilitatea de a introduce datele din nou .**

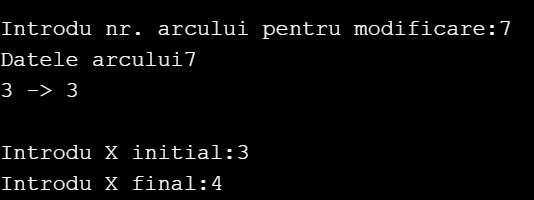




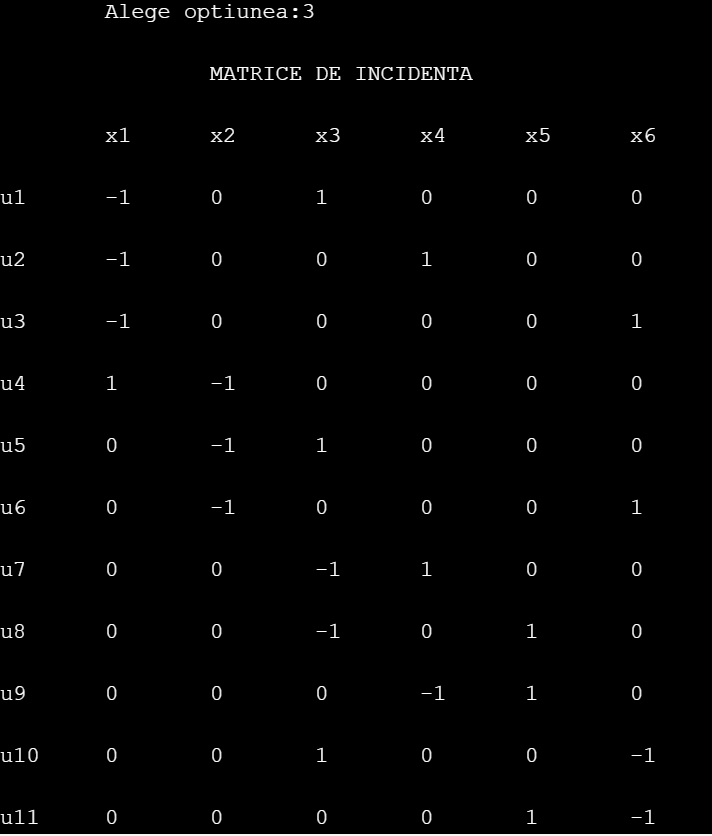
**Executarea opțiunii doi care permite modificarea datelor de intrare**

**Aceasta afișează arcurile existente și cere să introduci în ce arc faci corecții**

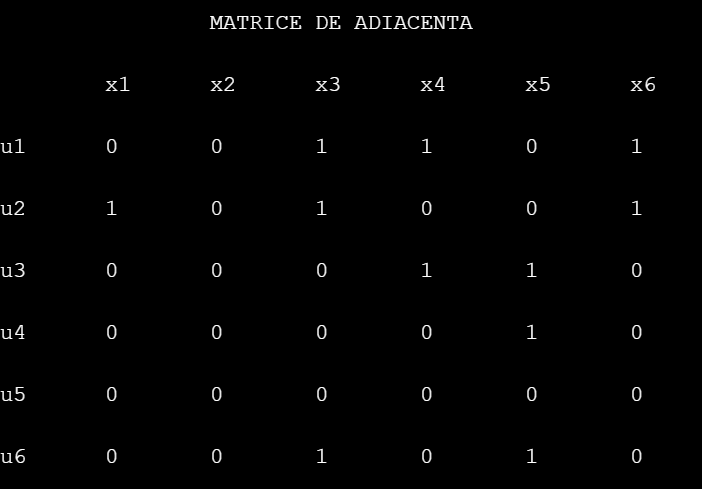




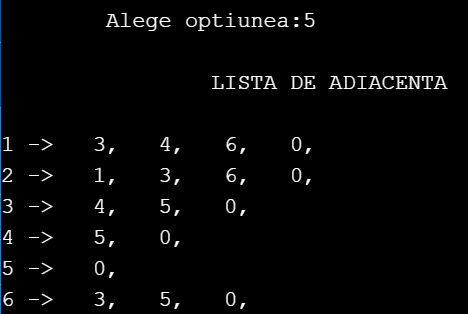
**Executarea opțiunii trei care afișează la ecran matricea de incidență**



