МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 4

з дисципліни
«Дискретна математика»

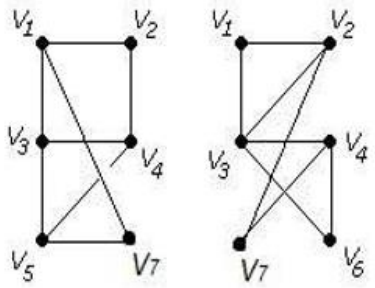
Виконав:
студент групи КН-114
Кмитюк Катерина
Викладач:
Мельникова Н.І.

Львів – 2019 р.

Тема: Основні операції над графами. Знаходження остова мінімальної ваги за алгоритмом Пріма-Краскала.
Мета: Набуття практичних вмінь та навичок з використання алгоритмів Пріма і Краскала.
Завдання з додатку 1:

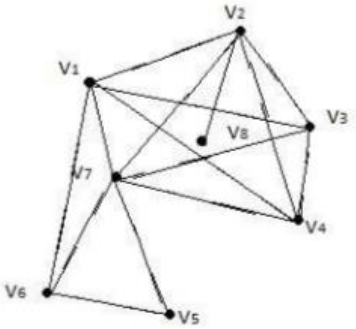
1. Виконати наступні операції над графами: 1) знайти доповнення до першого графу, 2) об'єднання графів, 3) кільцеву суму G1 та G2 (G1+G2), 4) розщепити вершину у другому графі, 5) виділити підграф А, що складається з 3-х вершин в G1 і знайти стягнення А в G1 (G1\ A), 6) добуток графів.

10



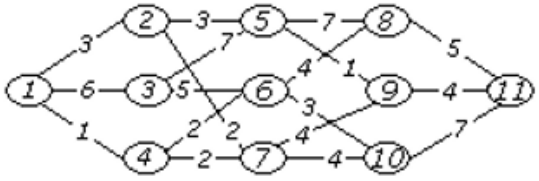
2. Знайти таблицю суміжності та діаметр графа.

10



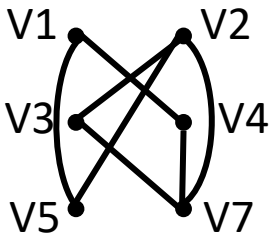
3. Знайти двома методами (Краскала і Прима) мінімальне остове дерево графа.

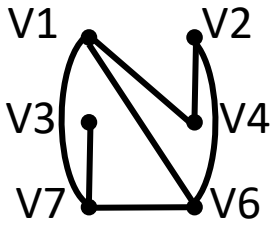
10



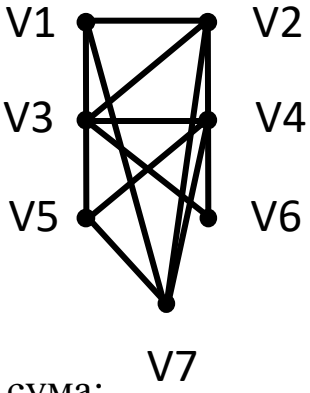
Розв'язання:

1. 1) Доповнення:

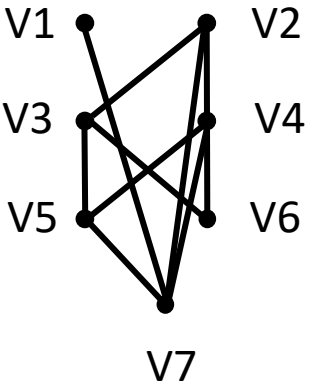
G1: 

G2: 

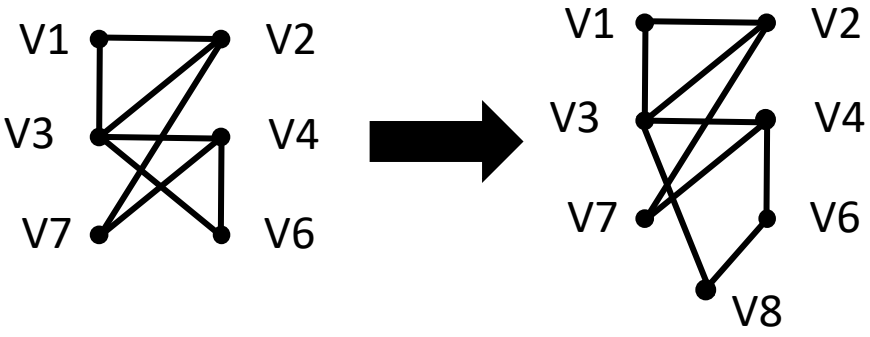
2) Об’єднання:



