МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Лабораторна робота №4

з дисципліни

«Дискретна математика»

Виконала:

Студентка КН-112

Пихней Вероніка

Викладач:

Мельникова Н.І.

Львів - 2019р.

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№12

Завдання № 1.

Розв'язати на графах наступні задачі:

1. Виконати наступні операції над графами:

   1) знайти доповнення до першого графу,

   2) об’єднання графів,

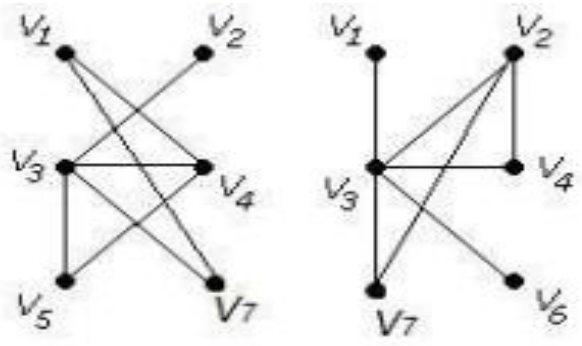
   3) кільцеву суму G1 та G2 (G1+G2),

   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),

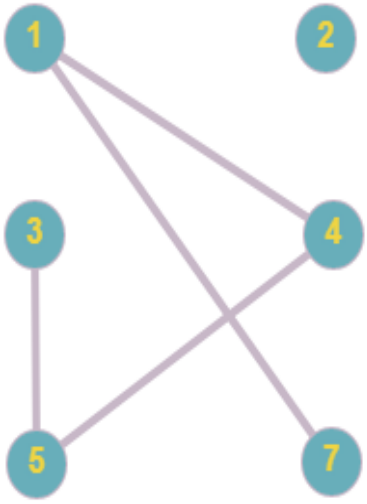
   6) добуток графів.

12

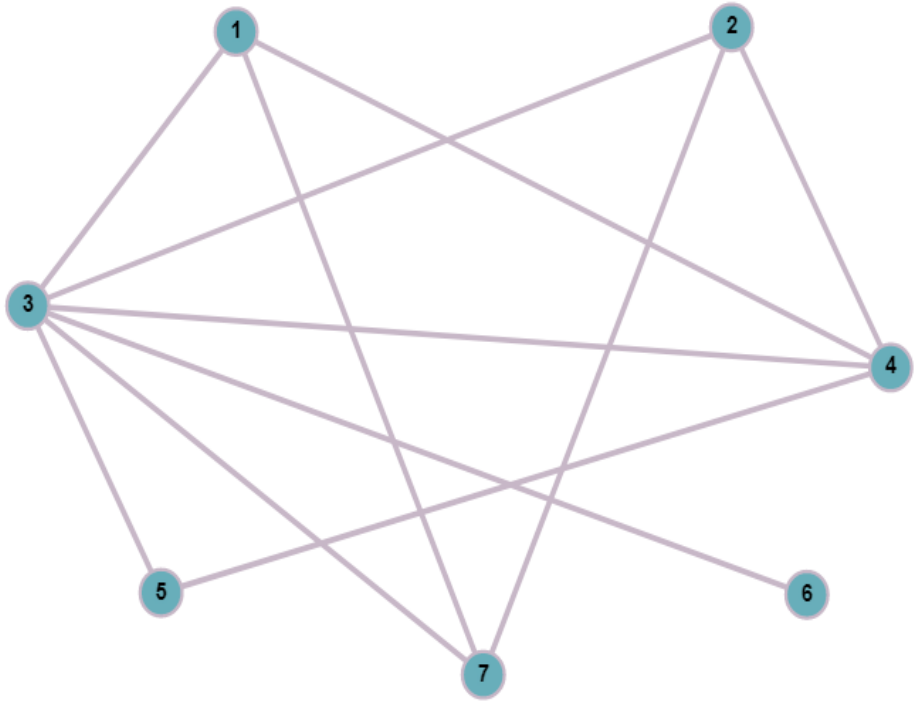


1)Доповнення

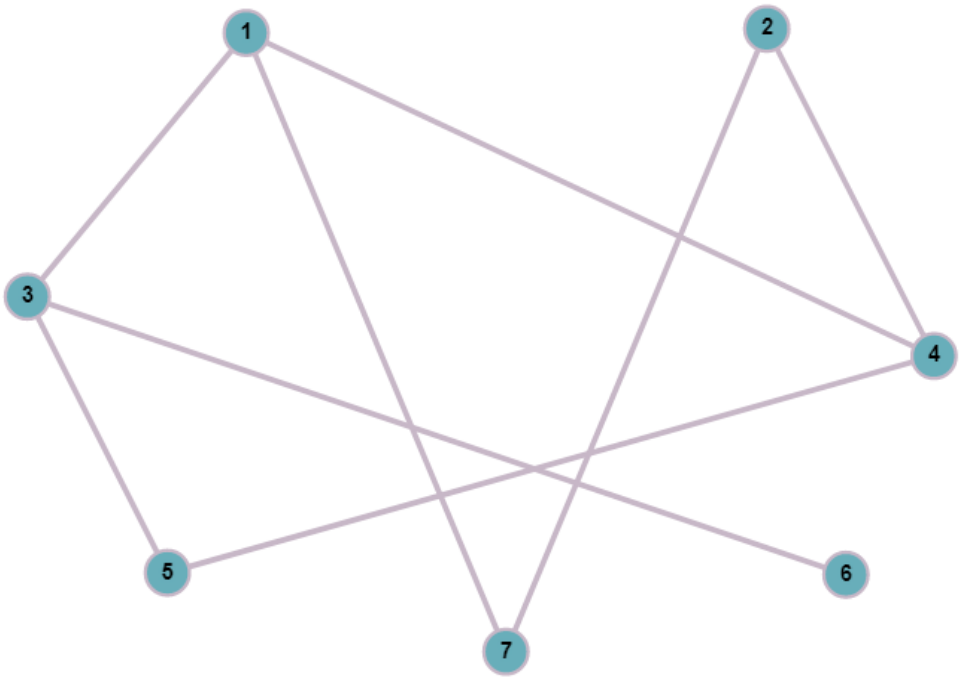
G1\G2:



2)об’єднання графів:

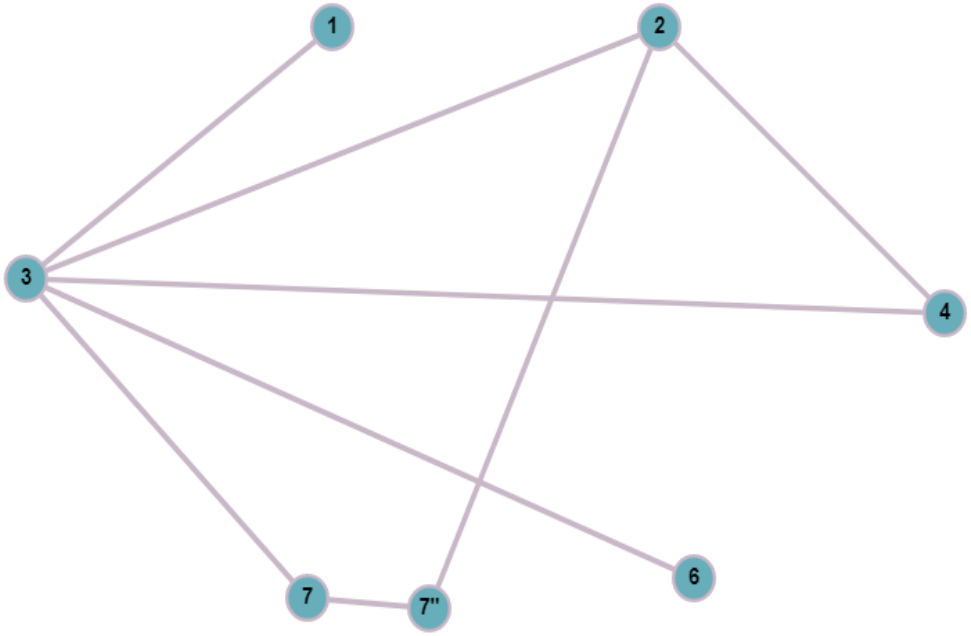


3)Кільцева сума:



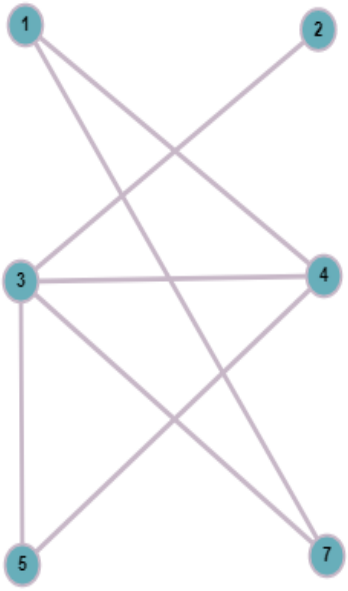
4)Розщепити вершину у другому графі:

Розщепимо вершину 7

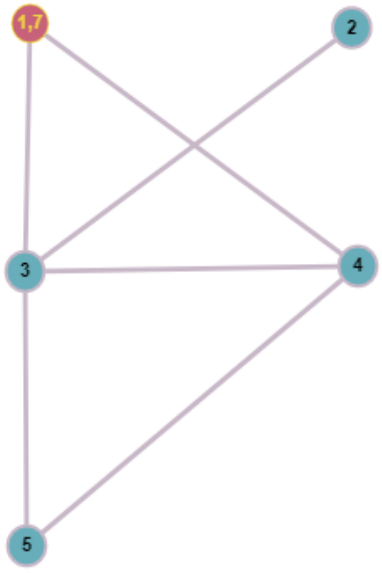


5)Виділити підграф A,що складається з V={1,3,7} в G1 і знайти стягнення A в G1

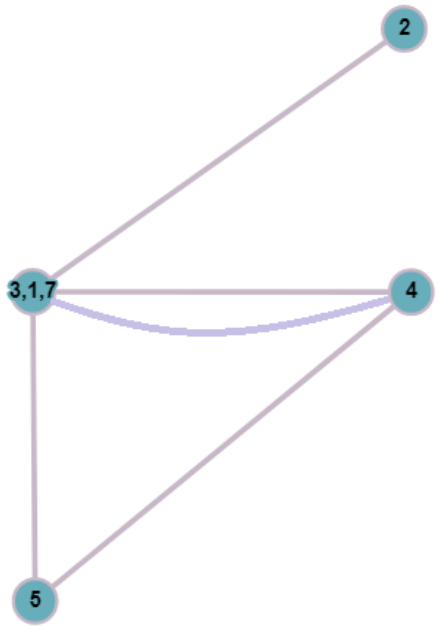
G1



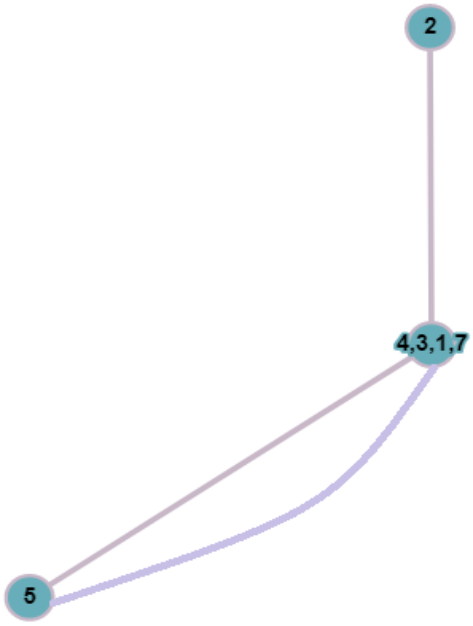
Стягуємо 7 в 1



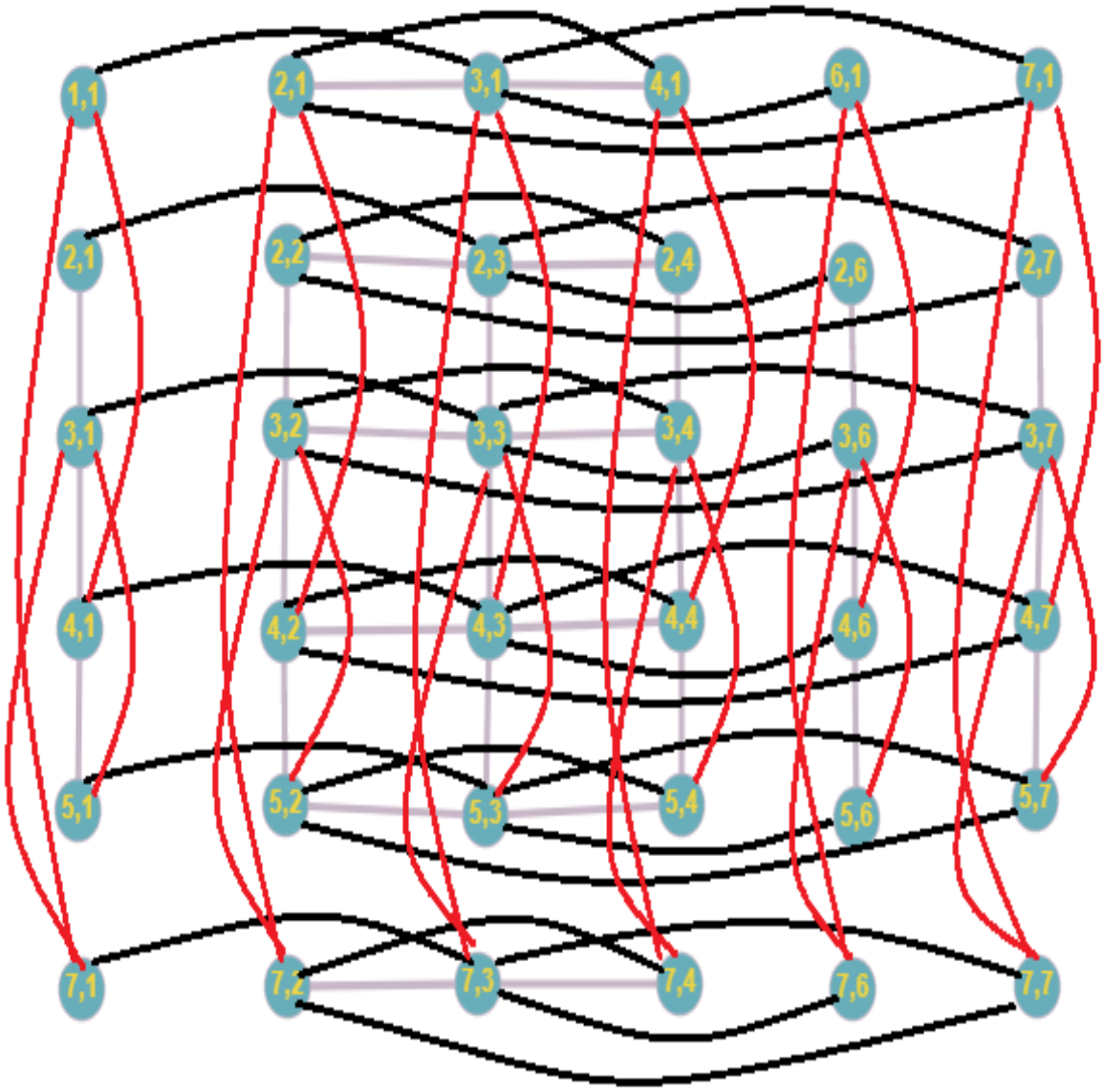
Стягуємо 1,7 в 3



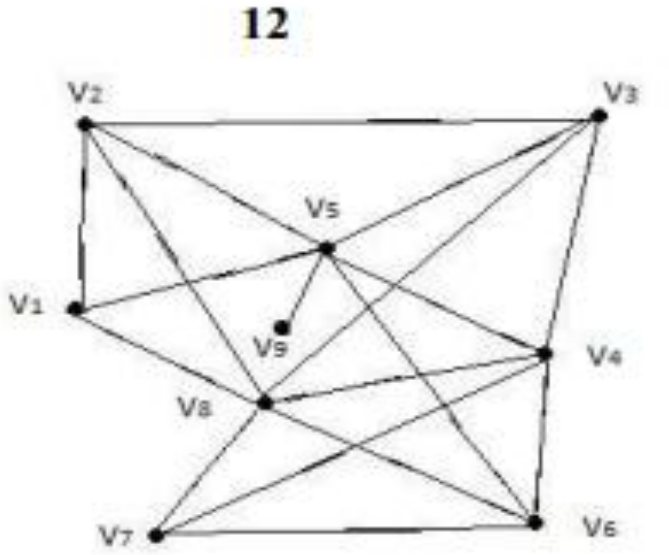
Стягуємо 3,7,1 в 4



6)Добуток графів



2. Знайти таблицю суміжності та діаметр графа.



