МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ В НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №4

з дисципліни
«Дискретна математика»

Виконав:
студент групи КН-115
Гончаренко Н.
Викладач:
Мельникова Н.І.

Львів – 2019 р.

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 6

Завдання № 1. Розв'язати на графах наступні задачі:

1. Виконати наступні операції над графами:

1) знайти доповнення до першого графу,

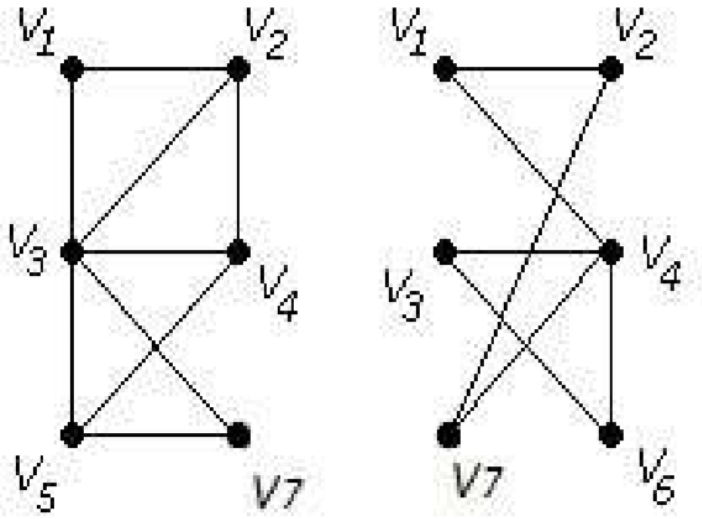
2) об’єднання графів,

3) кільцеву суму G1 та G2 (G1+G2),

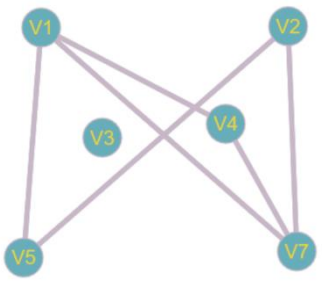
4) розщепити вершину у другому графі,

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

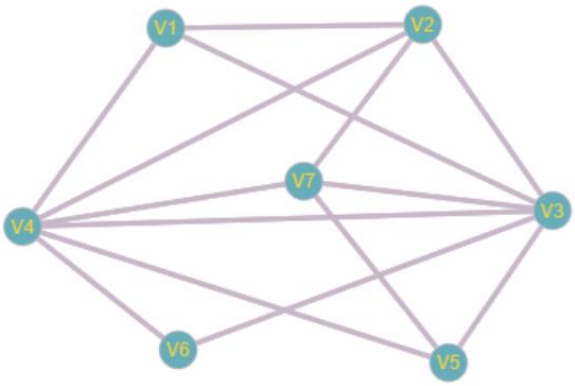
6) добуток графів



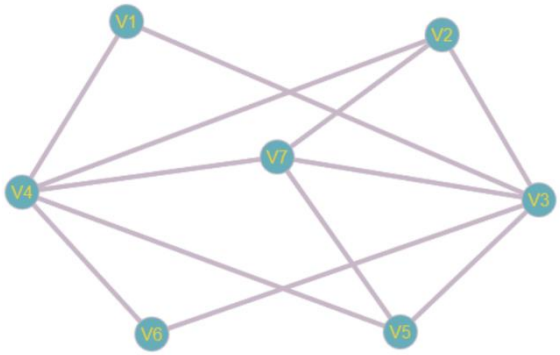
1)



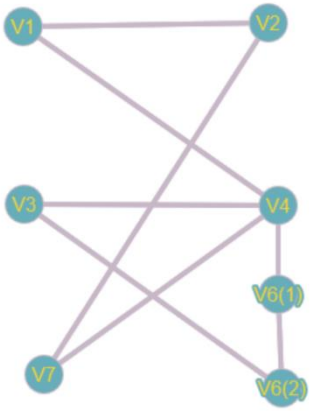
2)



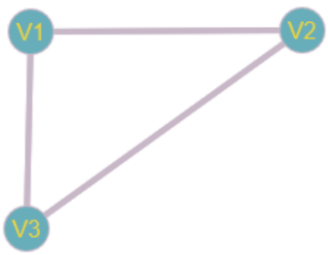
3)



4)

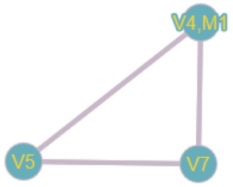


5) Виділимо підграф A , який складається з V1 , V2, V3.

