

**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**

Чорний Святослав

**Викладач:**

Мельникова Н.І.

**Львів – 2019р.**

## Варіант 14

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.

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

#### 1. Виконати наступні операції над графами:

1) знайти доповнення до першого графу,

2) об'єднання графів,

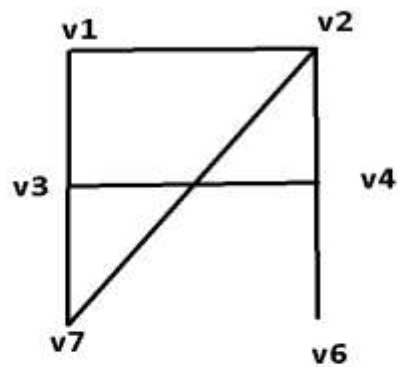
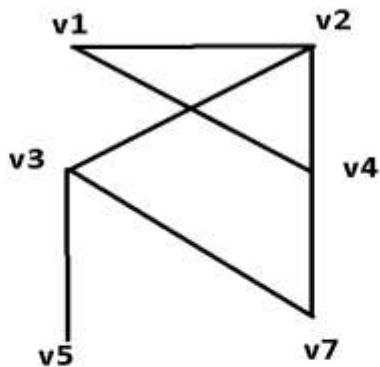
3) кільцеву суму  $G_1$  та  $G_2$  ( $G_1+G_2$ ),

4) розщепити вершину у другому графі,

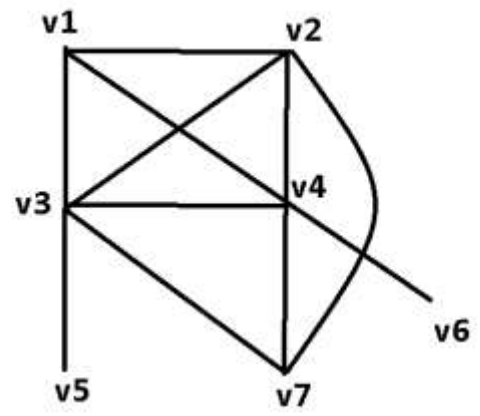
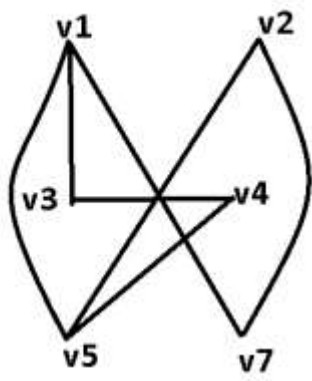
5) виділити підграф  $A$ , що складається з 3-х вершин в  $G_1$  і знайти стягнення  $A$  в  $G_1$  ( $G_1 \setminus A$ ),

6) добуток графів.

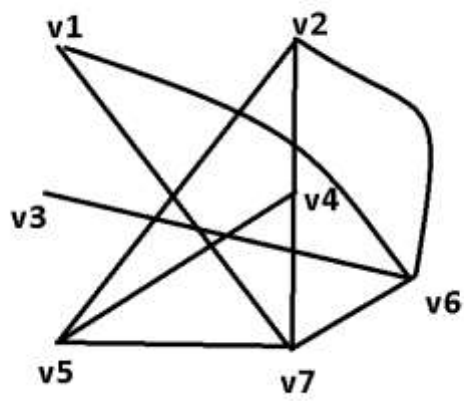
Дано:



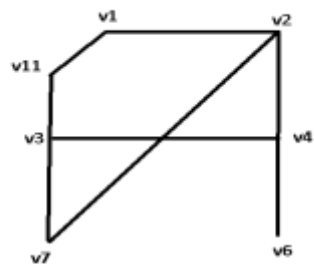
1.1;1.2



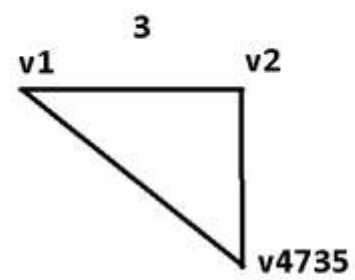
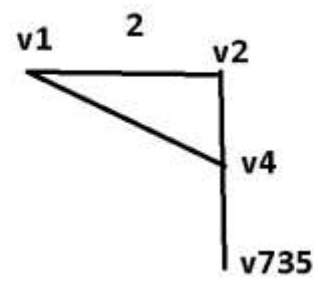
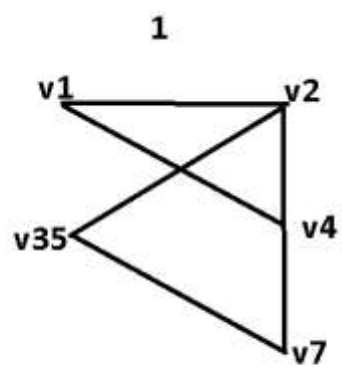
1



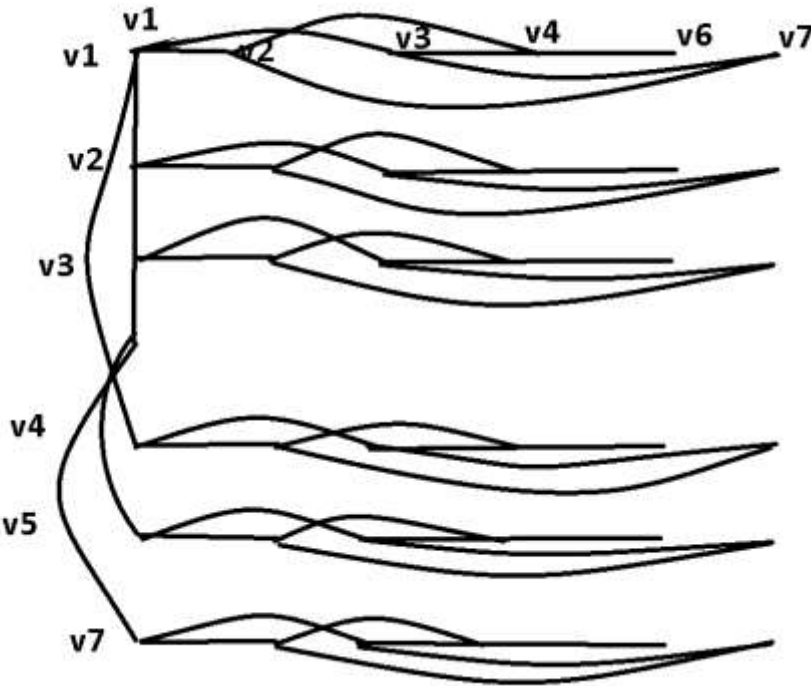
1.4



1.5



1.6

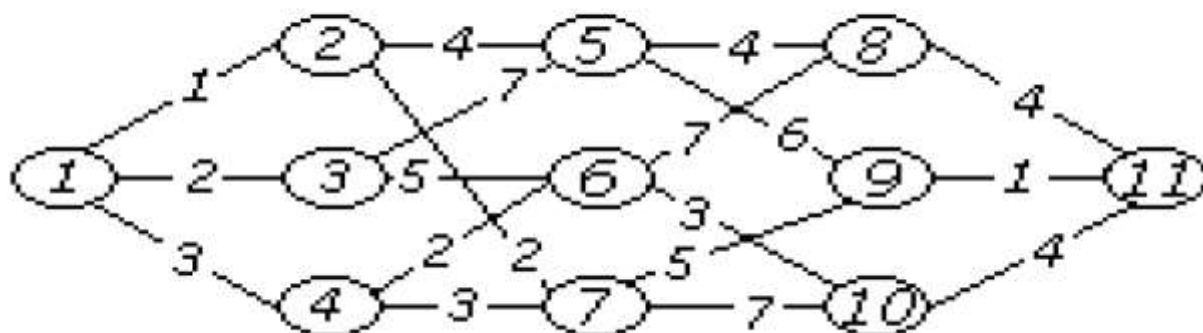


2.Знайти таблицю суміжності та діаметр графа.