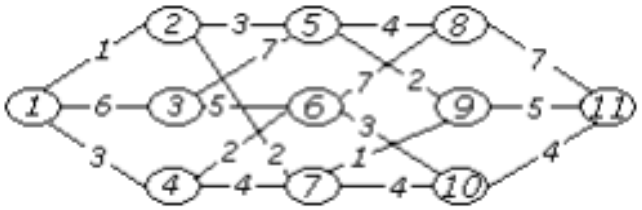
	1)	2)	3)	4)	5)	6)	7)	8)	9)
1)	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**1**	**0**
2)	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**
3)	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**
4)	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**
5)	**1**	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**1**
6)	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**0**
7)	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**0**
8)	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**0**
9)	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**

Найдовший шлях від V7 до V9 і V8 до V9

Діаметр = 3

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

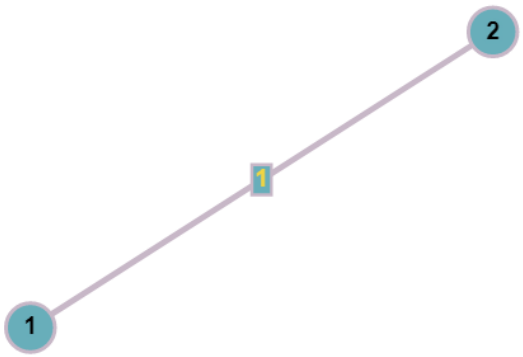
12



Метод Краскала

V={1,2}

E={(1,2)}

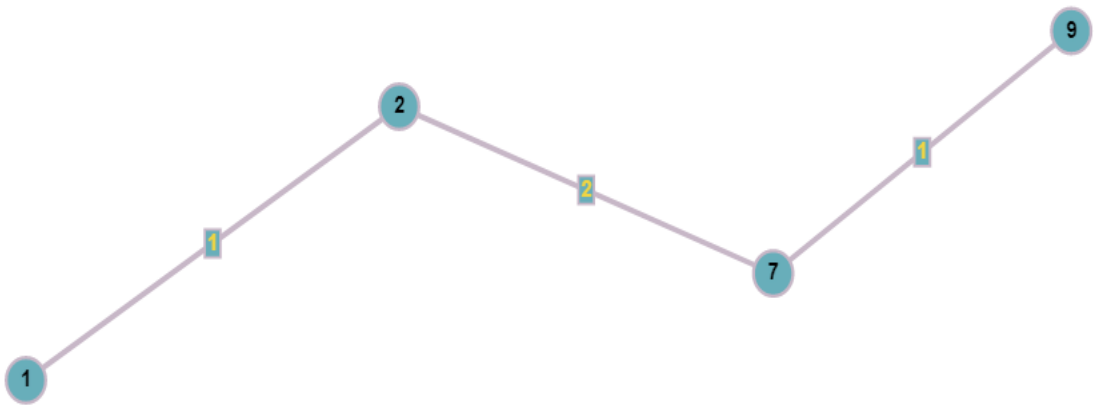


V={1,2,7,9}

E={(1,2),(7,9)}

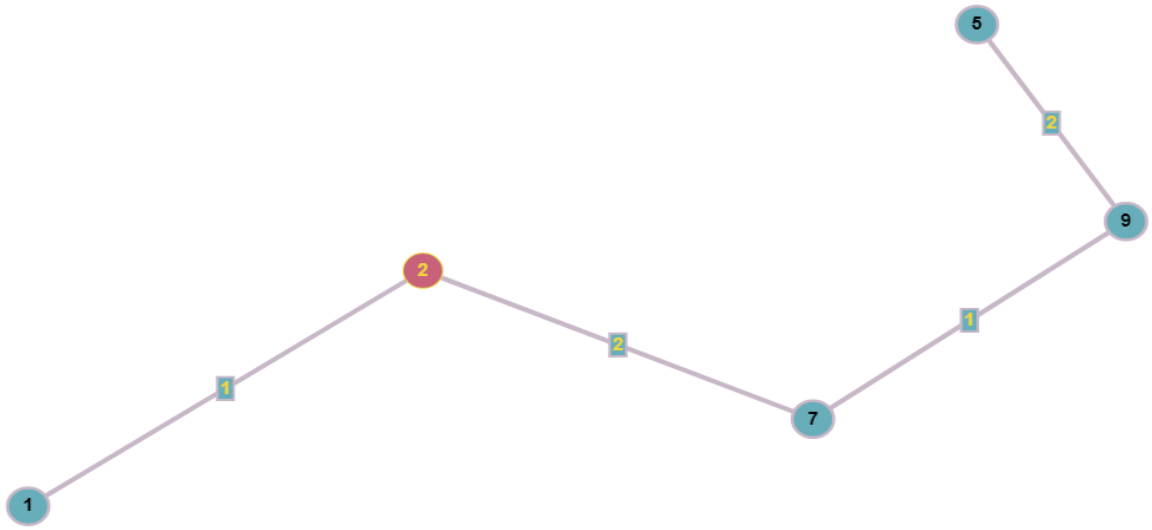
V={1,2,7,9}

E={(1,2),(7,9),(2,7)}



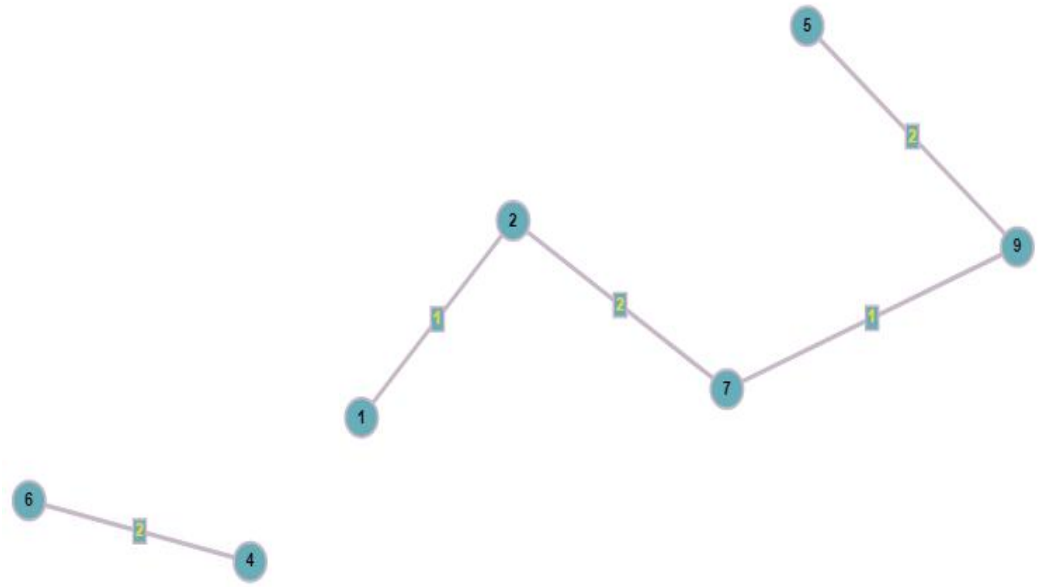
V={1,2,7,9,5}

E={(1,2),(7,9),(2,7),(5,9)}



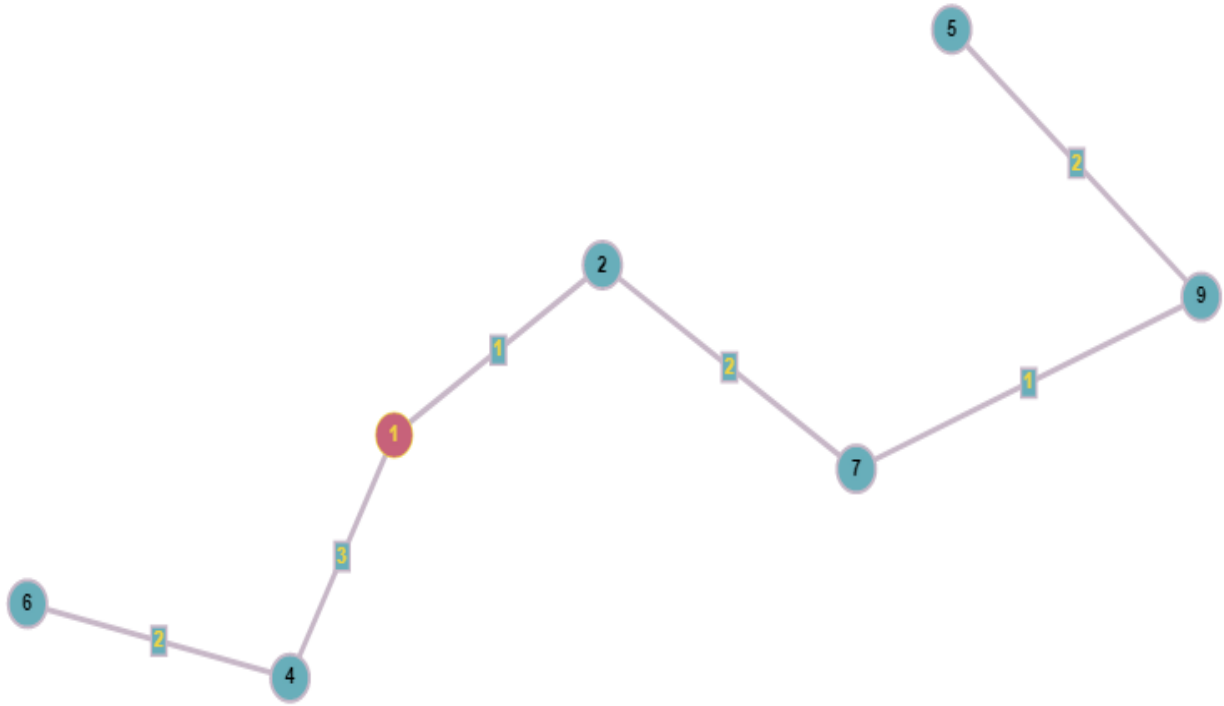
V={1,2,7,9,5,4,6}

E={(1,2),(7,9),(2,7),(5,9),(4,6)}



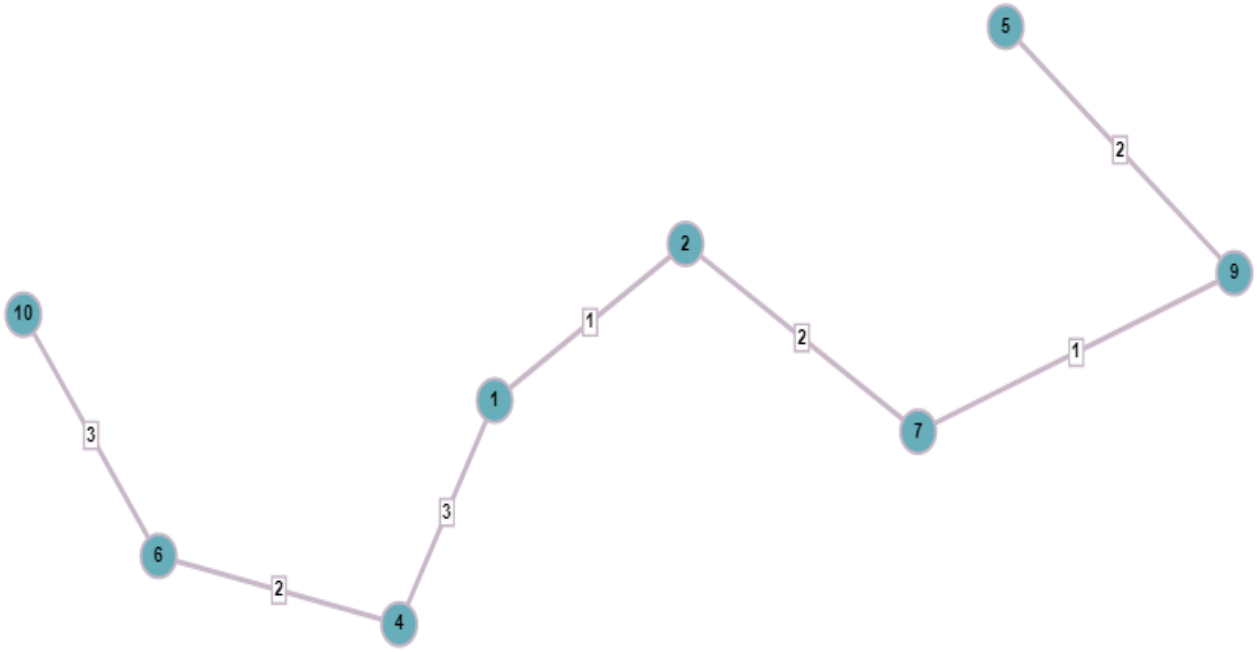
V={1,2,7,9,5,4,6}

