МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 4

з дисципліни

«Дискретна математика»

Виконав:

Дребот Владислав Олегович

студент групи КН-112

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 4

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

1. Виконати наступні операції над графами:

1) знайти доповнення до першого графу,

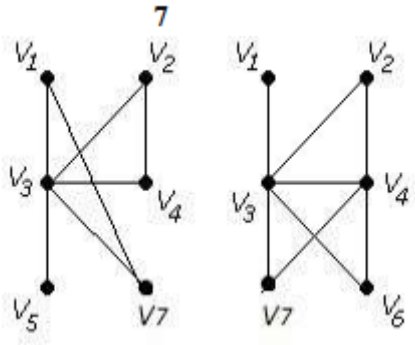
2) об’єднання графів,

3) кільцеву суму G1 та G2 (G1+G2),

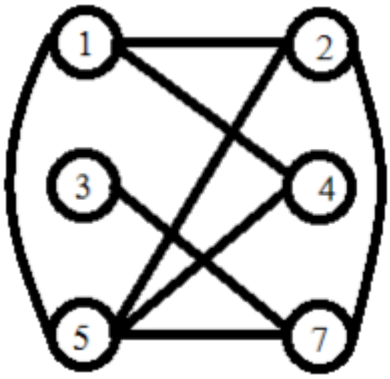
4) розщепити вершину у другому графі,

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

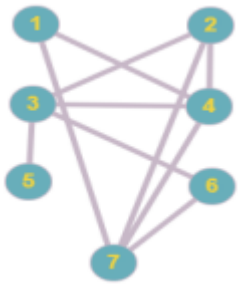
стягнення А в G1 (G1\ А), 6) добуток графів.



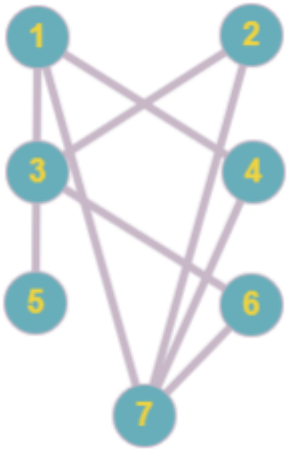
1) Доповнення до першого графу:



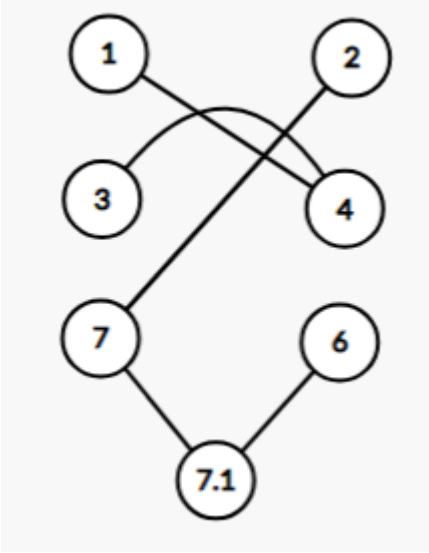
2) Об’єднання:



3) Кільцева сума :

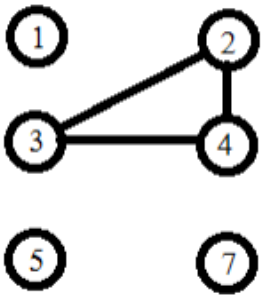


4) Розщеплення 7 вершини у 2 графі:

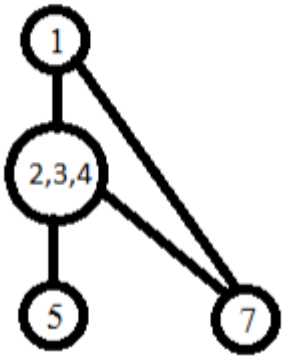


5) Виділити підграф А, що складається з трьох вершин в G1:

