МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Лабораторна робота №4

З дисципліни

«Дискретна математика»

Виконав :

Студент КН-113 Байдич Володимир

Викладач:

Мельникова. Н. І

Львів – 2019

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.

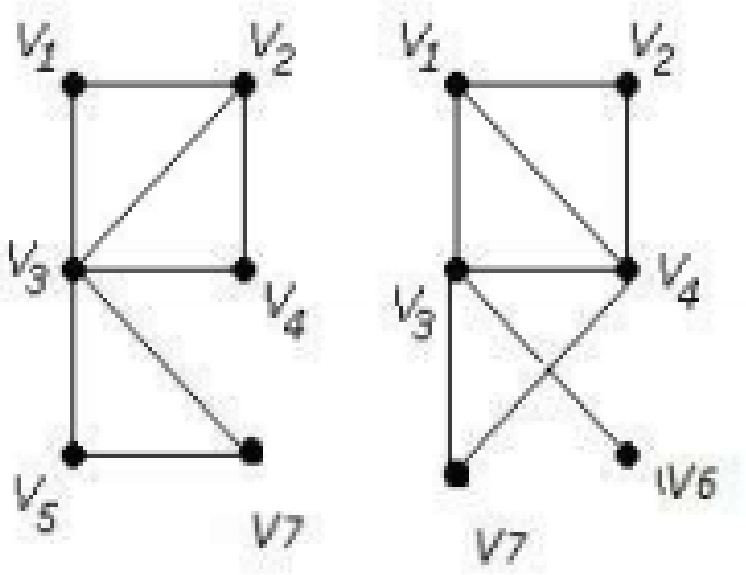
Варіант 2

Завдання 1

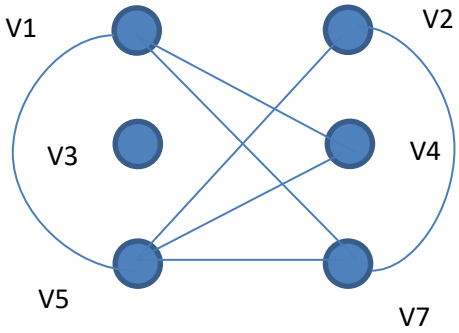
Розв'язати на графах наступні задачі:

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 A, що складається з 3-х вершин в G1 і знайти стягнення A в G1 (G1\ A)
   6) добуток графів.