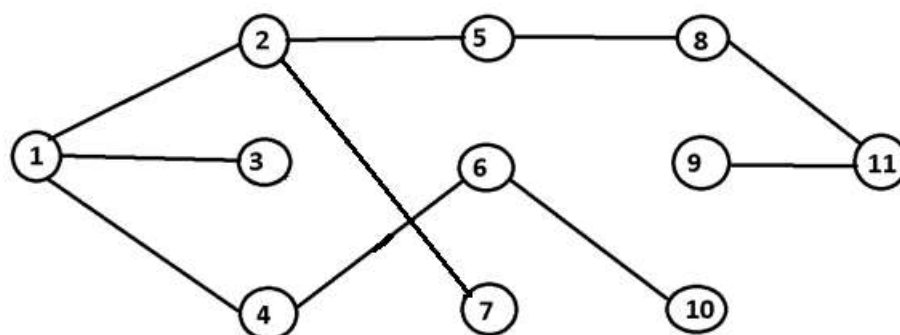
	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
V1	0	1	1	0	0	0	1	1	1	0
V2	1	0	1	0	0	0	0	0	1	1
V3	1	1	0	1	1	0	0	0	0	0
V4	0	0	1	0	1	0	0	0	1	0
V5	0	0	1	1	0	1	1	0	1	0
V6	0	0	0	0	1	0	0	0	1	0
V7	1	0	0	0	1	0	0	0	0	0
V8	1	0	0	0	0	0	0	0	1	0
V9	1	1	0	1	1	1	0	0	0	0
V10	0	1	0	0	0	0	0	0	0	0