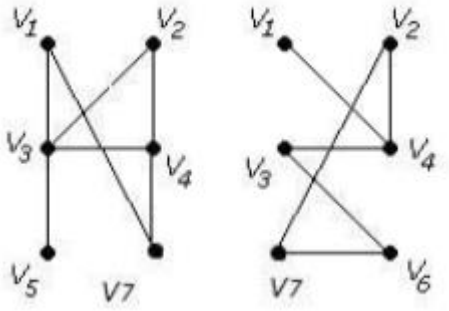
Підграф А

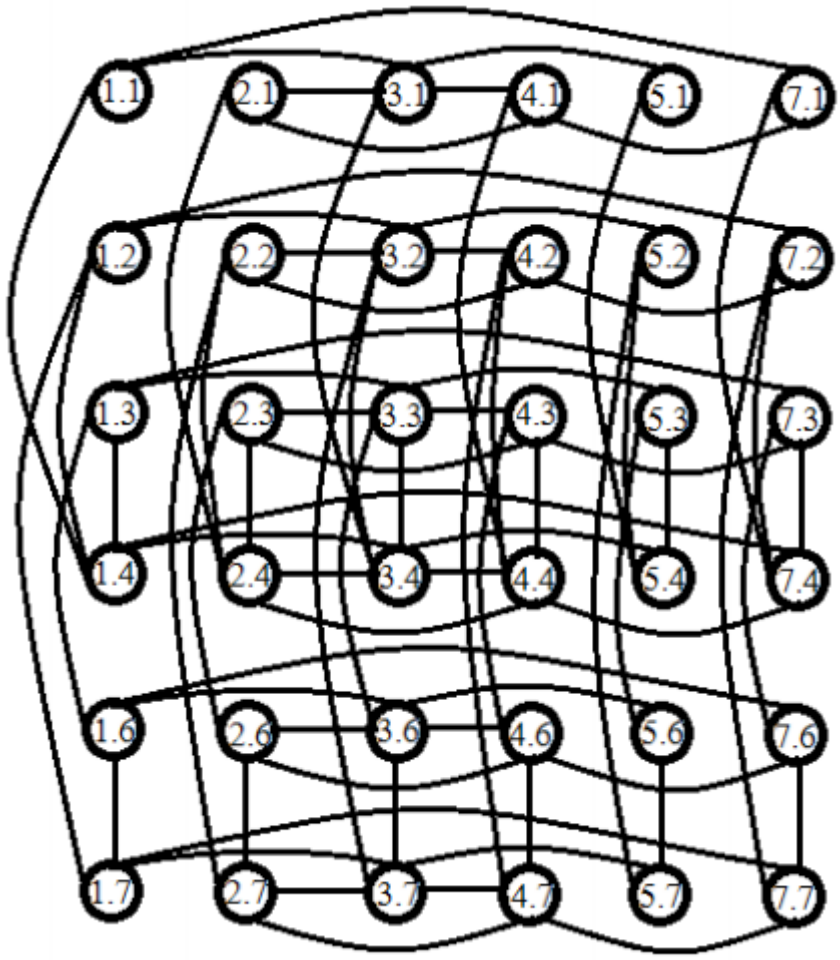


Стяганеня A в G1



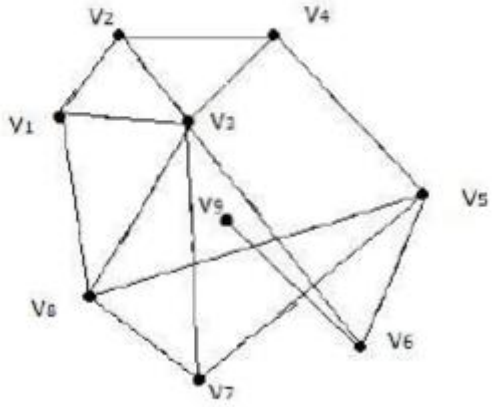
6) Добуток





2. Знайти таблицю суміжності та діаметр графа.