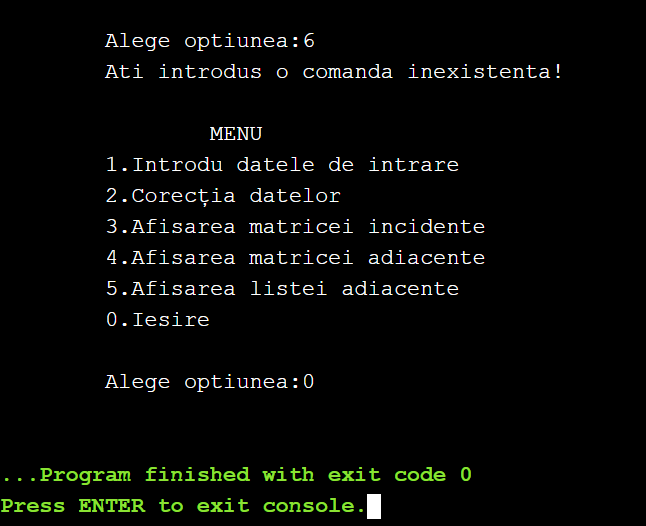
**Executarea opțiunii patru care afișează la ecran matricea de adiacență**



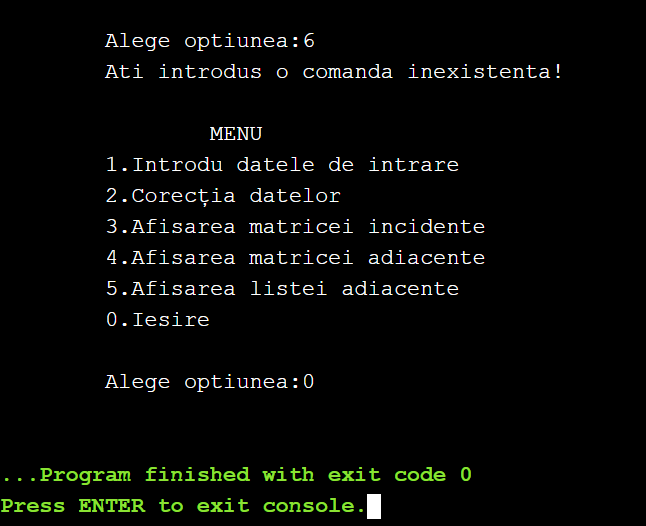
**Executarea opțiunii cinci care afișează la ecran lista de adiacență**



**Executarea opțiunii care nu este în meniu , de exemplu șase , afișează la ecran un mesaj de alertă pentru o comandă inexistentă**



**Executarea opțiunii zero care iese din program**



**Concluzia :**

În urma acestei lucrări de laborator am deprins tehnica procedurii de introducere a grafului sub una din cele trei forme cu posibilități de corecție a datelor și salvarea acestuia in memoria externă a calculatorului pentru a putea fi extrase pe ecran .

Teoria grafurilor a luat naştere de la problema podurilor din Konigsberg , cercetată de Euler . După ea apare în mai multe domenii practice , cum ar fi potrivirea șabloanelor , planificarea evenimentelor sportive , proiectarea locurilor unor persoane la evenimente , orarele de examen , programarea taxiurilor , și rezolvarea puzzle-urilor [Sudoku](https://ro.wikipedia.org/wiki/Sudoku) …

De exemplu în inginerie – trasarea rețelelor de comunicație , planificarea rutelor de transport … În economie – rezolvarea problemelor ce implica determinarea unor drumuri critice . În matematică – baza teoriei mulțimilor . În psihologie – reprezentarea relațiilor interumane . Și încă sunt foarte multe domenii care se intersectează cu acastă teorie .