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

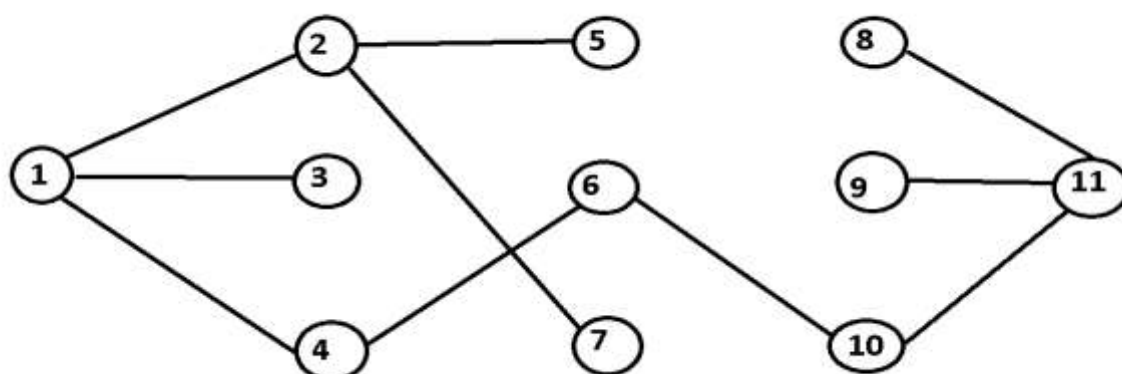
14



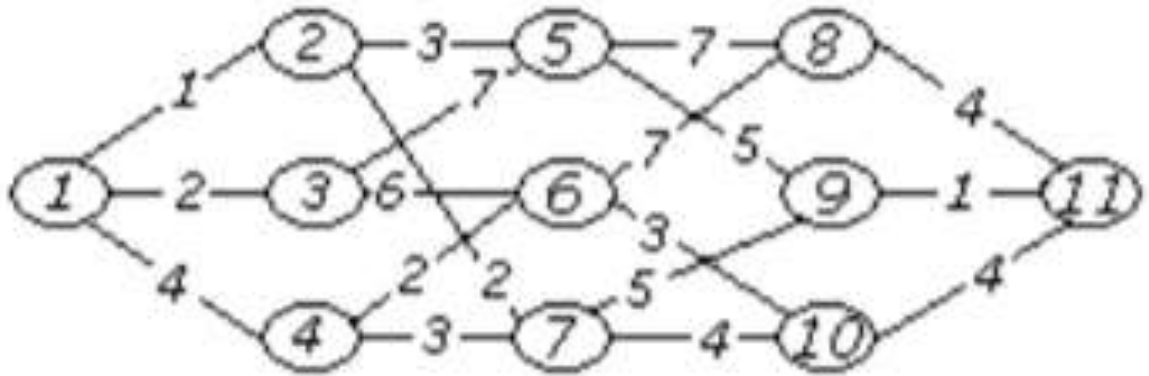
Краскала



Прима



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



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

int main()
{
    int n, m, weight, x, y;
    cout << "kil versun:"; cin >> n;
    cout << "kil reber:"; cin >> m;
    vector < pair < int, pair<int, int> > > g; // вес - вершина 1 - вершина 2

    //вводимо ребра(вершини позначаєм всьо як є
    for (int i = 0; i < m; i++) {
        cout << "Rebro[" << i << "] = " << endl;
        cout << "ver1:"; cin >> x;
        cout << "ver2:"; cin >> y;
        cout << "weight:"; cin >> weight;
        g.push_back({ weight, {--x,--y} });
    }

    int cost = 0;
    vector < pair<int, int> > res;
    //відсортували ребра
    sort(g.begin(), g.end());
    vector<int> tree_id(n);

    //алгоритм крускала
    for (int i = 0; i < n; ++i)
        tree_id[i] = i;
    for (int i = 0; i < m; ++i)
    {
        int a = g[i].second.first, b = g[i].second.second, l = g[i].first;
        if (tree_id[a] != tree_id[b])
        {
            cost += l;
            res.push_back(make_pair(a, b));
            int old_id = tree_id[b], new_id = tree_id[a];
            for (int j = 0; j < n; ++j)
                if (tree_id[j] == old_id)
                    tree_id[j] = new_id;
        }
    }

    cout << "Minimalna vaaga ostovogo drevi: " << cost << endl;
    for (int i = 0; i < res.size(); ++i)
        cout << "Rebro " << i << ": (" << res[i].first << ", " << res[i].second << ") weight: " << g[i].first << endl;
}
```

```

        tree_id[j] = new_id;
    }
}

//вивід всіх елементів з рез
for (auto index : res) {
    cout << index.first + 1 << " - " << index.second + 1 << endl;;
}
return 0;
}

```

```

kil versun:11
kil neber:18
Rebro[0] =
ver1:1
ver2:2
weight:1
Rebro[1] =
ver1:1
ver2:3
weight:2
Rebro[2] =
ver1:1
ver2:4
weight:4
Rebro[3] =
ver1:2
ver2:5
weight:3
Rebro[4] =
ver1:3
ver2:6
weight:6
Rebro[5] =
ver1:4
ver2:7
weight:3
Rebro[6] =
ver1:3
ver2:5
weight:7
Rebro[7] =
ver1:4
ver2:6
weight:2
Rebro[8] =
ver1:2
ver2:7
weight:2
Rebro[9] =
ver1:5
ver2:8
weight:7
Rebro[10] =
ver1:6
ver2:8
weight:7
Rebro[11] =
ver1:5
ver2:9
weight:5

```

```

ver2:7
weight:2
Rebro[9] =
ver1:5
ver2:8
weight:7
Rebro[10]
ver1:6
ver2:8
weight:7
Rebro[11]
ver1:5
ver2:9
weight:5
Rebro[12]
ver1:6
ver2:10
weight:3
Rebro[13]
ver1:7
ver2:9
weight:5
Rebro[14]
ver1:7
ver2:10
weight:4
Rebro[15]
ver1:8
ver2:11
weight:4
Rebro[16]
ver1:9
ver2:11
weight:1
Rebro[17]
ver1:10
ver2:11
weight:4
1 - 2
9 - 11
1 - 3
2 - 7
4 - 6
2 - 5
4 - 7
6 - 10
8 - 11
10 - 11

```



## Висновок

На даній лабораторній роботі ми набули практичних вмінь та навичок з використання алгоритмів Пріма і Краскала.