E={(1,2),(7,9),(2,7),(5,9),(4,6),(1,4)}



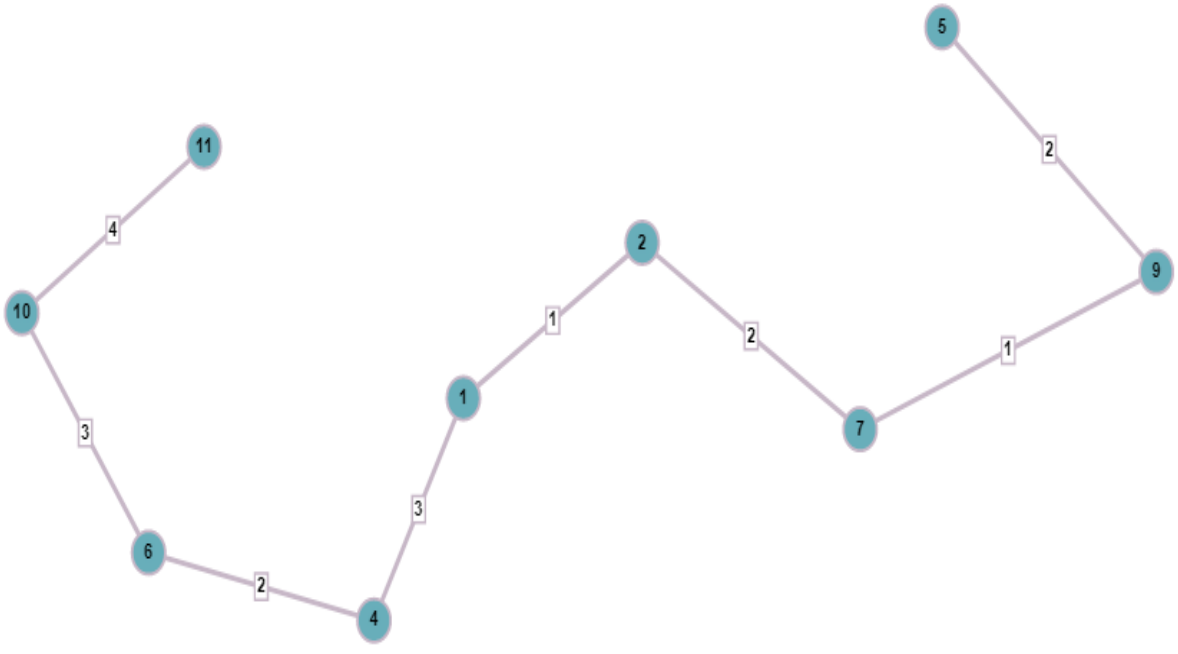
V={1,2,7,9,5,4,6,10}

E={(1,2),(7,9),(2,7),(5,9),(4,6),(1,4)(6,10)}



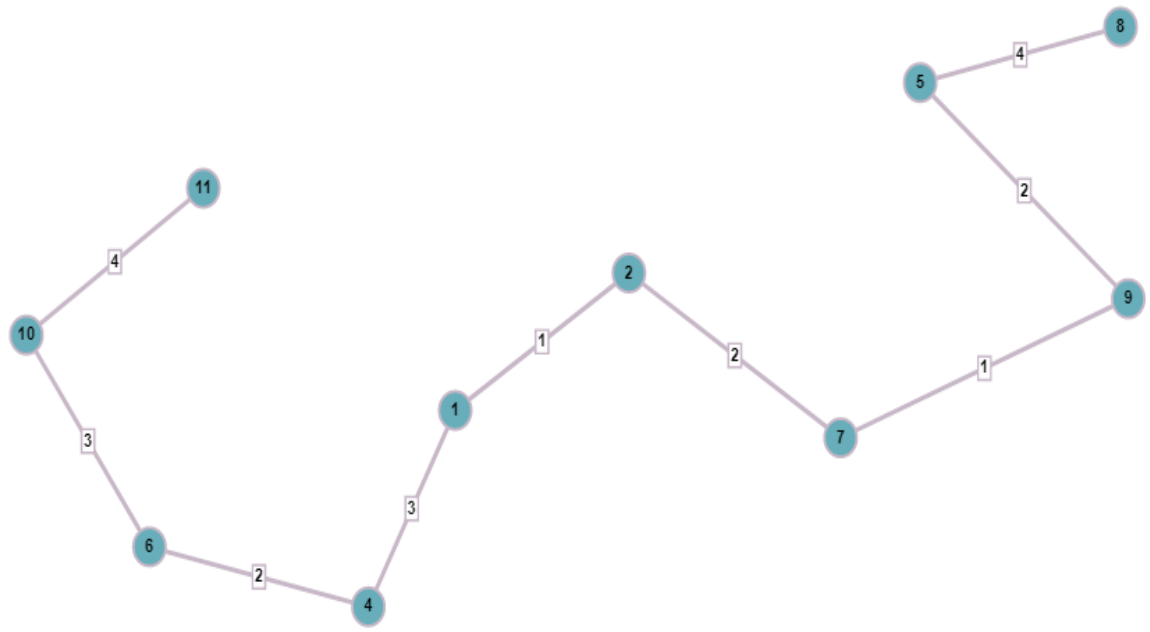
V={1,2,7,9,5,4,6,10,11}

E={(1,2),(7,9),(2,7),(5,9),(4,6),(1,4),(6,10),(10,11)}



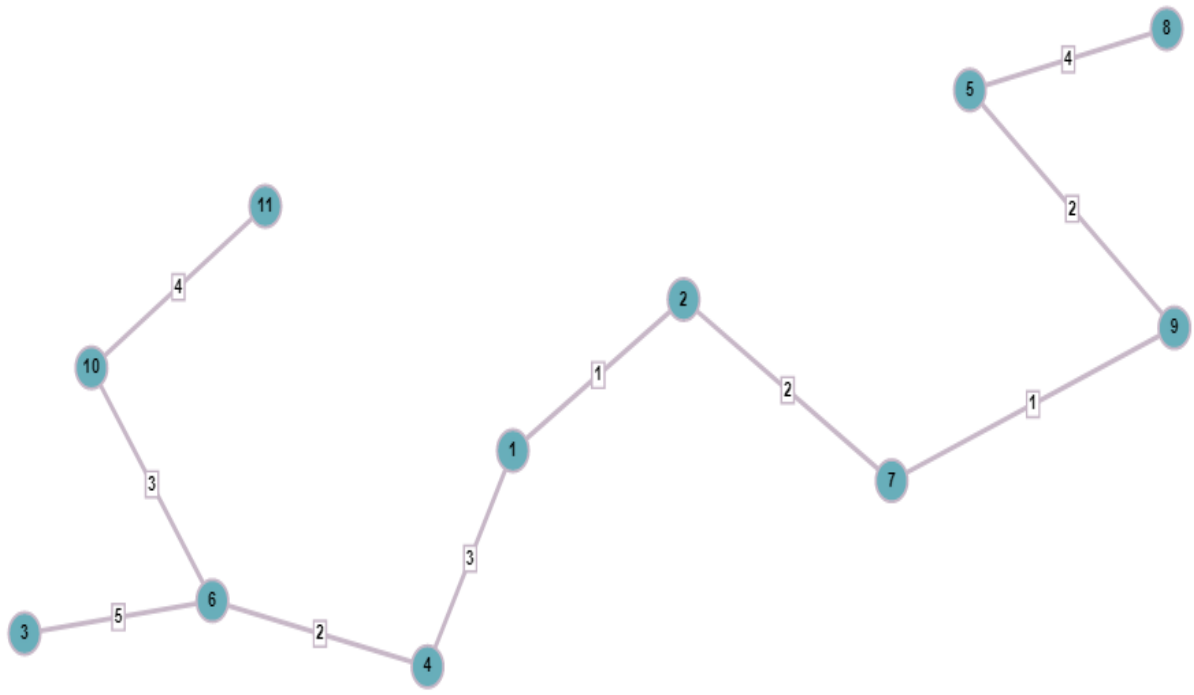
V={1,2,7,9,5,4,6,10,11,8}

E={(1,2),(7,9),(2,7),(5,9),(4,6),(1,4),(6,10),(10,11),(5,8)}



V={1,2,7,9,5,4,6,10,11,8,3}

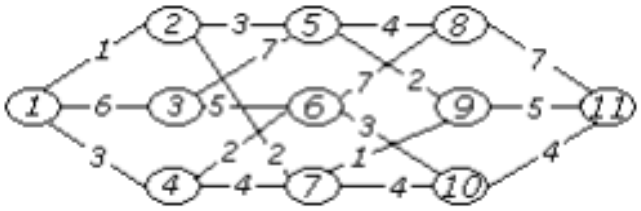
E={(1,2),(7,9),(2,7),(5,9),(4,6),(1,4),(6,10),(10,11),(5,8),(6,3)}



Weight=27

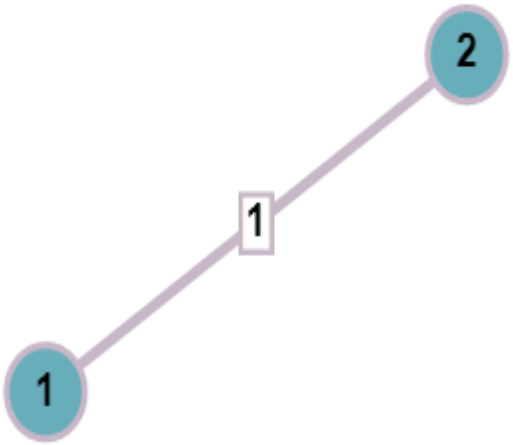
Метод Прима:

12



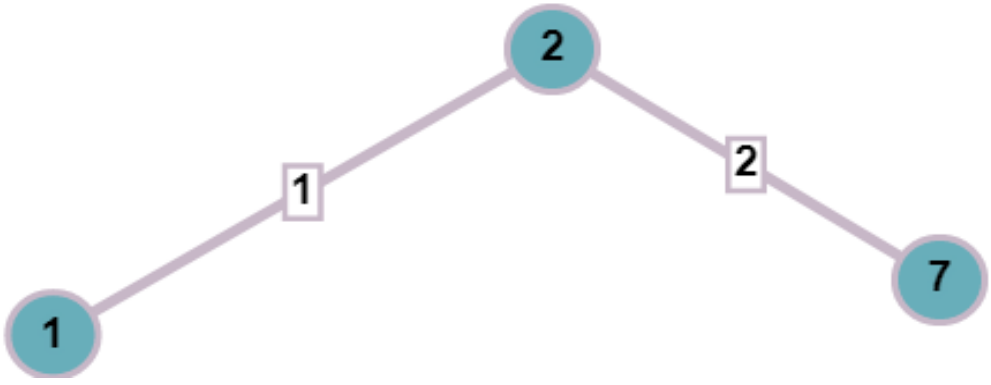
V={1,2}

E={(1,2)}



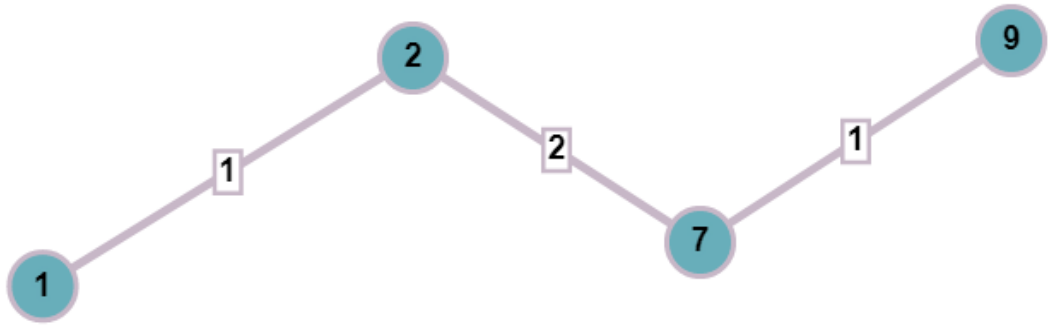
V={1,2,7}

E={(1,2),(2,7)}



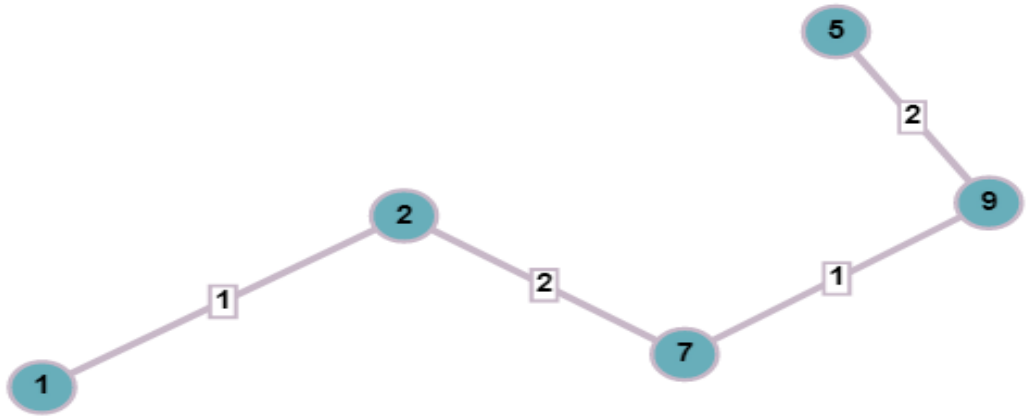
V={1,2,7,9}

E={(1,2),(2,7),(7,9)}



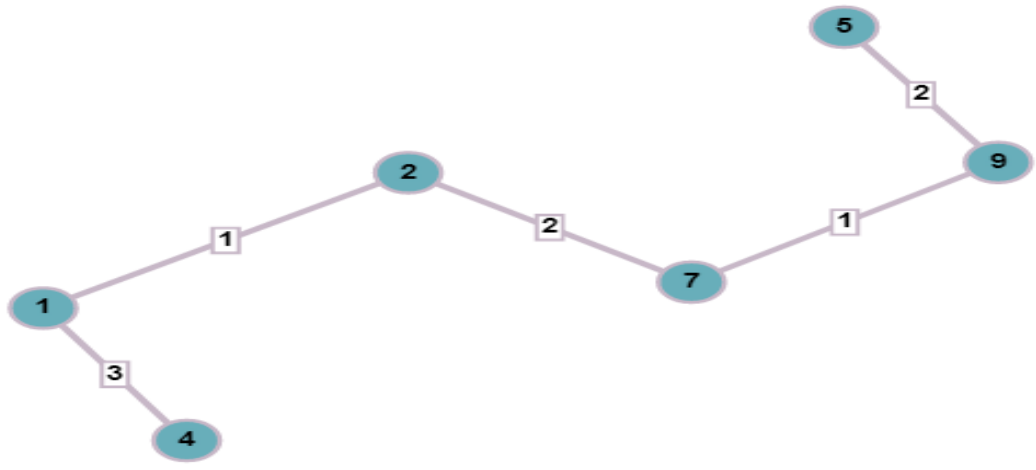
V={1,2,7,9,5}

E={(1,2),(2,7),(7,9),(9,5)}



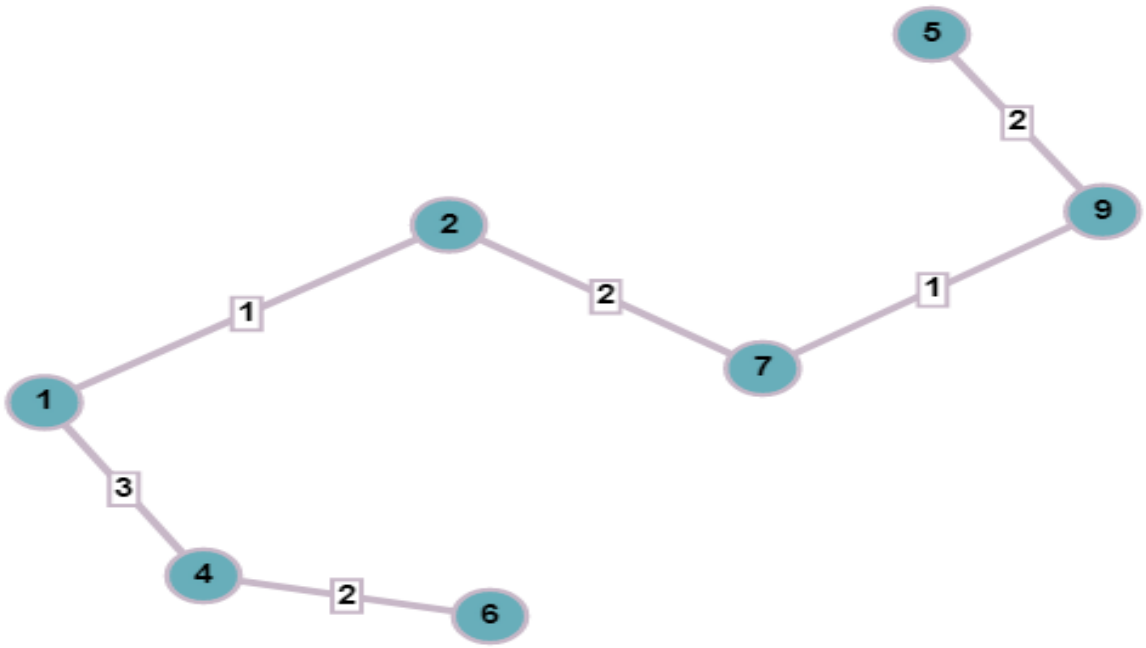
V={1,2,7,9,5,4}

E={(1,2),(2,7),(7,9),(9,5),(1,4)}



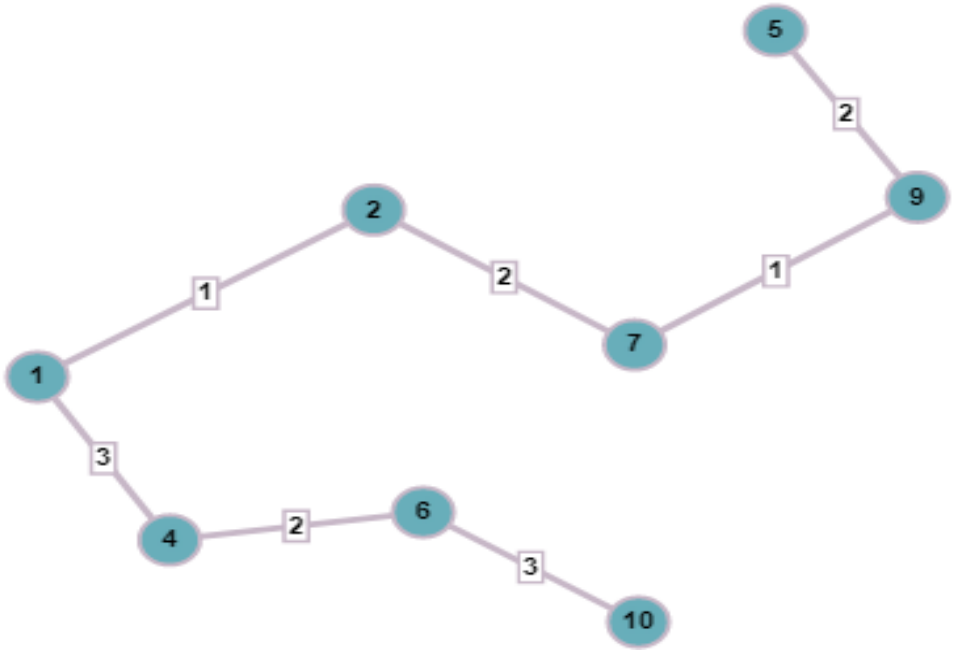
V={1,2,7,9,5,4,6}

E={(1,2),(2,7),(7,9),(9,5),(1,4),(4,6)}



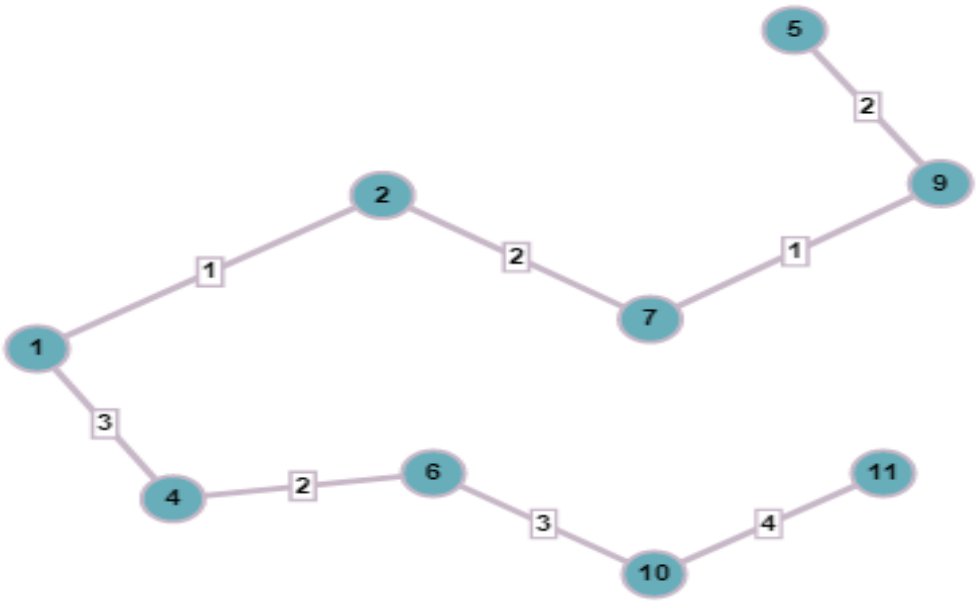
V={1,2,7,9,5,4,6,10}

E={(1,2),(2,7),(7,9),(9,5),(1,4),(4,6),(6,10)}



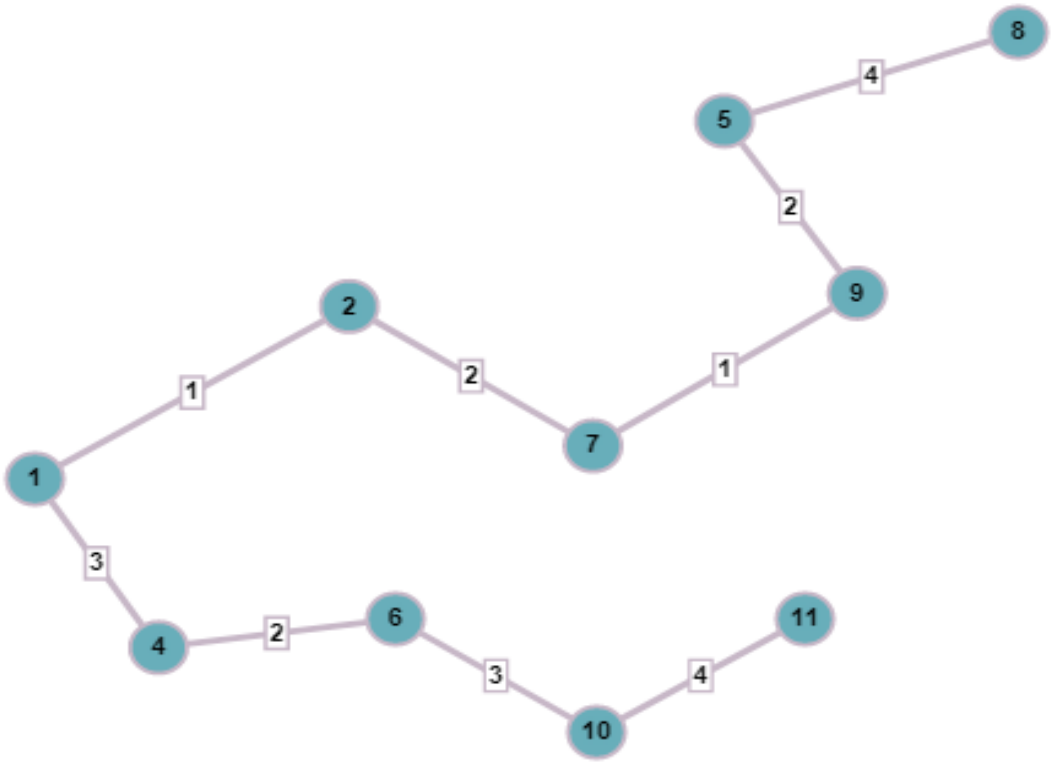
V={1,2,7,9,5,4,6,10,11}

E={(1,2),(2,7),(7,9),(9,5),(1,4),(4,6),(6,10),(10,11)}



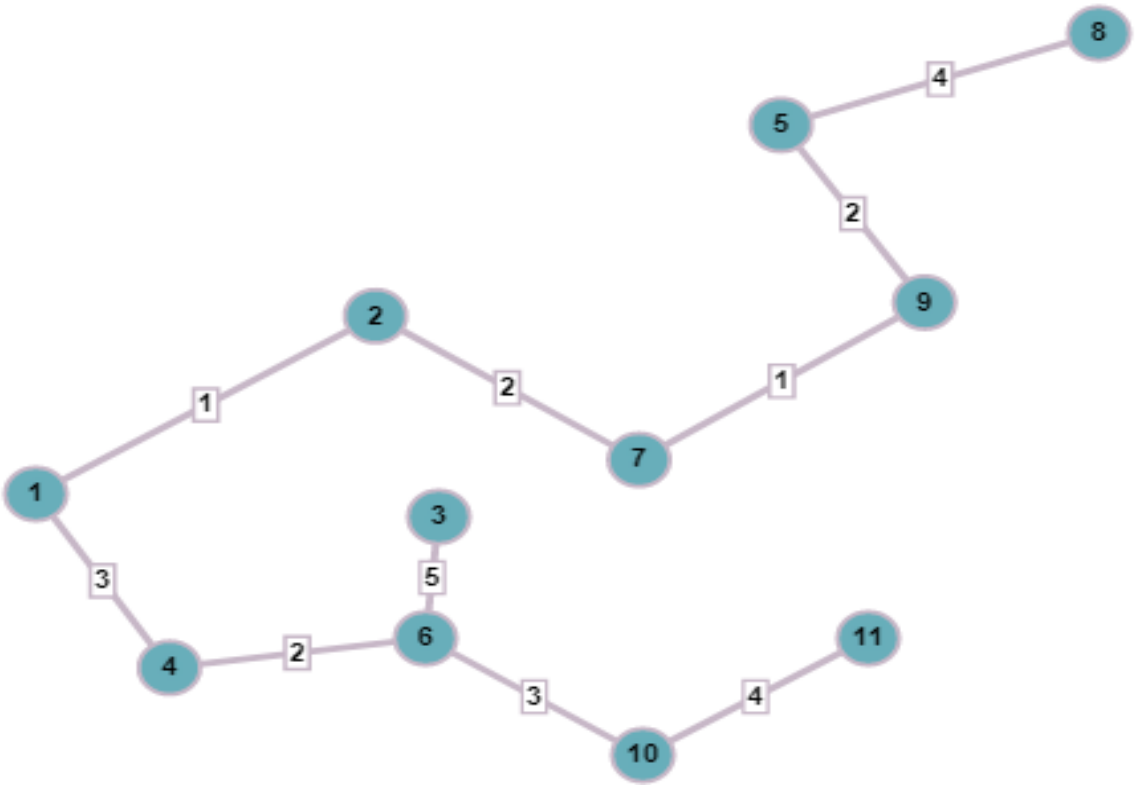
V={1,2,7,9,5,4,6,10,11,8}

E={(1,2),(2,7),(7,9),(9,5),(1,4),(4,6),(6,10),(10,11),(5,8)}



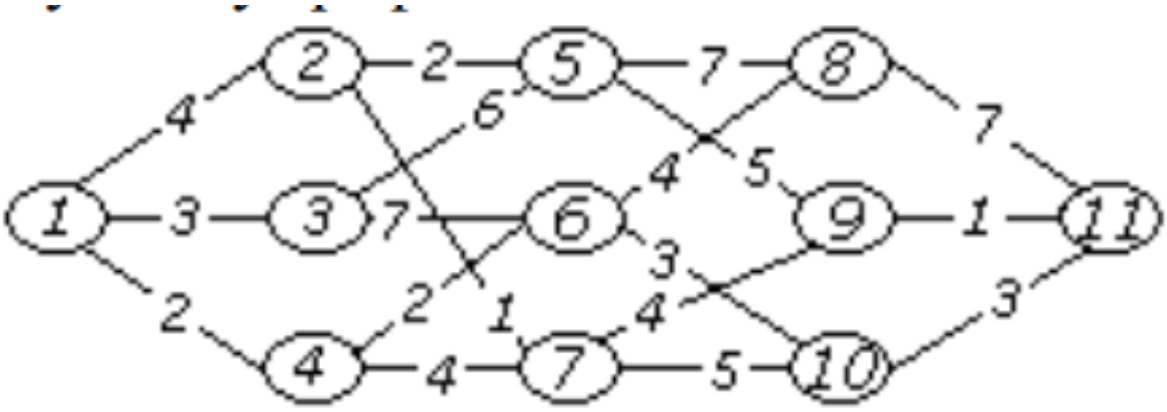
V={1,2,7,9,5,4,6,10,11,8,3}

E={(1,2),(2,7),(7,9),(9,5),(1,4),(4,6),(6,10),(10,11),(5,8),(6,3)}



Частина №2

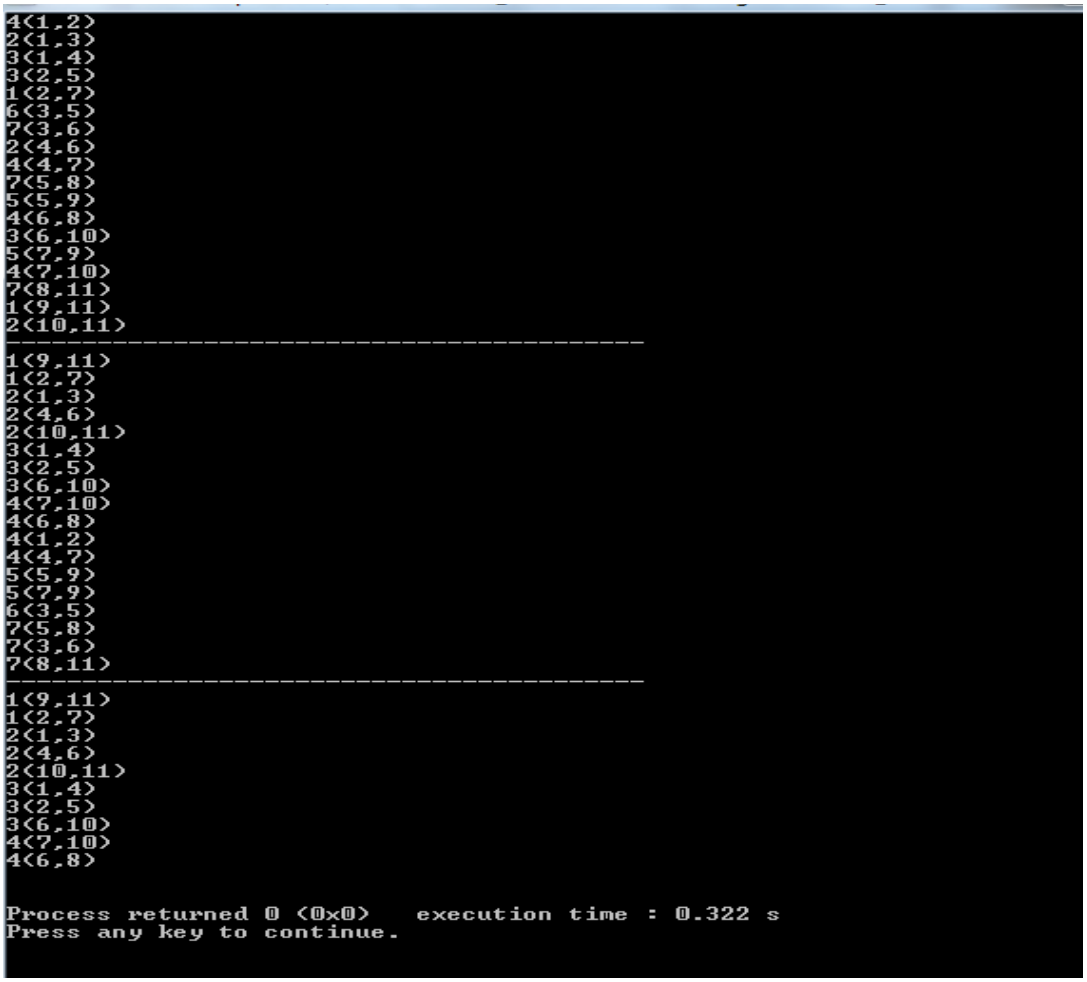
Реалізувати пошук кістякового дерева,алгоритмом Краскала.



```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

const int SIZE=11;

struct rebro
   {
    int e1;
    int e2;
    int weight;
   };

bool operator < (const rebro& rebro1, const rebro& rebro2)//моб сортування графа відбувалось по вазі ребра.weight
{
    return rebro2.weight < rebro1.weight;
}

bool checking_top(vector<int> mas,int top)
{
    for(int i=0;i<mas.size();i++)
    {
        if(mas[i]==top) return true;
    }
    return false;
}

bool checking_loop(vector<int> mas,int top1,int top2)
{
    bool res top1=false, res top2=false;
```

```cpp
    for(int i=0;i<mas.size();i++)
    {
        if(mas[i]==top1) res_top1=true;
