Міністерство освіти і науки України

Національному університеті "Львівська Політехніка"

Кафедра систем штучного інтелекту

Розрахункова робота

з дисципліни

≪ Дискретна математика ≫

Виконав:

студент групи КН-114

Павлик Богдан

Викладач:

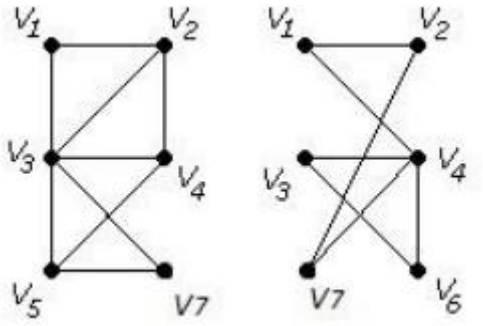
Мельникова Н.І.

Львів - 2019р.

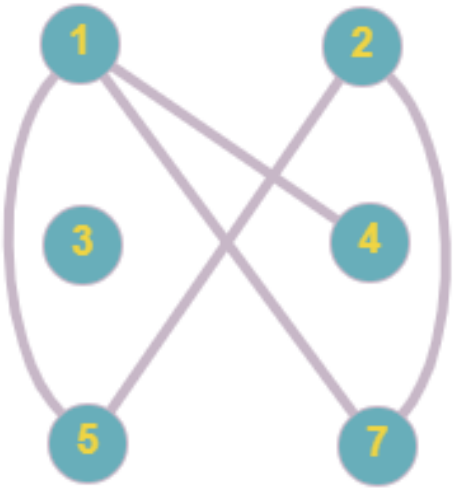
Варіант 6

Завдання № 1

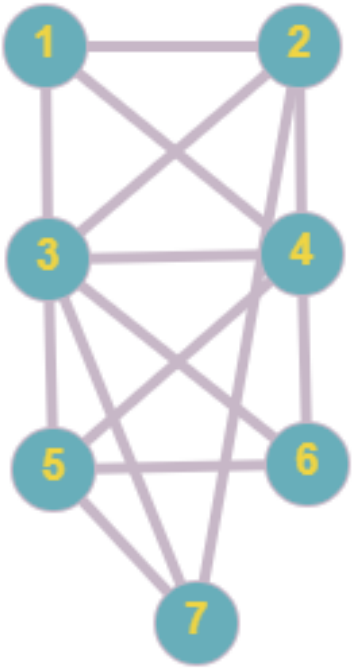
Виконати наступні операції над графами: 1) знайти доповнення до першого графу, 2) об'єднання графів, 3) кільцеву сумму G1 та G2 (G1+G2), 4) розмножити вершину у другому графі, 5) виділити підграф А - що скадається з 3-х вершин в G1 6) добуток графів.