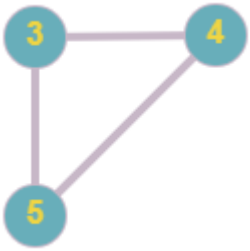
3) Кільцева сума:



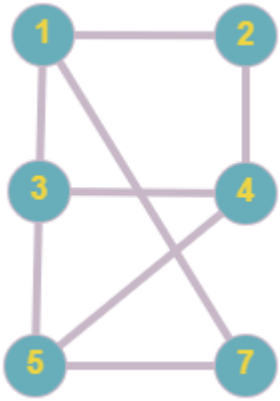
4) Розщеплення вершини V6 у другому графі:

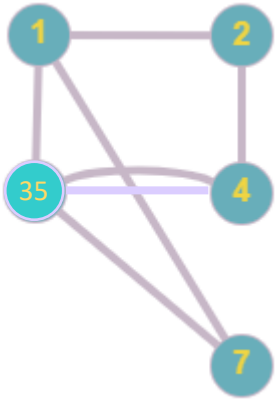


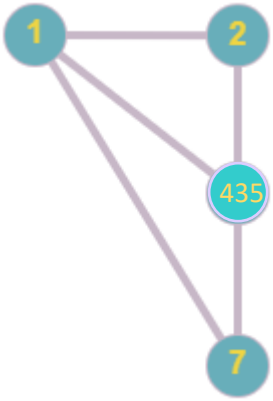
5) Виділення підграфа A з 3-х вершин в G1 і стягнення G1 в A: Стягуємо вершини G1, які належать A. Граф A складається з вершин V1, V2, V3.

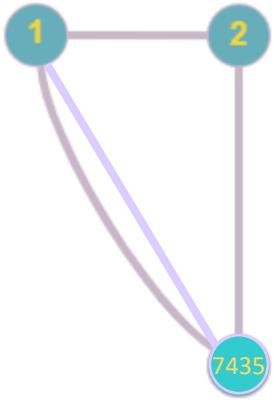


A: Стягування G1:

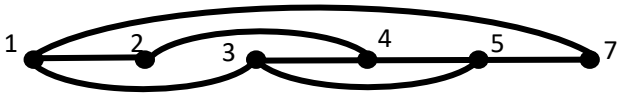


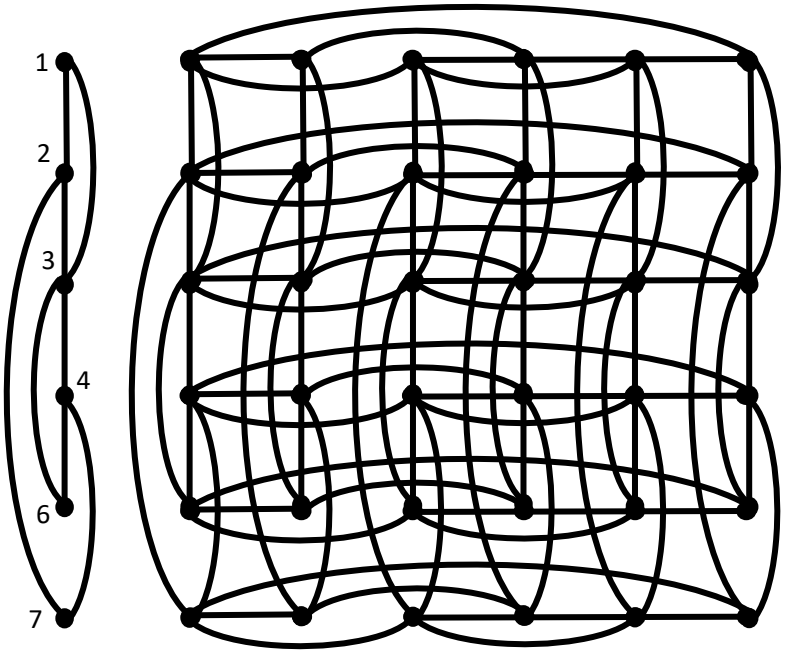






6) Добуток:



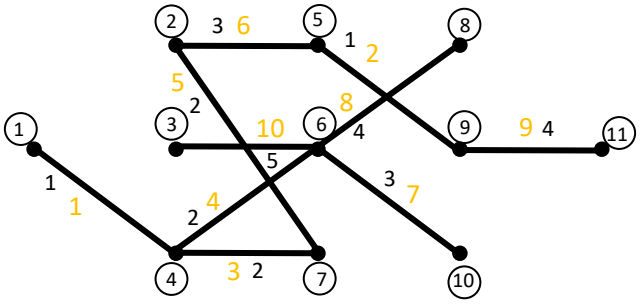


2. Таблиця суміжності:

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

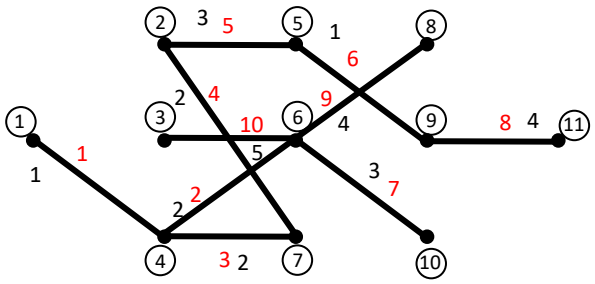
Діаметр графа = 3 (це максимальна відстань між вершинами графа 5 і 8, 6 і 8).

3. Метод Краскала:



Щоб отримати дане остове дерево методом Краскала потрібно знайти найменше ребро та додати його разом із вершинами, які воно сполучає. Якщо є кілька ребер, що важать однаково, обираємо довільне. Потім продовжуємо шукати мінімальні ребра та додавати їх в тому випадку, якщо це ребро не утворить цикл. Перевіривши усі ребра, отримуємо такий результат.

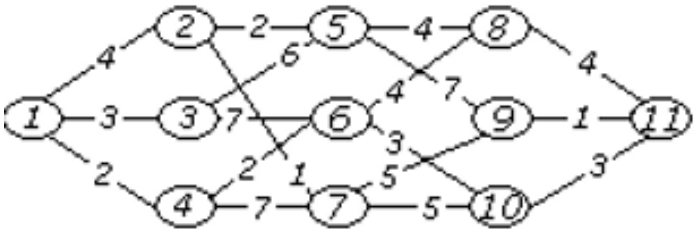
Метод Прима:



Для отримання остового дерева методом Прима обираємо довільну вершину і шукаємо інцидентне їй найменше ребро та

додаємо його, разом із суміжною вершиною. Далі шукаємо наступне найменше ребро серед ребер інцидентних вже доданим вершинам. Продовжуємо таким чином формувати остове дерево і отримуємо результат, ідентичний тому, що був одержаний завдяки методу Краскала. Отже, обидва методи дають однаковий результат.

Завдання з додатку 2:
За алгоритмом Краскала знайти мінімальне остове дерево графа. Етапи розв'язання задачі виводити на екран. Протестувати розроблену програму на наступному графі:



Програмна реалізація:

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

bool trans (int **c, int n, int m)
{
    int i,j,k;
    bool tran=1;
    for (i=0; i<n; i++)
    {
        for (j=0; j<m; j++)
        {
            if (c[i][j]==1)
            {
                for (k=0; k<m; k++)
                    if (c[j][k]==1)
                        if (c[i][k]==1)
                        tran*=1;
                        else tran*=0;
                    else tran*=0;
            }
        }
    }
    return tran;
}


int main() {
    int *v,*e;
    int k;
    int **w, **temp;
    bool cheki=1, chekj=1;

