**Ministerul Educaţiei Tineretului şi Sportului al Republicii Moldova**

**Universitatea Tehnică a Moldovei**

**RAPORT**

**LA MATEMATCA DISCRETA**

Tema: PĂSTRAREA GRAFURILOR ÎN MEMORIA CALCULATORULUI

Lucrarea de Laborator nr. 1

A efectuat Studentul grupei \_SI-212\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_Vozian Vladimir\_\_\_\_

*semnătura nume, prenume*

A verificat \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*nume, prenume profesor*

**Chisinău 2021**

1. **SCOPUL LUCRĂRII:**

* Studierea metodelor de definire a unui graf: matrice de incidenţă, matrice de adiacenţă, liste;
* Elaborarea unor proceduri de introducere, extragere şi transformare a diferitor forme de reprezentare internă a grafurilor cu scoaterea rezultatelor la display şi imprimantă.

1. **SARCINA DE BAZĂ**

1. Elaboraţi procedura introducerii 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 proceduri de transformare dintr-o formă de reprezentare în alta.

3. Folosind procedurile menţionate elaboraţi programul care va permite:

* introducerea grafului reprezentat sub oricare din cele trei forme cu posibilităţi de corecţie a datelor;
* păstrarea grafului în memoria externă în formă de listă de adiacenţă;
* extragerea informaţiei într-una din cele trei forme la imprimantă şi display.

import numpy as np

import pandas

import os

#citirea matricea de adiacenta

def citireAdiacenta():

    nods = int(input("Cate noduri are matricea? "))

    adiacenta = np.zeros((nods, nods), dtype = int)

    legaturi = int(input("Cate legaturi doriti sa aveti? "))

    for i in range(0, legaturi):

        rand, col = map( int, input("Dati legatura: ").split())

        while not (rand > 0 and rand <= nods and col > 0 and col <= nods):

            print("Date introduse gresit.")

            rand, col = map( int, input("Dati legatura: ").split())

        adiacenta[rand-1][col-1] = 1

    return adiacenta

#afisarea matricei de adiacenta

def afisareAdiacenta(adiacenta):

    rand = adiacenta.shape[0]

    print("\nMatrice de adiacenta:")

    col = []

    for i in range(1,rand+1):

        col.append('x'+str(i))

    adiacenta = pandas.DataFrame(adiacenta, columns = col,index=col)

    print(adiacenta)

#citirea matrice de incidenta

def citireIncidenta():

    col, rand = map(int, input("Cate noduri si muchii vor fi? ").split())

    incidenta = np.zeros((rand, col), dtype = int)

    for i in range(0, rand):

        iesire, intrare = map( int, input("Dati legatura dintre 2 noduri: ").split())

        while not (intrare > 0 and intrare <= col and iesire > 0 and iesire <= col):

            print("Date introduse gresit.")

            iesire, intrare = map( int, input("Dati legatura dintre 2 noduri: ").split())

        incidenta[i][iesire-1] = -1

        incidenta[i][intrare-1] = 1

    return incidenta

#afisarea matricei de incidenta

def afisareIncidenta(incidenta):

    print("\nMatrice de incidenta:")

    rand, col = incidenta.shape

    coloane = []

    for i in range(1,col+1):

        coloane.append('x'+str(i))

    randuri = []

    for i in range(1,rand+1):

        randuri.append('u'+str(i))

    incidenta = pandas.DataFrame(incidenta, columns = coloane, index = randuri)

    print(incidenta)

#citire lista

def citireLista():

    nods = int(input("Cate noduri are matricea? "))

    lista = np.zeros(nods, dtype = object)

    for i in range(0, nods):

        l = []

        n = int(input("Cate legaturi sunt pentru nodul " + str(i+1) + ":"))

        if (n != 0):

            l = list(map(int,input("Dati legaturile : ").strip().split()))[:n]

            while not all(i <= nods for i in l):

                print("Date introduse gresit.")

                l = []

                l = list(map(int,input("Dati legaturile : ").strip().split()))[:n]

        l.append(0)

        lista[i] = l

    return lista

#afisare lista

def afisareLista(lista):

    nods = lista.shape[0]

    print("Lista adiacenta:")

    for i in range(1,nods+1):

        print(i,'- ',end='')

        for j in lista[i-1]:

            if(j != 0):

                print(str(j)+'\_',end = '')

            else:

                print(j)

