МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА“

Кафедра систем штучного інтелекту

Лабораторна робота №4
З дисципліни
« Дискретна математика »

Виконала:
Студентка групи КН-115
Галік Вікторія
Викладач:
Мельникова Н.І.

Львів – 2019р.

Тема: Основні операції над графами. Знаходження остова мінімальної ваги за алгоритмом Пріма-Краскала **Мета роботи:** набуття практичних вмінь та навичок з використання алгоритмів Пріма і Краскала.

Варіант 5

Постановка завдання:

Завдання № 1.

Розв'язати на графах наступні задачі:

1. Виконати наступні операції над графами:

1) знайти доповнення до першого графу,

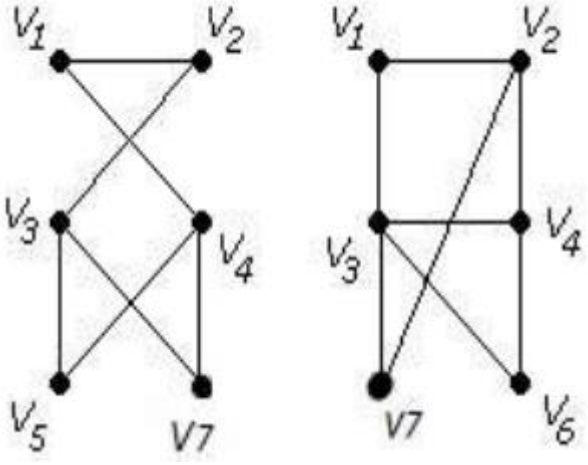
2) об’єднання графів,

3) кільцеву суму G1 та G2 (G1+G2),

4) розщепити вершину у другому графі,

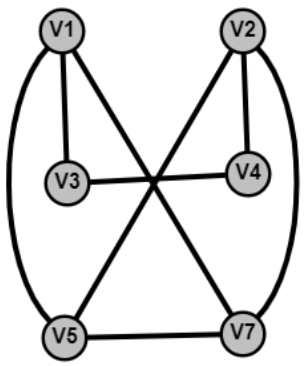
5) виділити підграф А, що складається з 3-х вершин в G1 і знайти стягнення А в G1 (G1\ A),

6) добуток графів.

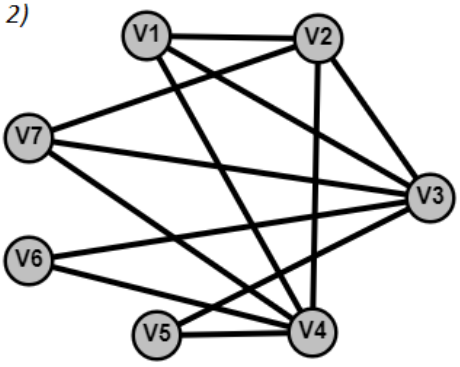


Розв’язання:

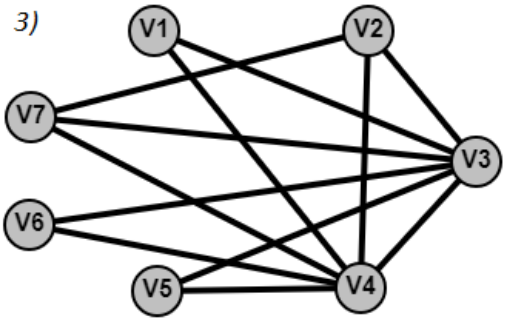
1)



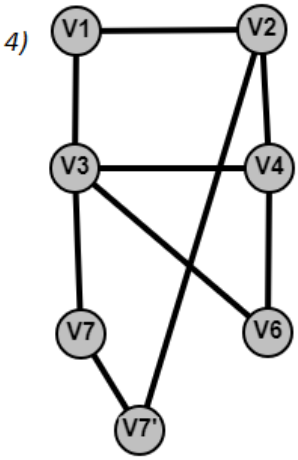
2)



