МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №4

З дисципліни

“Дискретна математика”

Виконала:

студентка групи КН-112

Тимчишин Марта

Перевірила:

Мельникова Н.І.

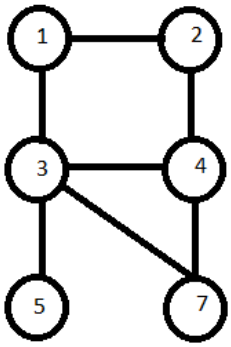
Львів – 2019 р.

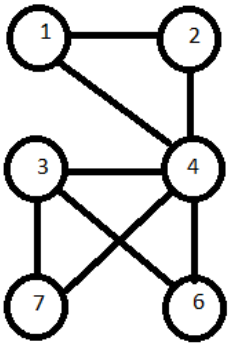
Тема: Основні операції над графами. Знаходження остова мінімальної ваги за алгоритмом Прима-Краскала.

Мета роботи: набуття практичних вмінь та навичок з використання алгоритмів Прима і Краскала.

Варіант 13

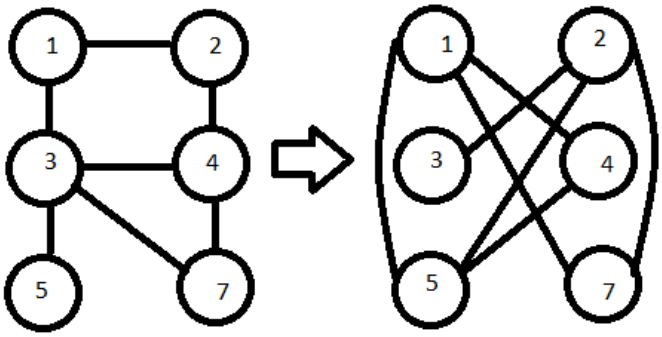
1. Виконати наступні операції над графами