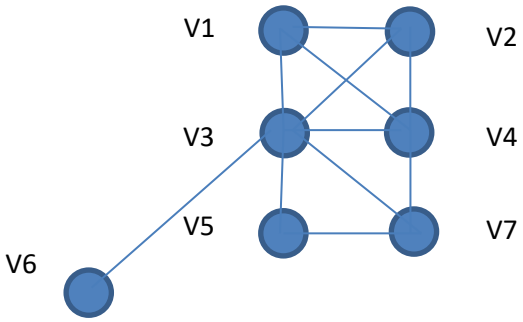
2



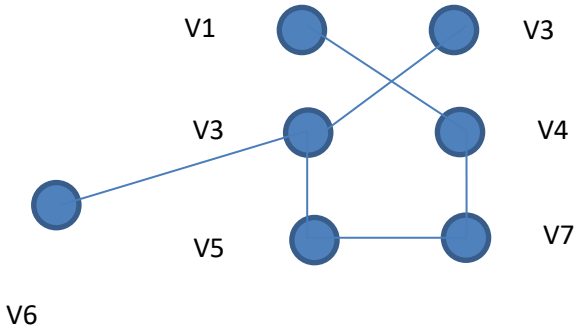
1. доповнення до графа G1



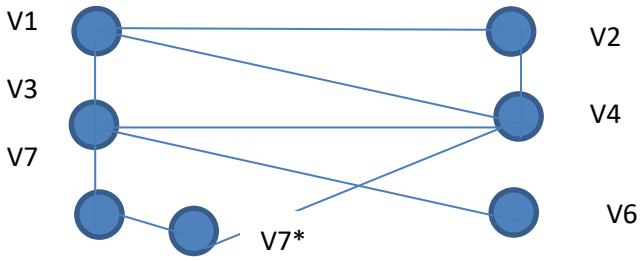
2. Обєднання графів G1 G2



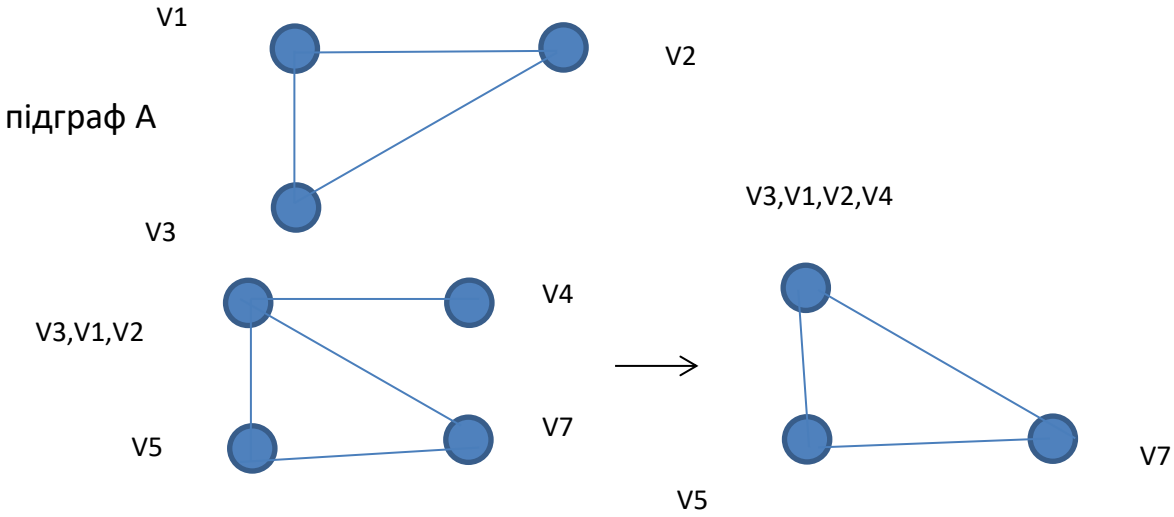
3. Кільцеву суму G1 та G2 (G1+G2)



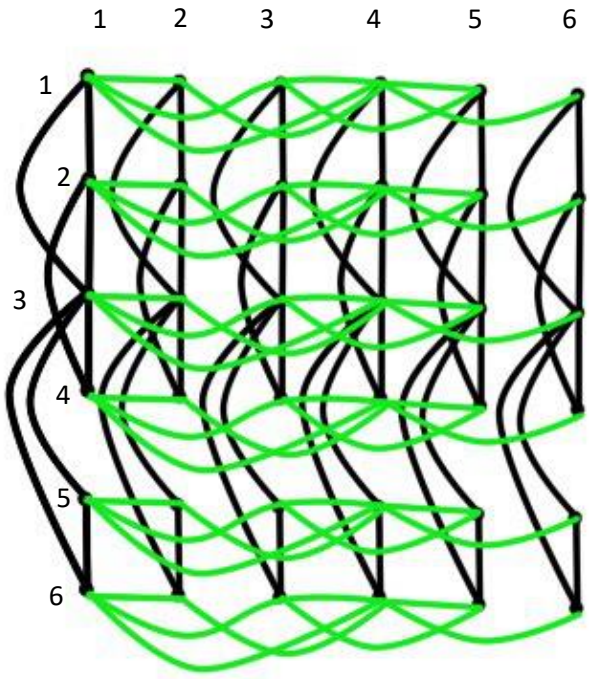
4. розщепити вершину у другому графі.