3)

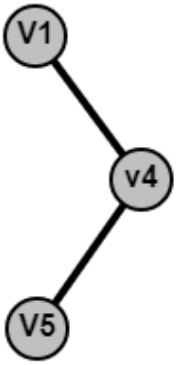


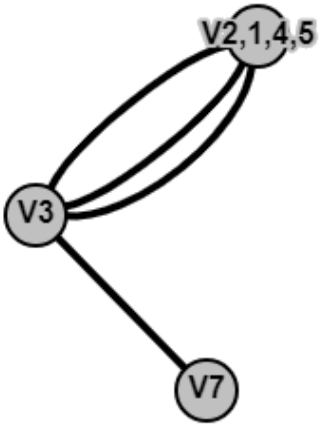
4)



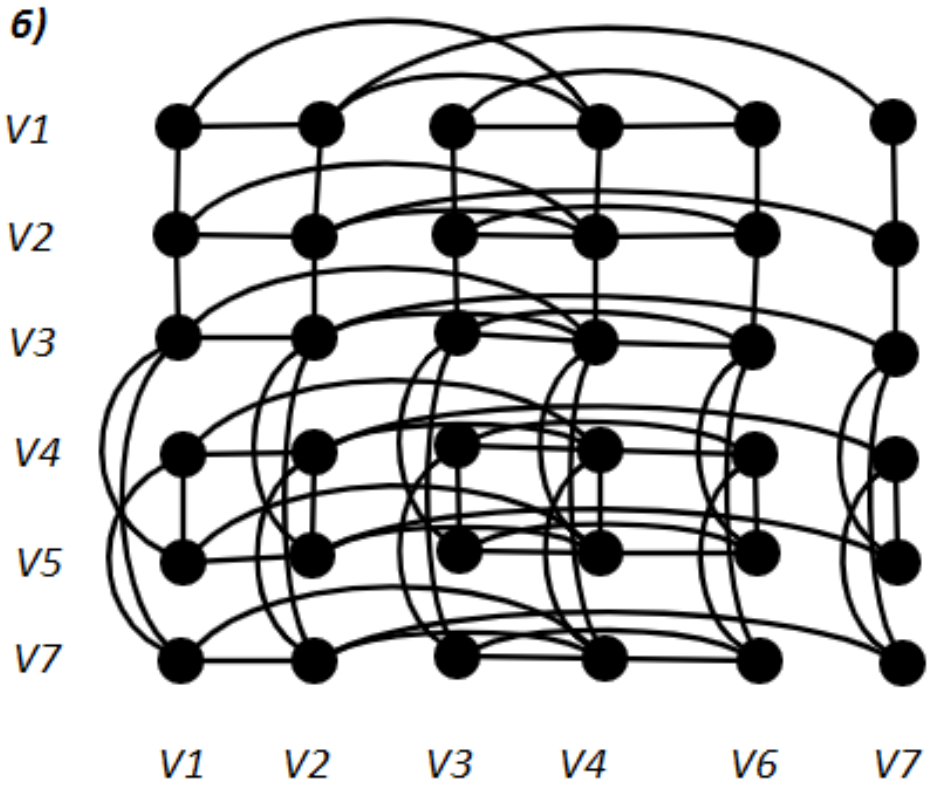
5)

Підграф А



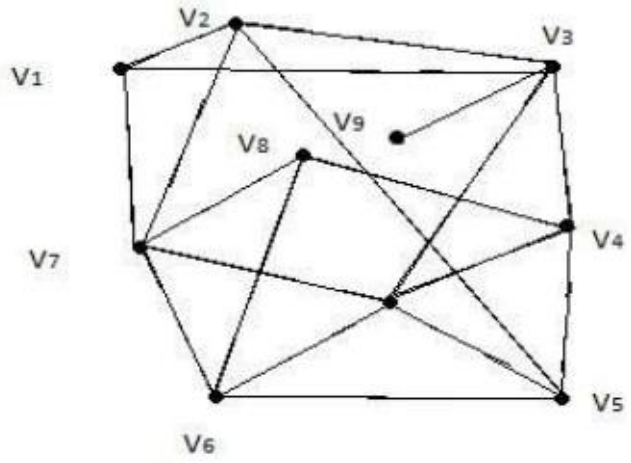


б)



2. Знайти таблицю суміжності та діаметр графа.