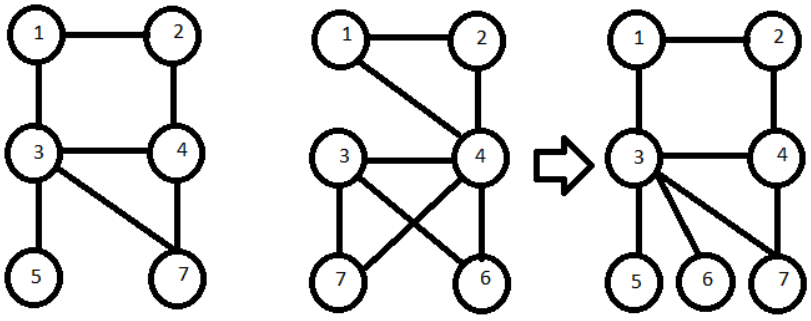


G1 G2

1.1 Знайти доповнення до першого графу

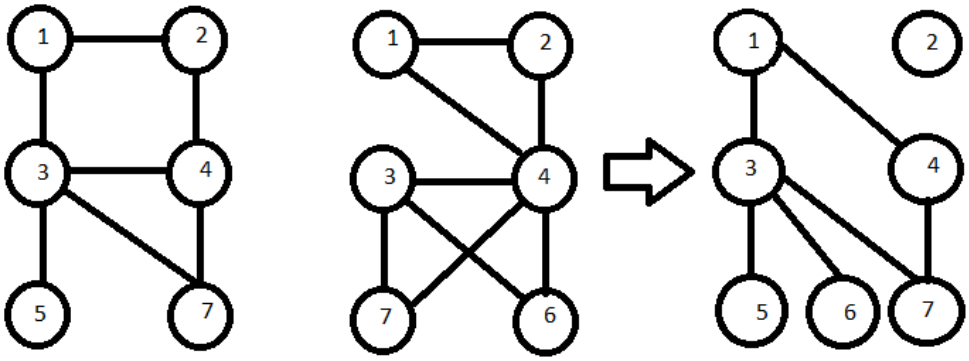


1.2 Знайти об’єднання графів G1 та G2



G1 G2

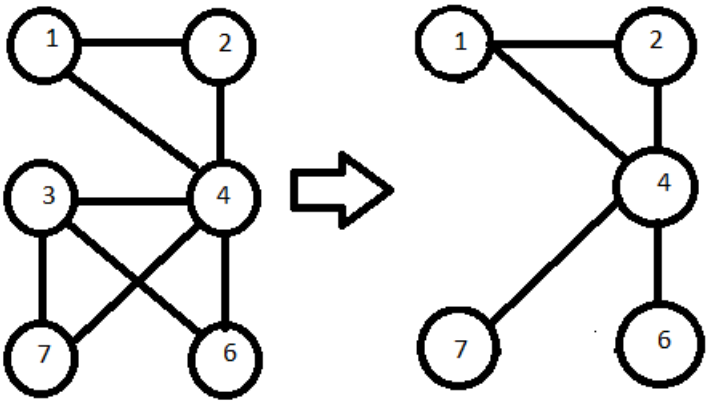
1.3 Знайти кільцеву суму G1 та G2



G1 G2

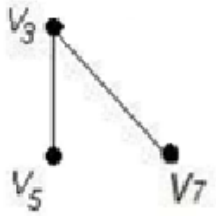
1.4 Розчепити вершину у G2

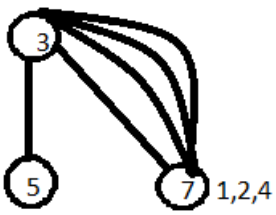
Знищуємо вершину 3, разом з вершиною зникають і всі інцидентні до неї ребра



1.5 Виділити підграф A, що складається з 3-х вершин в G1 і знайти стягнення A в G1 (G1\ A).

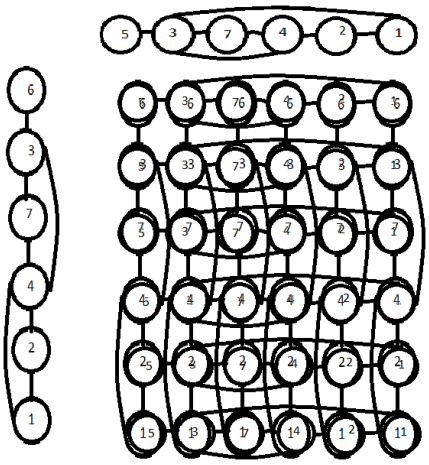
Підграф:



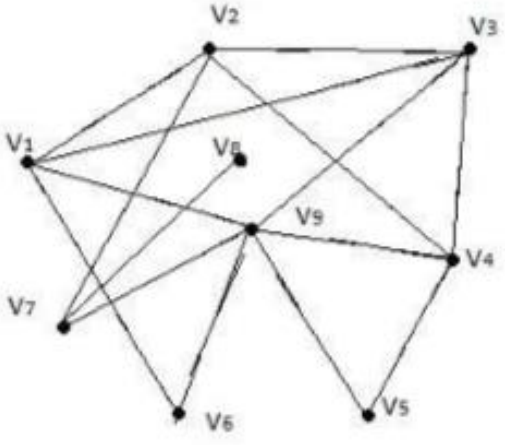


Стягнення :

1.6 добуток графів



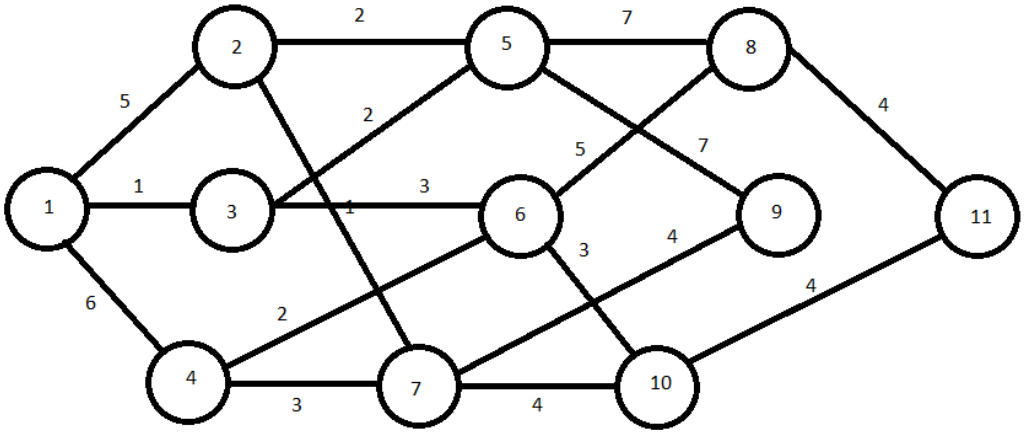
2.0 Знайти таблицю суміжності , та діаметр графа



	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
1	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**1**
2	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
3	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**1**
4	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**1**
5	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**
6	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
7	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
8	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
9	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**

Діаметр : 3

3 Знайти двома методами (Краскала і Прима) мінімальне остове дерево графа.