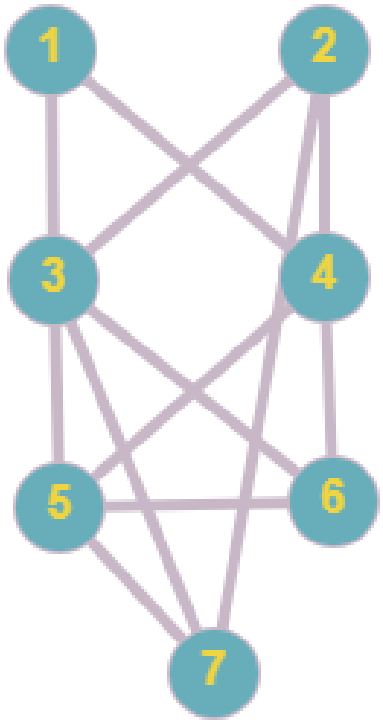
1) Доповнення до першого графа



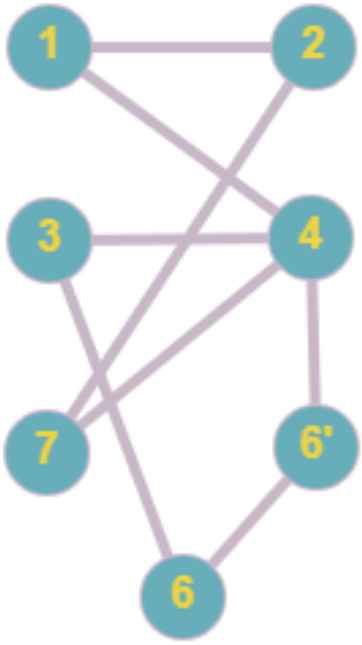
2) Об'єднання графів



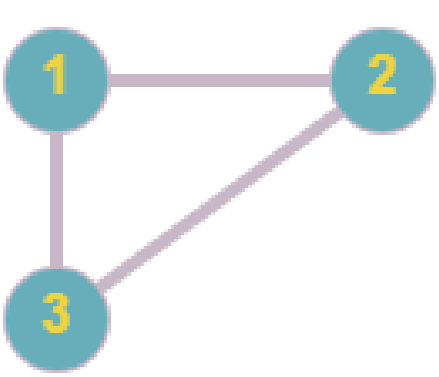
3) Кільцева сума G1 і G2

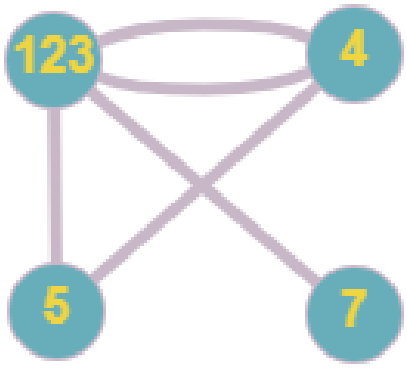


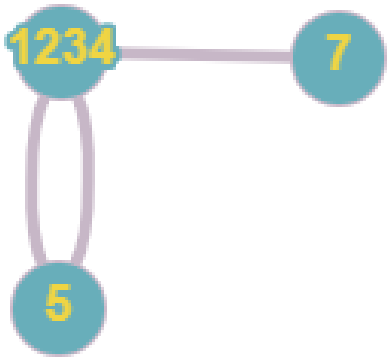
4) Розщеплюємо вершину V6



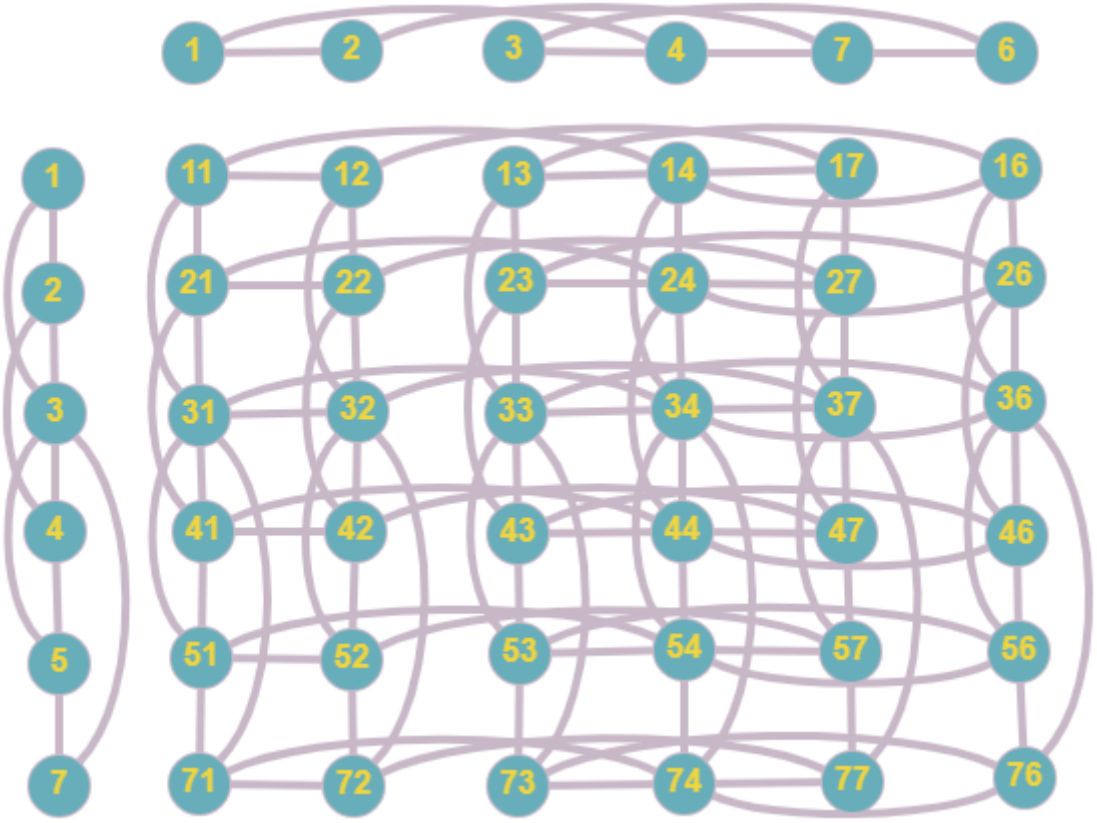
5) Виділяємо підграф V1V2V3





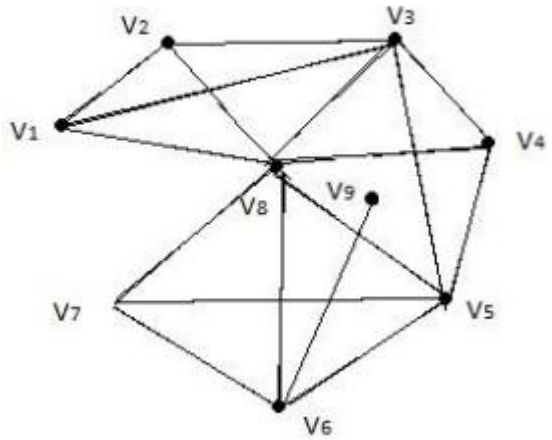


6) добуток графів



Завдання № 2

Скласти таблицю суміжності для орграфа.



Таблиця суміжності

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

Завдання № 3

Для графа з другого завдання знайти діаметр.

Діаметр графа - 3, оскільки це максимальний ексцентриситет вершини, тобто відстань від однієї вершини до найвіддаленішої від неї вершини.

Завдання № 4

Для графа з другого завдання виконати обхід дерева вглиб (варіант закінчується на непарне число) або вшир (закінчується на парне число).

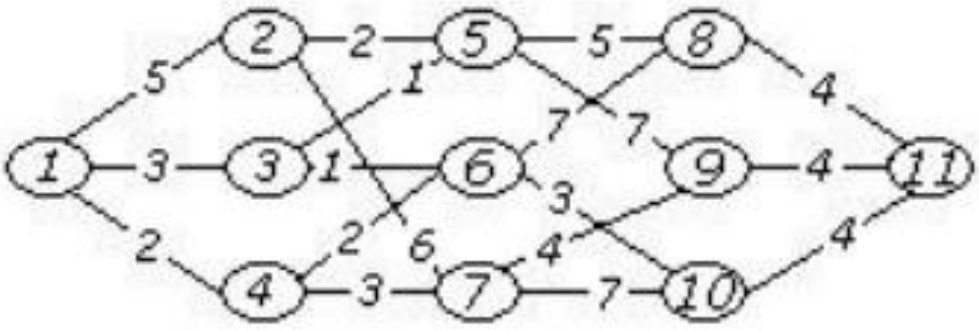
Вершина	BFS вершини	Вміст черги