        if(mas[i]==top2) res_top2=true;
    }
    return res_top1&&res_top2;
}

bool trees_unit(vector<int> trees[SIZE], int first, int second)
{
    int i,j,first_tree=0,second_tree=0;

    for(i=0; i<SIZE; i++)
    {
        if(trees[i].size()!=0)
        {
            for(j=0;j<trees[i].size();j++)
            {
                if(trees[i][j]==first) first_tree=i;
                if(trees[i][j]==second) second_tree=i;
            }

        }
    }

    for(i=0; i<trees[second_tree].size(); i++)trees[first_tree].push_back(trees[second_tree][i]);
    trees[second_tree].clear();
```

```
int main()
{
    vector<rebro> Graf, Graf_MOT;
    vector<int> trees[SIZE];

    int Matrix[SIZE][SIZE]={
        {0,4,2,3,0,0,0,0,0,0,0},
        {4,0,0,0,3,0,1,0,0,0,0},
        {2,0,0,0,6,7,0,0,0,0,0},
        {3,0,0,0,0,2,4,0,0,0,0},
        {0,3,6,0,0,0,0,7,5,0,0},
        {0,0,7,2,0,0,0,4,0,3,0},
        {0,1,0,4,0,0,0,0,5,4,0},
        {0,0,0,0,7,4,0,0,0,0,7},
        {0,0,0,0,5,0,5,0,0,0,1},
        {0,0,0,0,0,3,4,0,0,0,2},
        {0,0,0,0,0,0,0,7,1,2,0}
    };

    for(int i=0;i<SIZE;i++)
    {
        for(int j=i;j<SIZE;j++)
        {
            if(Matrix[i][j]!=0)
            {
                rebro help;
                cout<<Matrix[i][j]<<"("<<i+1<<","<<j+1<<")"<<endl;
                help.e1=i;
                help.e2=j;
                help.weight=Matrix[i][j];
                Graf.push_back(help);

            }
        }
    }

    cout<<"----------------------------------------"<<endl;

    //сортуєм граф по зростанню ваги ребер
    sort(Graf.rbegin(),Graf.rend());
    for(int i=0; i<Graf.size();i++) cout<<Graf[i].weight<<"("<<Graf[i].e1+1<<","<<Graf[i].e2+1<<")"<<endl;