4

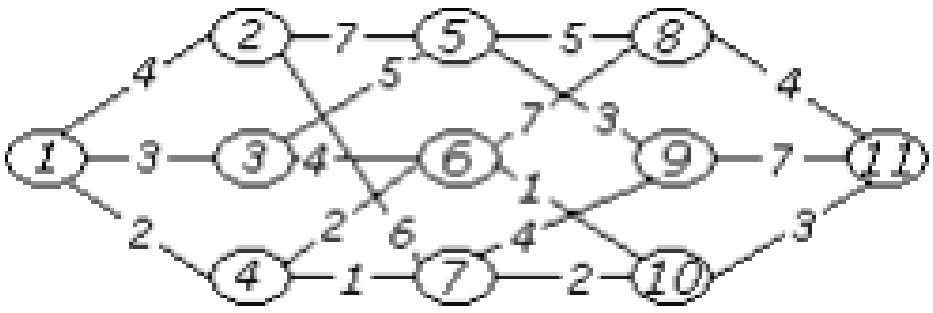


Діаметр=3(V1-V3-V6-V9)

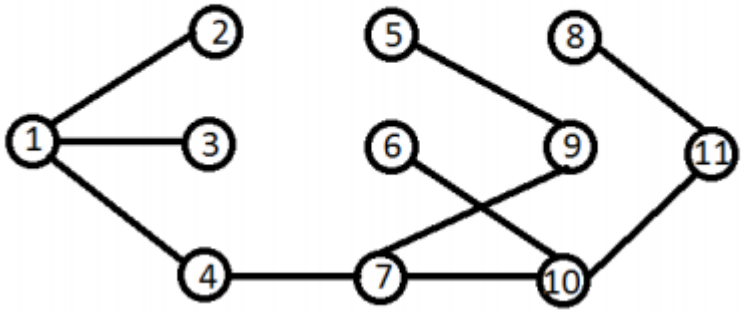
	V1	V2	V3	V4	V5	V6	V7	V8	V9
V1	0	1	1	0	0	0	0	1	0
V2	1	0	1	1	0	0	0	0	0
V3	1	1	0	1	0	1	1	1	0
V4	0	1	1	0	1	0	0	0	0
V5	0	0	0	1	0	0	1	1	0
V6	0	0	1	0	0	0	0	0	1
V7	0	0	1	0	1	0	0	1	0
V8	1	0	1	0	1	0	1	0	0
V9	0	0	0	0	0	1	0	0	0

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

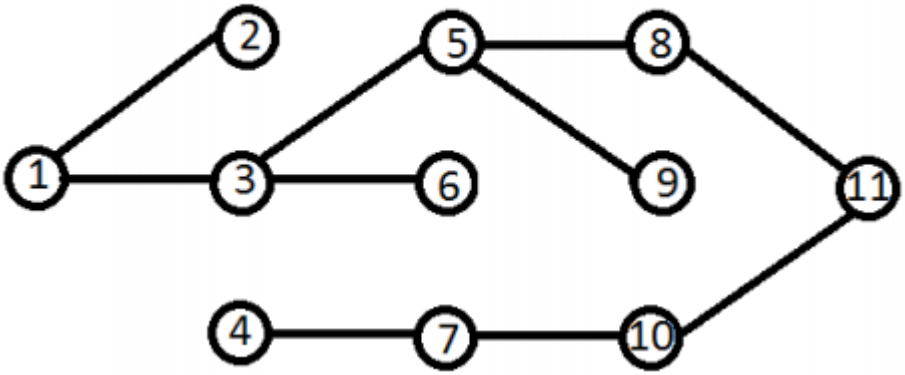
4



Краскала

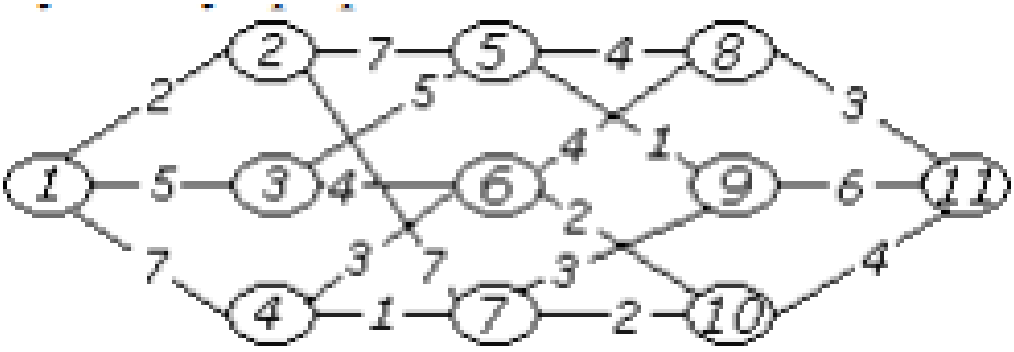


Прима



Завдання 2.