v1	1	v1
v2	2	v1 v2
v3	3	v1 v2 v3
-	-	v2 v3
v8	4	v2 v3 v8
-	-	v3 v8
v4	5	v3 v8 v4
v5	6	v3 v8 v4 v5
-	-	v8 v4 v5
v6	7	v8 v4 v5 v6
v7	8	v8 v4 v5 v6 v7
-	-	v4 v5 v6 v7
-	-	v5 v6 v7
-	-	v6 v7
v9	9	v6 v7 v9
-	-	v7 v9
-	-	v9
-	-	∅

```cpp
#include <iostream>
#include <queue>
using namespace std;
int main()
{
  queue <int> queue_bfs;
  int mas[9][9] = {
  { 0,1,1,0,0,0,0,1,0 },
  { 1,0,1,0,0,0,0,1,0 },
  { 1,1,0,1,1,0,0,1,0 },
  { 0,0,1,0,1,0,0,1,0 },
  { 0,0,1,1,0,1,1,1,0 },
  { 0,0,0,0,1,0,1,1,1 },
  { 0,0,0,0,1,1,0,1,0 },
  { 1,1,1,1,1,1,1,0,0 },
  { 0,0,0,0,0,1,0,0,0 } };
  int vertices[9];
  for (int i = 0; i < 9; i++)
  vertices[i] = 0;
  queue_bfs.push(0);
  cout << "So, BFS: " << endl;
  while (!queue_bfs.empty())
  {
    int node = queue_bfs.front();
    queue_bfs.pop();
    vertices[node] = 2;
    for (int j = 0; j < 9; j++)
        {
      if (mas[node][j] == 1 && vertices[j] == 0)
      {
        queue_bfs.push(j);
        vertices[j] = 1;
      }
    }
    cout << node + 1 << " ";
```