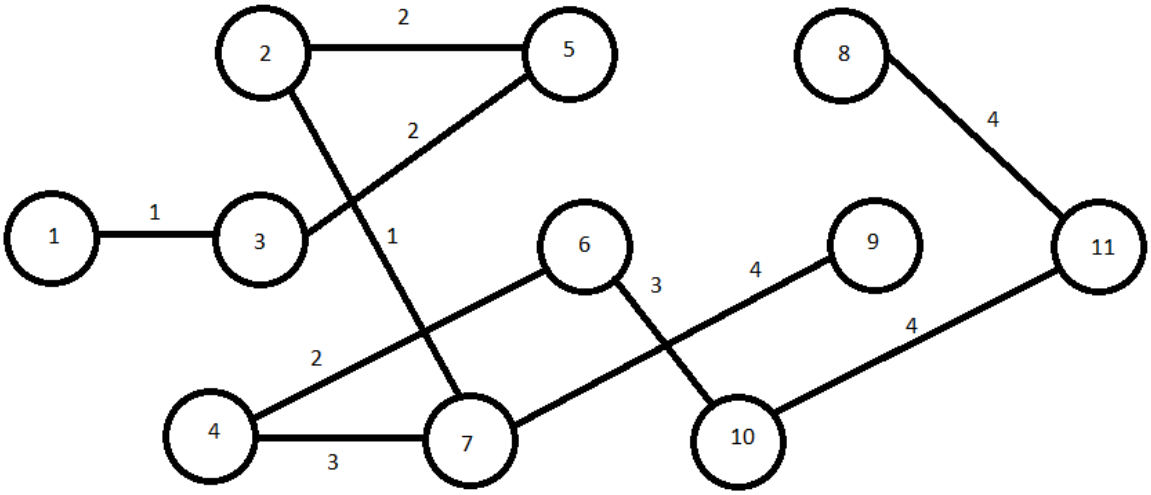
Прима :

(V1 V3) > (V3 V5) > (V5 V2) > (V2 V7) > (V7 V4) > (V4 V6) > (V6 V10) > (V10 V11) > (V11 V8) > (V7 V9)

для алгоритма Краскала:

(V1 V3) > (V2 V7) > (V3 V5) >(V5 V2)> (V4 V6)> (V7 V4) > (V6 V10)> (V10 V11)> (V11 V8) >(V7 V9)

Мінімальне остове дерево :



Код програми

```
#include <iostream>
using namespace std;

int main()
{

	int v, count = 0, min = 0, k, t;
	bool check = false;
	cout << "The number of vertices of the graph : ";
	cin >> v;
	int* tops = new int[v];
	//для матриці ваг
	int** matrix = new int* [v];
	for (int i = 0; i < v; i++) {
		matrix[i] = new int[v];
	}
	//для ребер
	int** rebra = new int* [v - 1];

	for (int i = 0; i < v - 1; i++) {
		rebra[i] = new int[2];
	}
	//ввід матриці ваг
	for (int i = 0; i < v; i++) {
		for (int j = 0; j < v; j++) {
			cin >> matrix[i][j];
		}
	}
	//беремо вершину один як початкову

	tops[count] = 1;
	count++;

	for (int i = 0; count < v; i++) {
		for (int j = 0; j < count; j++) {
			for (int a = 0; a < v; a++) {
				for (int m = 0; m < count; m++) {
					//перевірка чи вершина вже була пройде, якщо так, то
переходимо до наступної ітерації
					if (tops[m] == a + 1) {
						check = true;
					}
				}

				if (check) { check = false; continue; }
				//шукаємо мінімальний елемент в рядку, спочатку береться
просто перший ненульовий елемент, а потім вже шукаєм відносно нього, це в наступному іфі
				if (min == 0 && matrix[tops[j] - 1][a] > 0) {
					min = matrix[tops[j] - 1][a];
					k = rebra[count - 1][0] = tops[j]; t = rebra[count -
1][1] = a + 1;
					continue;
				}

				if (matrix[tops[j] - 1][a] > 0 && matrix[tops[j] - 1][a] <
min) {
					min = matrix[tops[j] - 1][a];
					//записуємо ребро
					k = rebra[count - 1][0] = tops[j]; t = rebra[count -
1][1] = a + 1;
				}
			}
		}
```

```
            //Обнулюємо ребро, бо вже не можемо по ньому проходити і маємо опускати в
подальшому пошуку
            matrix[k - 1][t - 1] = 0; matrix[t - 1][k - 1] = 0;
            //Додаємо знайдену вершину в масив вершин
            tops[count] = t;
            count++;
            min = 0;
      }
      //output
  //Виводимо наш масив вершин, який вийшов, які послідовно додавались
      cout << "V: { ";

      for (int j = 0; j < v; j++) {
            cout << tops[j] << ", ";
      }

      cout << "}";
      //і виводимо ребра
      cout << endl << "E:{ ";

      for (int j = 0; j < v - 1; j++) {
            cout << "(" << rebra[j][0] << "," << rebra[j][1] << "),";
      }
      cout << "}";
      return 0;

      }
```

Висновок :

Я набула практичних вмінь та навичок з використання алгоритмів Прима і Краскала.