    cout << "Input count of vertices" << endl;
    cin >> k;

```

```
    w=(int **)calloc(k, sizeof(int *));
    temp=(int **)calloc(k,sizeof(int *));
    for (int i=0; i<k; i++)
    {
        w[i]=(int *)calloc(k+1, sizeof(int));
        temp[i]=(int *)calloc(k+1, sizeof(int));
    }

    v = (int *)calloc(k, sizeof(int));
    e = (int *)calloc((k-1), sizeof(int));

    cout << "Input weight of edges" << endl;
    for (int i=0; i<k; i++)
    {
        for (int j=0; j<k; j++)
        {
            if (j>i)
            {
                cout << i+1 << " - " << j+1 << " ";
                cin >> w[i][j];
            }
            else w[i][j]=0;
        }
        cout << endl;
    }
int C=k;
int p,q;
int minim=INT_MAX;
    for (int l=0; l<C; l++)
    {
        for (int i=0; i<k; i++)
            for (int j=0; j<k; j++)
            if (w[i][j]<minim && w[i][j]!=0)
            {
                p=i; q=j;
                 minim=w[i][j];
             }
        temp[p][q]=1;
        temp[q][p]=1;
        for (int t=0; t<C; t++)
        {
            if (v[t]!=(p+1)) cheki*=1;
            else cheki*=0;
            if (v[t]!=(q+1)) chekj*=1;
            else chekj*=0;
        }
        if ((cheki+chekj)==2)
        {
            e[l]=w[p][q];
            v[l]=p+1;
            l++;
            C++;
            v[l]=q+1;
        }
        else{
        if (cheki==1)
        {v[l]=p+1;    e[l]=w[p][q]; }
        else{
        if (chekj==1)
        {v[l]=q+1;    e[l]=w[p][q]; }
        else{
        if (cheki+chekj==0 && trans(temp,k,k)==0)
        {e[l]=w[p][q];}}}}
        minim=INT_MAX;
        w[p][q]=0;
        cheki=1;
        chekj=1;
    }

    int news=0;
```

```
106         for (int i=0; i<(C-1); i++)
107             if (e[i]!=0 && news<(k-1))
108             {
109                 e[news]=e[i];
110                 news++;
111             }
112         cout << "E: { ";
113         for (int i=0; i<news; i++)
114             cout << e[i] << " ";
115         cout << "}\n";
116
117         news=0;
118         for (int i=0; i<C; i++)
119             if (v[i]!=0)
120             {
121                 v[news]=v[i];
122                 news++;
123             }
124         cout << "V: { ";
125         for (int i=0; i<(news-1); i++)
126             cout << v[i] << " ";
127         cout << "}\n";
128
129         return 0;
130     }
131
```

Виконання програми:

```
Input count of vertices
11
Input weight of edges
1 - 2 4
1 - 3 3
1 - 4 2
1 - 5 0
1 - 6 0
1 - 7 0
1 - 8 0
1 - 9 0
1 - 10 0
1 - 11 0

2 - 3 0
2 - 4 0
2 - 5 2
2 - 6 0
2 - 7 1
2 - 8 0
2 - 9 0
2 - 10 0
2 - 11 0

3 - 4 0
3 - 5 6
3 - 6 7
3 - 7 0
3 - 8 0
3 - 9 0
3 - 10 0
3 - 11 0

4 - 5 0
4 - 6 2
4 - 7 7
4 - 8 0
4 - 9 0
4 - 10 0
4 - 11 0

5 - 6 0
5 - 7 0
5 - 8 4
5 - 9 7
5 - 10 0
5 - 11 0

6 - 7 0
6 - 8 4
6 - 9 0
6 - 10 3
6 - 11 0

7 - 8 0
7 - 9 5
7 - 10 5
7 - 11 0

8 - 9 0
8 - 10 0
8 - 11 4

9 - 10 0
9 - 11 1

10 - 11 3


E: { 1 1 2 2 2 3 3 3 4 4 }
V: { 2 7 9 11 1 4 5 6 3 10 8 }
```

Висновок: у ході лабораторної роботи я навчилась виконувати операції над графами, будувати таблицю суміжності, визначати діаметр графа, набула практичних вмінь та навичок з використання алгоритмів Прима і Краскала.