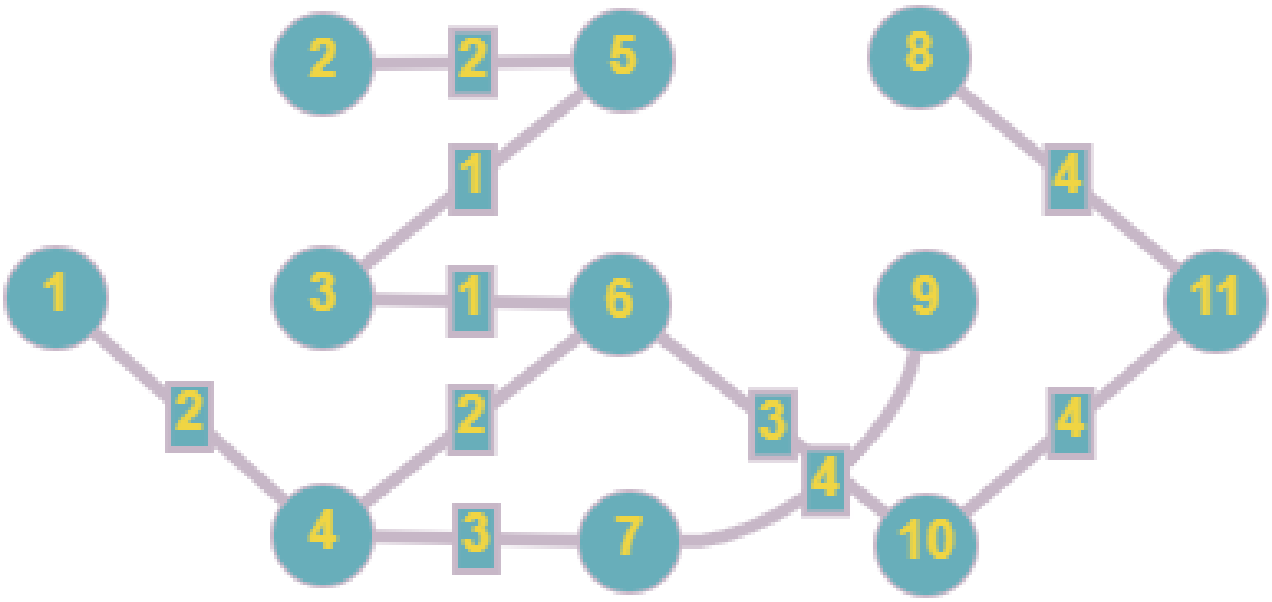
```
So, BFS:
1 2 3 8 4 5 6 7 9
```

Завдання № 5

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



1) Алгоритм Краскала



Пройдені вершини по порядку:

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

Пройдені ребра по порядку:

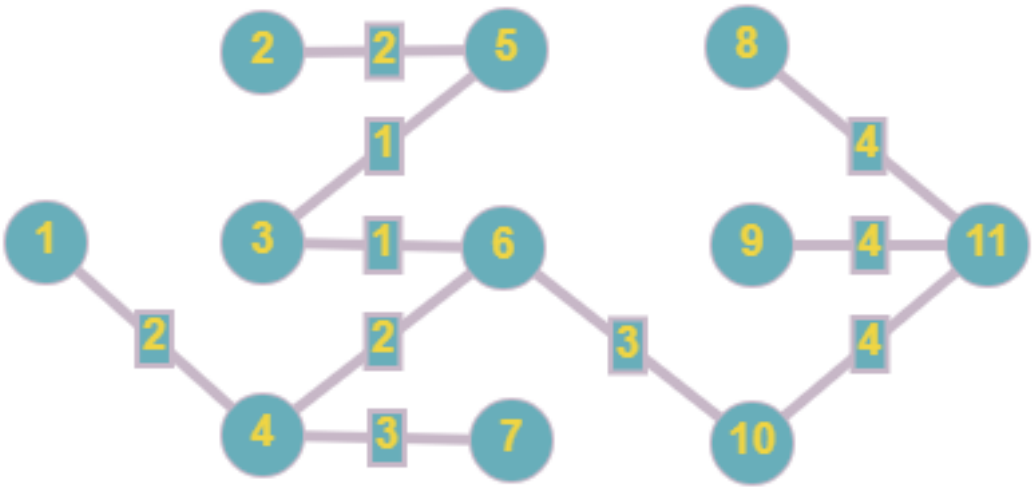
{(3,5),(3,6),(1,4),(4,6),(4,7),(6,10),(7,9),(10,11),(11,8) }