5



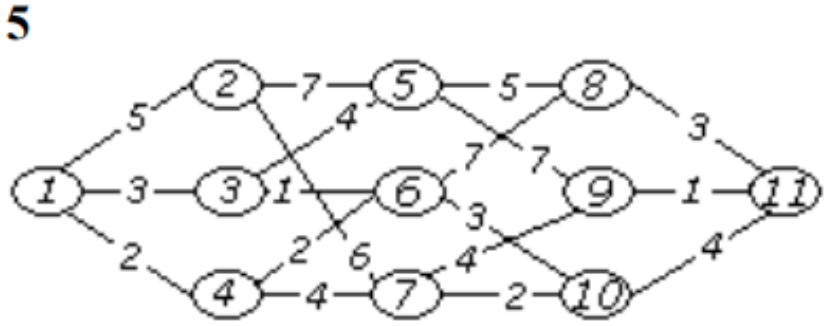
Розв’язання:

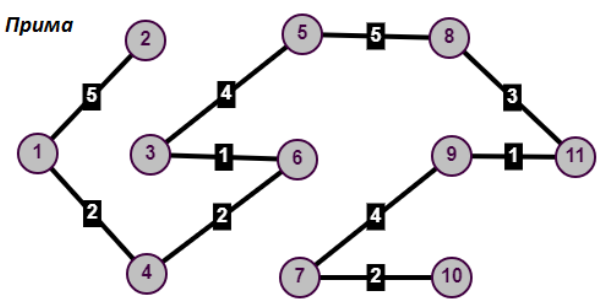
Діаметр = 3 (найдовша відстань між вершинами)

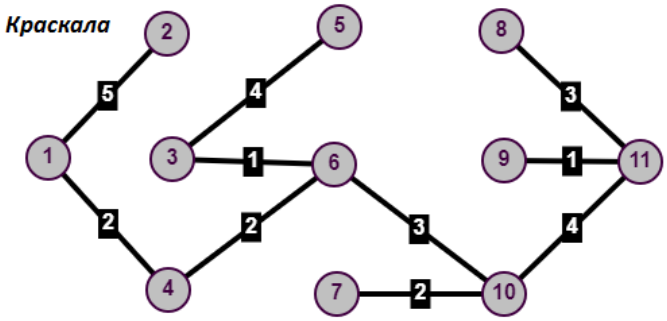
Матриця суміжності:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
V1	0	1	1	0	0	0	1	0	0	0
V2	1	0	1	0	1	0	1	0	0	0
V3	1	1	0	1	0	0	0	0	1	1
V4	0	0	1	0	1	0	0	1	0	1
V5	0	1	0	1	0	1	0	0	0	0
V6	0	0	0	0	1	0	1	1	0	1
V7	1	1	0	0	0	1	0	1	0	1
V8	0	0	0	1	0	1	1	0	0	0
V9	0	0	1	0	0	0	0	0	0	0
V10	0	0	1	1	0	1	1	0	0	0

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







V = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11}

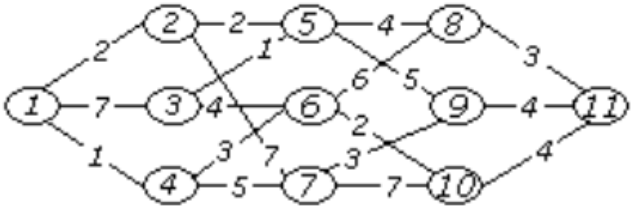
Прима : E = { (1,4), (4,6), (6,3), (3,5), (5,8), (8,11), (11,9), (9,7), (7,10), (1,2) }

Краскала : E = { (3,6), (9,11), (1,4), (4,6), (7,10), (6,10), (8,11), (10,11), (3,5), (1,2) }

Завдання №2. Написати програму, яка реалізує алгоритм знаходження остового дерева мінімальної ваги згідно свого варіанту.

