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

**Universitatea Tehnică a Moldovei**

**RAPORT**

**LA MATEMATCA DISCRETA**

Tema: ALGORITMUL DE CĂUTARE ÎN LĂRGIME

Lucrarea de Laborator nr. 3

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

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

*semnătura nume, prenume*

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

*nume, prenume profesor*

**Chisinău 2021**

1. **SCOPUL LUCRĂRII:**

### Studierea algoritmului de căutare în lărgime;

* Elaborarea programului de căutare în lărgime.

1. **SARCINA DE BAZĂ**
2. Elaboraţi procedura care va realiza algoritmul de parcurgere a grafului în lărgime;
3. Folosind procedurile din lucrările precedente, elaboraţi programul care va permite:

* introducerea grafului în calculator;
* parcurgerea grafului în lărgime;
* extragerea datelor la display şi printer.

import numpy as np

vizitate = []

def Largimea(lista, initial):

    vizitat = []

    asteptare = []

    print("\nParcurgerea in largime a grafului este : ", end='')

    vizitat.append(initial - 1)

    asteptare.append(initial - 1)

    while asteptare:

        nod = asteptare.pop(0)

        print(nod + 1, end =' ')

        for vecini in lista[nod][0:len(lista[nod]) - 1]:

            if vecini - 1 not in vizitat:

                vizitat.append(vecini - 1)

                asteptare.append(vecini - 1)

#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 = citireLista()

afisareLista(lista)

init = int(input("Introduceti nodul initial =>"))

Largimea(lista,init)

## Exemple de executie a programului:

## Graful a fost introdus în formă de listă de adiacență, ca exemplu de execuție a fost ales exemplul parcurgerii la curs:

## 

## 

## Observăm că rezultatele coincid, deci programul funcționează corect.

## Concluzii:

Lucrarea dată era efectuată in python, sa exersat asupra cunoștiințelor parcurgerii grafurilor și anume parcurgerea grafului în lărgime cu efectuarea lui prin program, sa lucrat anume cu introducerea grafului prin lista adiacentă prin comoditatea ei.