#Lista in adiacent matrix

def Lista\_in\_adiacenta(lista):

    nods = lista.shape[0]

    adiacenta = np.zeros((nods, nods) , dtype = int)

    for i in range(0, nods):

        for j in lista[i]:

            if(j != 0):

                adiacenta[i][j-1] = 1

    return adiacenta

#Lista in incident matrix

def Lista\_in\_incidenta(lista):

    col = lista.shape[0]

    rand = 0

    for i in range(0,col):

        rand += len(lista[i]) - 1

    incidenta = np.zeros((rand,col), dtype = int)

    u = 0

    for i in range (0,col):

        for j in lista[i]:

            if (j != 0):

                incidenta[u][i] = -1

                incidenta[u][j-1] = 1

                u += 1

    return incidenta

#Adiacenta in incidenta

def Adiacenta\_in\_Incidenta(adiacenta):

    rand = np.count\_nonzero(adiacenta == 1)

    col = adiacenta.shape[0]

    incidenta = np.zeros((rand,col), dtype = int)

    u = 0

    for i in range (0,col):

        for j in range(0,col):

            if adiacenta[i][j] == 1:

                incidenta[u][i] = -1

                incidenta[u][j] = 1

                u += 1

    return incidenta

#Adiacenta in lista

def Adiacenta\_in\_Lista(adiacenta):

    rand = adiacenta.shape[0]

    lista = []

    for i in range(0,rand):

        l\_temp = []

        for j in range(0,rand):

            if adiacenta[i][j] == 1:

                l\_temp.append(j+1)

        l\_temp.append(0)

        lista.append(l\_temp)

    return np.asarray(lista, dtype = object)

#Incidenta in adiacenta

def Incidenta\_in\_Adiacenta(incidenta):

    rand, col = incidenta.shape

    adiacenta = np.zeros((col,col), dtype = int)

    for i in range(0,rand):

        for j in range(0,col):

            if incidenta[i][j] == -1:

                iesire = j

            if incidenta[i][j] == 1:

                intrare = j

        adiacenta[iesire][intrare] = 1

    return adiacenta

#Incidenta in Lista

def Incidenta\_in\_Lista(incidenta):

    rand, col = incidenta.shape

    lista = []

    for i in range(0,col):

        lista.append([])

    for i in range(0,rand):

        a = np.where(incidenta[i] == -1)

        b = np.where(incidenta[i] == 1)

        lista[a[0][0]].append(b[0][0] + 1)

    for i in range(0,col):

        lista[i].append(0)

    return np.asarray(lista, dtype = object)

#main craft

option = input("1.Matricea de adiacenta.\n2.Matricea de incidenta.\n3.Lista\nIn ce tip doriti sa transformati?  ")

match option:

    case "1":

        adiacenta = citireAdiacenta()

        afisareAdiacenta(adiacenta)

        incidenta = Adiacenta\_in\_Incidenta(adiacenta)

        afisareIncidenta(incidenta)

        lista = Adiacenta\_in\_Lista(adiacenta)

        afisareLista(lista)

    case "2":

        incidenta = citireIncidenta()

        afisareIncidenta(incidenta)

        adiacenta = Incidenta\_in\_Adiacenta(incidenta)

        afisareAdiacenta(adiacenta)

        lista = Incidenta\_in\_Lista(incidenta)

        afisareLista(lista)

    case "3":

        lista = citireLista()

        afisareLista(lista)

        adiacenta = Lista\_in\_adiacenta(lista)

        afisareAdiacenta(adiacenta)

        incidenta = Lista\_in\_incidenta(lista)

        afisareIncidenta(incidenta)

## Exemple de executie a programului:

## 

## A fost introdusă matricea de adiacență

## 

## A fost introdusă matricea de incidență

## 

## A fost introdusă lista

## Controlarea corectitudinii introducerii datelor a fost facută direct la citirea matricei

## 

## Exemplu de date introduse incorect dupa analiza carora programul afisează eroarea și cere din nou introducerea datelor.

## Concluzii:

Lucrarea dată era efectuată in python, în timpul lucrării sa lucrat cu cele 3 forme de reprezentare a grafurilor și trecerea dintr-o formă în alta elaborând un program. Dupa executarea acestui laborator mam familiarizat cu grafurile și cu programarea în baza lor.