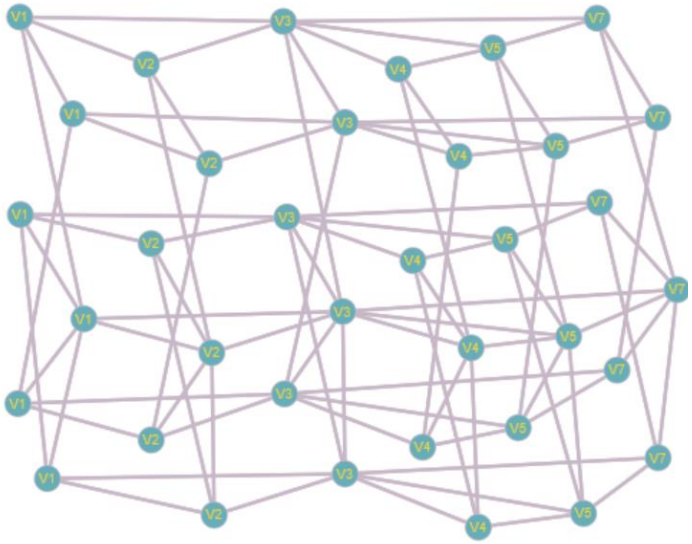


Стягнемо в M1 =>

Стягнемо A в G1 :



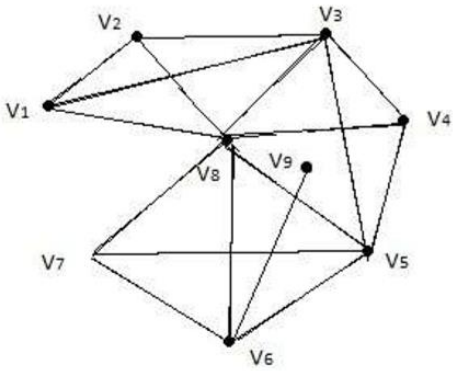
6)



2. Знайти таблицю суміжності та діаметр графа.

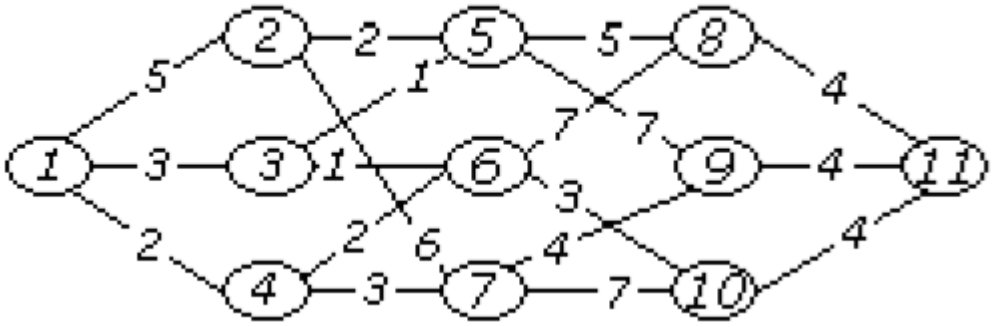
Таблиця суміжності

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



Діаметр = 3.

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

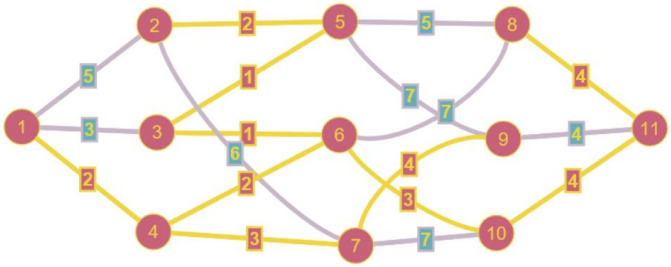


Краскала:

1) Виберемо ребра вага яких дорівнює 1. З них виберемо довільне(3 – 6).

2) Послідовно будемо додавати ребра графа від меншої ваги до більшої перевіряючи на створення циклу.

Перебравши всі ребра з вагою 4 , отримаємо мінімальне остове дерево.