Варіант № 5

За алгоритмом Прима знайти мінімальне остове дерево графа. Етапи розв'язання задачі виводити на екран. Протестувати розроблену програму на наступному графі:

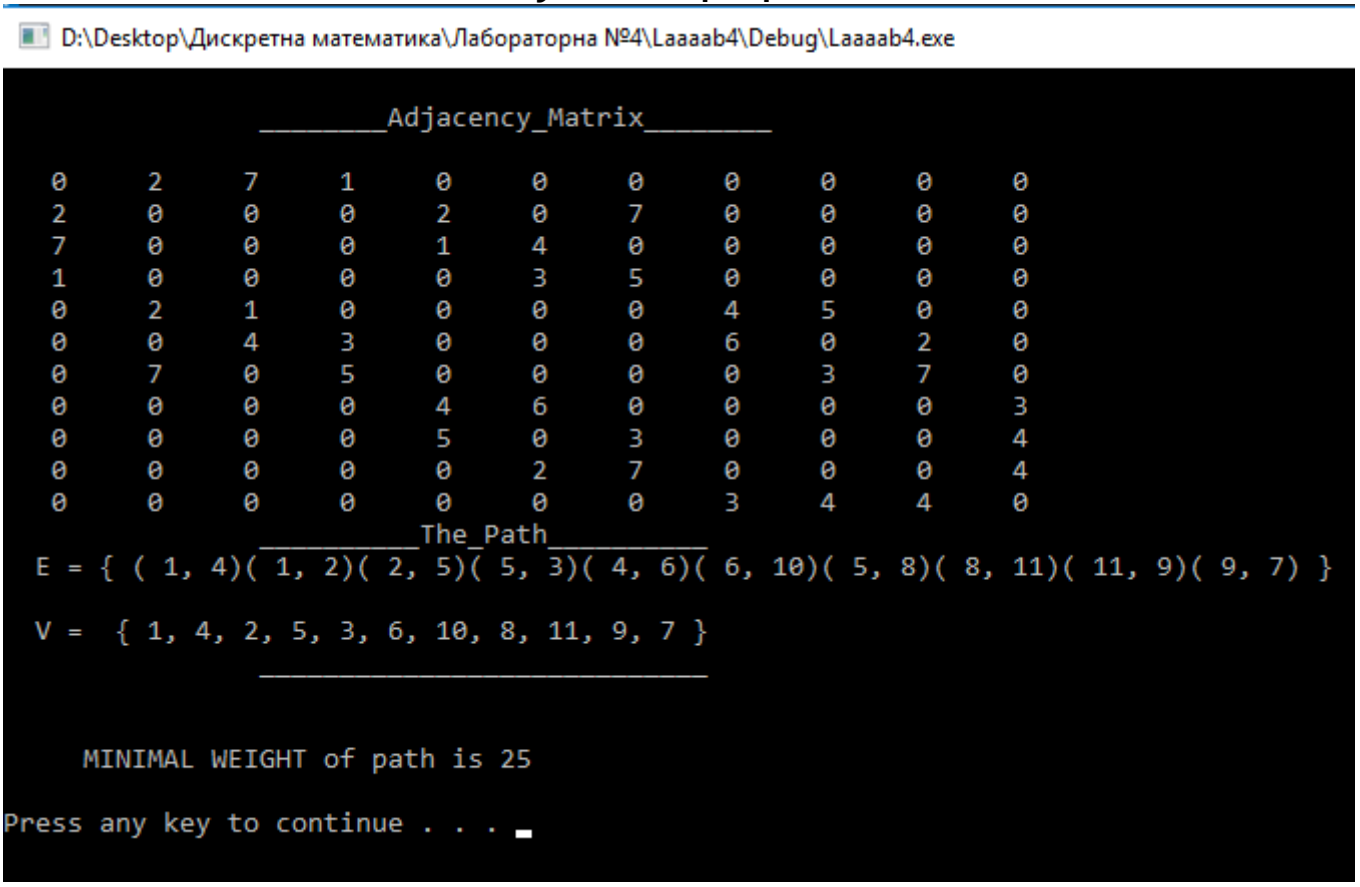


Програмна реалізація :

```
#include<conio.h>
#include<iostream>

using namespace std;

int main(){

	int a, b, u, v, n, i, j, ne = 1;
	int visited[20] = { 0 };          // пройдені вершини
	int min;                          // мінімальна вага
	int minweight = 0;
	int path[100] = { 0 };           // шлях по вершинах

	int path_index = 0;

	cout << endl << "\t\t________Adjacency_Matrix________" << endl << endl;

	n = 11;
	int weight[12][12] = { {0,0,0,0,0,0,0,0,0,0,0,0},      // матриця інцидентності
				{0,0,2,7,1,0,0,0,0,0,0,0},
					{0,2,0,0,0,2,0,7,0,0,0,0},
					{0,7,0,0,0,1,4,0,0,0,0,0},
					{0,1,0,0,0,0,3,5,0,0,0,0},
					{0,0,2,1,0,0,0,0,4,5,0,0},
					{0,0,0,4,3,0,0,0,6,0,2,0},
					{0,0,7,0,5,0,0,0,0,3,7,0},
					{0,0,0,0,0,4,6,0,0,0,0,3},
					{0,0,0,0,0,5,0,3,0,0,0,4},
			{0,0,0,0,0,0,2,7,0,0,0,4},
					{0,0,0,0,0,0,0,0,0,3,4,4,0}
	};

    for (int i = 1; i <=  n; i++)            // вивід матриці
	{
		for ( int j = 1; j <= n ; j++)
		{
			cout << "   " << weight[i][j] << "  ";
		}
		cout << endl;
	}

	for (int i = 1; i <= n; i++)
	{
		for (int j = 1; j <= n; j++)
		{
			if (weight[i][j] == 0) {
				weight[i][j] = 100;        // позначення не інцидентного ребра