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




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

int create(int n, int A[11][11]) {
  for (int i = 0; i < 11; i++) {
    for (int j = 0; j < 11; j++) {
      A[i][j] = 0;
    }
```

```
  }
  for (int i = 0; i < 11; i++) {
    A[i][i] = i + 1;
  }
  return A[11][11];
}
void RemoveDuplicats(int n, int A[11][11]) {
  for (int i = 0; i < 11; i++) {
    for (int j = 0; j < 11; j++) {
      if (j < i) {
        A[i][j] = 0;
      }
    }
  }
  for (int i = 0; i < 11; i++) {
    for (int j = 0; j < 11; j++) {
      cout << A[i][j] << " ";
    }
    cout << endl;
  }
}
int NotOne(int n, int A[11][11],int f, int s) {
  int tmp,tmp1;
  for (int i = 0; i < 11; i++) {
    tmp = tmp1 = 0;
    for (int j = 0; j < 11; j++) {
      if (A[i][j] == f) {
        tmp = 1;
      }
    }
    for (int k = 0; k < 11; k++) {
      if (A[i][k] == s) {
        tmp1 = 1;
      }
    }
    if (tmp && tmp1) {
      return 0;
    }
  }
```

```
 return 1;
}

void add(int n, int A[11][11], int f, int s) {
 int tmp;
 for (int i = 0; i < 11; i++) {
  for (int j = 0; j < 11; j++) {
   if (A[i][j] == s) {
    tmp = i;
   }
  }
 }
 for (int i = 0; i < 11; i++) {
  for (int j = 0; j < 11; j++) {
   if (A[i][j] == f) {
      for (int k = 0; k < 11; k++) {
      if (A[tmp][k]) {
       A[i][k] = A[tmp][k];
       A[tmp][k] = 0;
       }
      }
     }
   }
  }

 }

int main()
{
 int MS[11][11] = {
  { 0, 2, 7, 1, 0, 0, 0, 0, 0, 0, 0 },
  { 2, 0, 0, 0, 2, 0, 7, 0, 0, 0, 0 },
  { 7, 0, 0, 0, 1, 4, 0, 0, 0, 0, 0 },
  { 2, 0, 0, 0, 2, 0, 7, 0, 0, 0, 0 },
  { 1, 0, 0, 0, 0, 3, 5, 0, 0, 0, 0 },
  { 0, 2, 1, 0, 0, 0, 0, 4, 5, 0, 0 },
  { 0, 0, 4, 3, 0, 0, 0, 6, 0, 2, 0 },
  { 0, 7, 0, 5, 0, 0, 0, 0, 3, 7, 0 },
  { 0, 0, 0, 0, 4, 6, 0, 0, 0, 0, 3 },
  { 0, 0, 0, 0, 0, 2, 7, 0, 0, 0, 4 },
  { 0, 0, 0, 0, 0, 0, 0, 3, 4, 4, 0 }
```

```
  };
  RemoveDuplicats(11, MS);
  for (int i = 1; i <= 7; i++) {
    cout << endl << "Edges with weight  " << i <<": ";
    for (int j = 1; j <= 11; j++) {
      for (int k = 1; k <= 11; k++) {
        if (MS[j - 1][k - 1] == i) {
          cout << " (" << j << ";" << k << ") ";
        }
      }
    }
    cout << endl;
  }

  int B[11][11];
  create(11, B);
  cout << endl << endl << "New Tree:" << endl;
    for (int i = 1; i <= 7; i++) {

      for (int j = 1; j <= 11; j++) {
        for (int k = 1; k <= 11; k++) {
          if (MS[j - 1][k - 1] == i && NotOne(11,B,j,k)){
            add(11, B, j, k);
            cout << " (" << j << ";" << k << ") ";
            cout << endl;
          }

        }
      }
    }

    cout << endl;

    return 0;
}
```