    cout<<"----------------------------------------"<<endl;

    //початкова ініціалізація масиву trees
    for(int i=0;i<SIZE;i++)trees[i].push_back(i);

     for(int i=0; i<Graf.size(); i++)
     {
        for(int j=0; j<SIZE; j++)
        {
                                    bool res_top1=checking_top(trees[j],Graf[i].e1);
                                    bool res_top2=checking_top(trees[j],Graf[i].e2);
                                    bool res_loop=!checking_loop(trees[j],Graf[i].e1,Graf[i].e2);

                                    if((res_top1||res_top2)&&res_loop) //if(!checking_loop(trees[j],Graf[i].e1,Graf[i]
                                    {
                                       trees_unit(trees,Graf[i].e1,Graf[i].e2);

                                       rebro help;
                                       help.e1=Graf[i].e1;
                                       help.e2=Graf[i].e2;
                                       help.weight=Graf[i].weight;

                                    Graf_MOT.push_back(help);

                                     break;
                                    }
                                }
                            }
     }

    for(int i=0; i<Graf_MOT.size();i++) cout<<Graf_MOT[i].weight<<"("<<Graf_MOT[i].e1+1<<","<<Graf_MOT[i].e2+1<<")"<<endl;

    cout <<endl;
    return 0;
```


4(1,2)
2(1,3)
3(1,4)
3(2,5)
1(2,7)
6(3,5)
7(3,6)
2(4,6)
4(4,7)
7(5,8)
5(5,9)
4(6,8)
3(6,10)
5(7,9)
4(7,10)
7(8,11)
1(9,11)
2(10,11)
1(9,11)
1(2,7)
2(1,3)
2(4,6)
2(10,11)
3(1,4)
3(2,5)
3(6,10)
4(7,10)
4(6,8)
4(1,2)
4(4,7)
5(5,9)
5(7,9)
6(3,5)
7(5,8)
7(3,6)
7(8,11)
1(9,11)
1(2,7)
2(1,3)
2(4,6)
2(10,11)
3(1,4)
3(2,5)
3(6,10)
4(7,10)
4(6,8)
Process returned 0 (0x0) execution time : 0.322 s
Press any key to continue.