			}
```

```
		}
	}

	visited[1] = 1;    //початок шляху
	cout << "\t\t__________The_Path__________" << endl;
	cout << "  E = { ";                                  //вивід множини ребер
	while (ne < n){
	 for (i = 1, min = 100; i <= n; i++) {

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

				if (weight[i][j] < min) {             // перевірка на мінімальну вагу

					if (visited[i] != 0){             // перевірка чи вершина пройдена

						min = weight[i][j];

						a = u = i;
						b = v = j;

					}
				}
			}
		}

		if (visited[u] == 0 || visited[v] == 0){     // перевірка на цикл
			path[path_index] = b;
			path_index++;
			ne++;

			minweight += min;
			visited[b] = 1;                          // позначення вершини як пройденої
			cout << "( " << a << ", " << b << ")";

		}
		weight[a][b] = weight[b][a] = 100;
	}
	cout << " }" << endl;
	cout << endl;

	cout << "  V =  { ";          // вивід множини вершин
	cout << 1 << ", ";
	for (int i = 0; i < n - 1; i++)
	{
		cout << path[i];
		if (i < n - 2) cout << ", ";
	}
	cout << " }" << endl;

	cout << "\t\t_____________________________" << endl << endl;
	cout << endl << "     MINIMAL WEIGHT of path is " << minweight << endl << endl;     // сума
шляху

	system("pause");
	return 0;
}
```

Результат програми :



Висновок : ми набули практичних вмінь та навичок з використання алгоритмів Прима і Краскала.