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



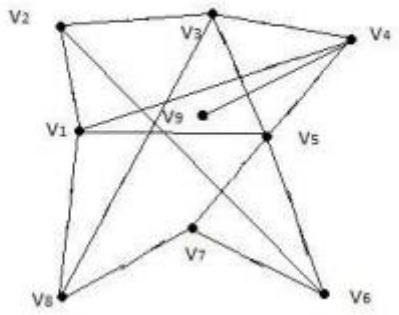
6. добуток графів



Завдання №2

Знайти матриці суміжності та діаметр графа

2



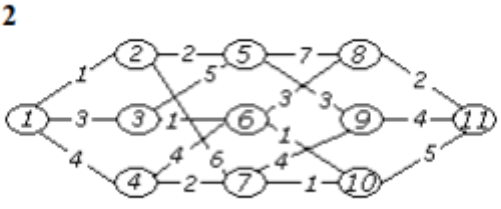
	V1	V2	V3	V4	V5	V6	V7	V8	V9
V1	0	1	2	1	1	2	2	1	2
V2	1	0	1	2	2	1	2	2	3
V3	2	1	0	1	1	2	2	1	2
V4	1	2	1	0	1	2	2	2	1
V5	1	2	1	1	0	1	1	2	2
V6	2	1	2	2	1	0	1	2	3
V7	2	2	2	2	1	1	0	1	3
V8	1	2	1	2	2	2	1	0	3
V9	2	3	2	1	2	3	3	3	0

Найбільши ексцентриситет або діаметр графа дорівнює 3.

Завдання №3

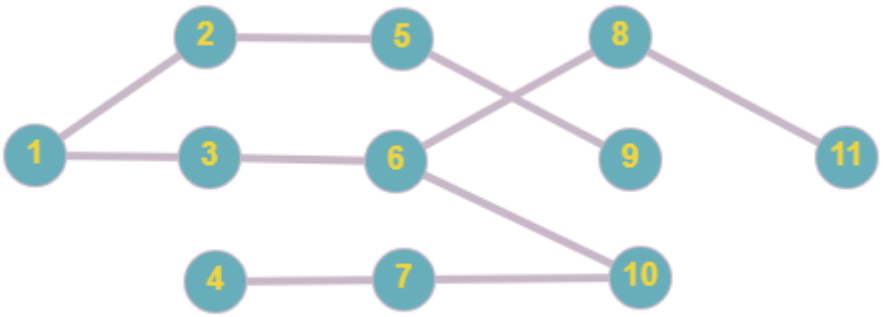
Знайти двома методами (Краскала і Прима) мінімальне остове дерево графа.

2



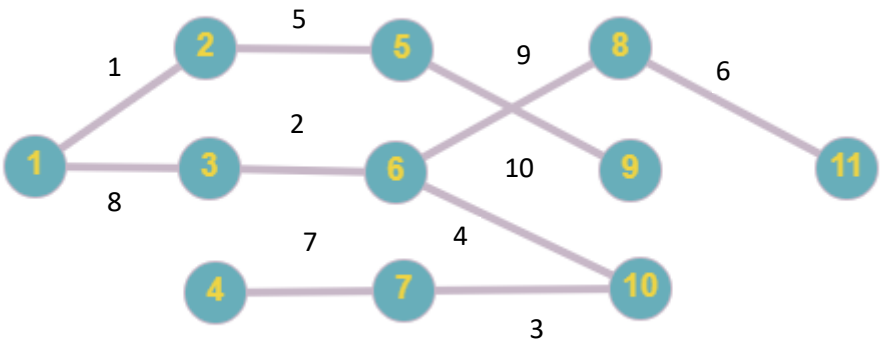
1. за прима:вибираємо найменше ребро :
   1-2 1
   2-5 2
   5-9 3
   1-3 3
   3-6 1
   6-10 1

10-7	1
7-4	2
6-8	3
8-11	2



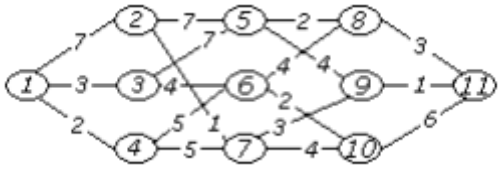
2. за краскала:

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



Частина 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:




```
#include <iostream>
#include <stdio.h>
using namespace std;
struct edge {

 int leng;
 int p1;
 int p2;
 bool in = false;
};
struct mas {

 int arr[11] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
 int c = 0;
};

int in(edge *reb, int n) {
 setlocale(LC_ALL, "Ukrainian");
 for (int i = 0; i < n; i++)
 {
  cout << "Введіть довжину " << i + 1 << " ребра: ";
  cin >> reb[i].leng;
  cout << "Введіть першу суміжну вершину з " << i + 1 << " ребром: ";
  cin >> reb[i].p1;
  cout << "Введіть другу суміжну вершину з " << i + 1 << " ребром: ";
  cin >> reb[i].p2;
  cout << endl;
 }
 return 0;
}
```

```
int main() {
 setlocale(LC_ALL, "Ukrainian");
 int n = 0, x = 100, y = 100;

 cout << "Введіть кількість ребер у графі: ";
 cin >> n;
 int z;
 cout << "Введіть кількість вершин у графі: ";
 cin >> z;
 cout << endl;

 edge *reb = new edge[n];
 mas inn[5];

 in(reb, n);

 for (int i = 0; i < n - 1; i++)
 {
  for (int j = 0; j < n - 1; j++)
  {
   if (reb[j].leng > reb[j + 1].leng) { swap(reb[j].leng, reb[j + 1].leng); swap(reb[j].p1, reb[j + 1].p1);
  }
 }
  int c = -1;
  for (int i = 0; i < n; i++)
 {
   for (int j = 0; j < 5; j++)
  {
    for (int k = 0; k < z; k++)
   {
     if (reb[i].p1 == inn[j].arr[k]) { x = j; goto point0;; }
    }
   }
  point0:;

   for (int j = 0; j < 5; j++)
  {
    for (int k = 0; k <z; k++)
   {
     if (reb[i].p2 == inn[j].arr[k]) { y = j; goto point1; }
    }
   }
  point1:;
   if (x != y && x == 100) { inn[y].arr[inn[y].c] = reb[i].p1; inn[y].c++; }

   if (x != y && y == 100) { inn[x].arr[inn[x].c] = reb[i].p2; inn[x].c++; }

  if (x != y && x != 100 && y != 100) {
   if (x < y) {
    for (int l = 0; l < inn[y].c; l++)
    {
     inn[x].arr[inn[x].c+l] = inn[y].arr[l];
     inn[y].arr[l] = 0;
    }
```

```
    inn[y].arr[inn[y].c+1] = inn[x].arr[1];
    inn[x].arr[1] = 0;
   }
   inn[y].c += inn[x].c;
  }
 }
 if (x == 100 && y == 100) { c++;  inn[c].arr[inn[c].c] = reb[i].p1; inn[c].arr[inn[c].c + 1] = reb[i].

 reb[i].in = true;

 if (x == y && x != 100) { reb[i].in = false; }

 x = 100; y = 100;
}

cout << "ребра остового дерева: " << endl;

int v = 0, s = 0;;

for (int i = 0; i < n; i++)
{
 if (reb[i].in == true) { cout << "Ребро" << ", що сполучає вершини " << reb[i].p1 << " " << reb[i].p2
 if (v == z-1) { break; }
}
cout << "Остове дерево мінімальної ваги для даного графа: " << s;
return 0;}
```

```
ребра остового дерева:
Ребро, що сполучає вершини 9 11
Ребро, що сполучає вершини 2 7
Ребро, що сполучає вершини 5 8
Ребро, що сполучає вершини 4 1
Ребро, що сполучає вершини 6 10
Ребро, що сполучає вершини 8 11
Ребро, що сполучає вершини 1 3
Ребро, що сполучає вершини 7 9
Ребро, що сполучає вершини 10 7
Ребро, що сполучає вершини 3 6
Остове дерево мінімальної ваги для даного графа: 25
Process returned 0 (0x0)   execution time : 162.677 s
Press any key to continue.
```