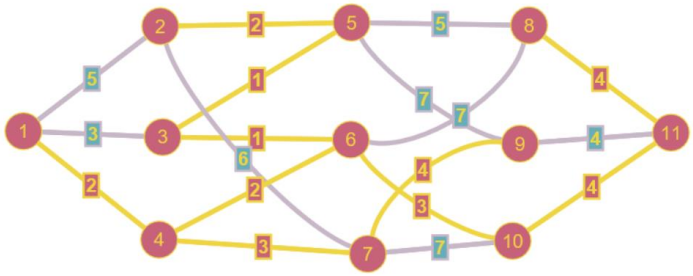
Прима:

1) Виберемо довільну вершину (1).

2) Розглянемо усі інцидентні ребра і виберемо з найменшою вагою (1 – 4).

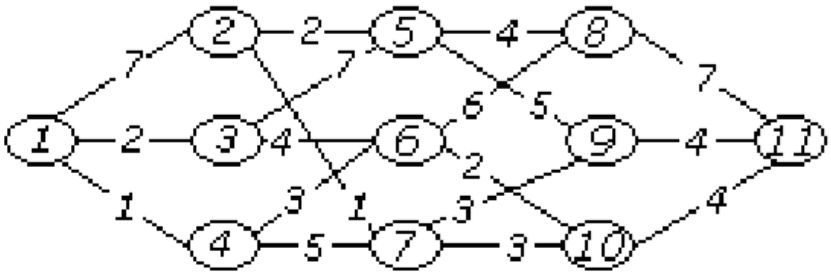
3) Перейдемо до вершини (4) і проробимо з нею крок 2*.

Пройшовши 10 вершин отримаємо остове дерево:



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

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



Програма:

```
#include <iostream>
#include <fstream>

struct Node
{
    int a1, a2, w;
};
struct Graf
{
    int a1, a2;
};

void Enter_Node_from_file(Node* p);
void Enter_Node(Node*);
void Krass(Node*, Graf*);
void Print_Graf(Graf*);
void Sort_Nodes(Node*);

using namespace std;
int V, R;
int main()
{
    cout << "Enter number of Vertices: ";  cin >> V;
    cout << "Enter number of Edges: "; cin >> R;
    Graf* g = new Graf [V - 1];
    Node* p = new Node [R];
    Enter_Node_from_file(p);
    Sort_Nodes(p);
    Krass(p, g);
    Print_Graf(g);
    delete[] g;
    delete[] p;
}


void Enter_Node_from_file(Node* p)
{
    ifstream fp;
    fp.open("graf.txt");
    if (fp.is_open())
    {
        for ( int i = 0 ; i < R ; i++)
        {
```

```
                fp >> p[i].a1 >> p[i].a2 >> p[i].w;
            }
        }
        fp.close();
    }

    void Enter_Node(Node* p)
    {
        for (int i = 0; i < R; i++)
        {
            cin >> p[i].a1 >> p[i].a2 >> p[i].w;
        }
    }

    void Krass(Node* p, Graf* g)
    {
        int* arr_for_v = new int [V];
        for (int i = 0; i < V; i++)
        {
            arr_for_v[i] = i;
        }
        int counter = 0;
        for (int i = 0; i < R ; i++)
        {
            int a1 = p[i].a1, a2 = p[i].a2;
            if (arr_for_v[a1-1] != arr_for_v[a2-1])
            {
                g[counter].a1 = a1; g[counter].a2 = a2;
                counter++;
                int new_ver = arr_for_v[a1-1], old_ver = arr_for_v[a2-1];
                for (int j = 0; j < V ; j++)
                {
                    if (arr_for_v[j] == old_ver)
                    {
                        arr_for_v[j] = new_ver;
                    }
                }
            }

        }
        delete[] arr_for_v;
    }

    void Print_Graf(Graf* g)
    {
        for (int i = 0; i < V - 1; i++)
        {
            cout << g[i].a1 << " " << g[i].a2 << endl;
        }
    }
    void Sort_Nodes(Node* p)
    {
        Node T = { 0 ,0 ,0 };
        for (int i = 0; i < R; i++)
        {
            for (int j = 0; j < R - i; j++)
            {
                if (p[j].w > p[j + 1].w)
                {
                    T = p[j];
                    p[j] = p[j + 1];
                    p[j + 1] = T;
                }
            }
        }
    }
```

Результати виконання програми:

```
Enter number of Vertices: 11
Enter number of Edges: 18
1 4
2 7
1 3
2 5
6 10
4 6
7 9
7 10
5 8
9 11
```