Вага дерева - 26.

```cpp
#include<iostream>
#include<vector>
#include<algorithm>
using namespace std;
int main() {
    int m=18,n=11;
    vector < pair < int, pair<int,int> > > g (m); // вага - вершина 1 - вершина 2
    cout<<"Input your graph(1 & 2 are vertexes, 3 is weight):\n";
        for(int i=0;i<m;i++)
        cin>> g[i].second.first >> g[i].second.second >> g[i].first;

    int cost = 0,l=0;
    vector < pair<int,int> > res;
    sort (g.begin(), g.end());
    vector<int> tree (n);
    for (int i=0; i<n-1; ++i)
        tree[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[a] != tree[b])
        {
            cost += l;
            res.push_back (make_pair (a, b));
            int old_id = tree[a],  new_id = tree[b];
            for (int j=0; j<n; ++j)
                if (tree[j] == old_id)
                    tree[j] = new_id;
        }
    }
    cout<<"Vertexes of graph are\n";
    for (int i = 0; i < n -1 ; i++)
      cout << res[i].first << " " << res[i].second<<endl;
    cout<<"Sum of graph is="<<cost;
}
```

```
3 6 1
4 6 2
4 7 3
5 8 5
5 9 7
6 8 7
6 10 3
7 9 4
7 10 7
8 11 4
9 11 4
10 11 4
Vertexes of graph are
3 5
3 6
1 4
2 5
4 6
4 7
6 10
7 9
8 11
9 11
Sum of graph is=26
```

2)Алгоритм Прима



Пройдені вершини по порядку:

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

Пройдені ребра по порядку:{(1,4),(4,6),(3,6),(3,5),(5,2),(4,7),(6,10),(10,11),(11,9),(11,8)}

Вага дерева - 26.

```cpp
#include <iostream>
#include <cstring>
using namespace std;
#define INF 999999
#define V 11

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

int main () {

  int number_of_edge;
  int sum = 0;
  int selected[V];
  memset (selected, false, sizeof (selected));
  number_of_edge = 0;

  selected[0] = true;

  int x;
  int y;

  cout << "Edge" << " : " << "Weight";
  cout << endl;
  while (number_of_edge < V - 1) {


     int min = INF;
     x = 0;
     y = 0;

     for (int i = 0; i < V; i++) {
       if (selected[i]) {
           for (int j = 0; j < V; j++) {
             if (!selected[j] && G[i][j]) {
                 if (min > G[i][j]) {
                     min = G[i][j];
                     x = i;
                     y = j;
                 }

             }
         }
       }
     }
     cout << x+1 <<  " - " << y+1 << " :  " << G[x][y];
     sum += G[x][y];
     cout << endl;
     selected[y] = true;
     number_of_edge++;
   }
  cout << "The weight of the way: " << sum;
  return 0;
}

```

Завдання № 6

Розв'язати задачу комівояжера для повного 8-ми вершинного графа методом «іди у найближчий», матриця вагів якого має вигляд:

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

Розглянемо 8 випадків

1) Почнімо з вершини 1

Жовтим кольором позначаю стовбець і рядок, які я видаляю

	1	2	3	4	5	6	7	8
1	-	**1**	6	7	3	5	4	6
2	**1**	-	6	5	4	2	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	1
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	12	3	4	5	6	7	8
12	-	6	5	4	**2**	3	5
3	6	-	1	7	6	1	5
4	5	1	-	6	7	5	1
5	4	7	6	-	5	6	5
6	**2**	6	7	5	-	5	4
7	3	1	5	6	5	-	7
8	5	5	1	5	4	7	-

	3	4	5	126	7	8
3	-	1	7	6	1	5
4	1	-	6	7	5	1
5	7	6	-	5	6	5
126	6	7	5	-	5	**4**
7	1	5	6	5	-	7
8	5	1	5	**4**	7	-

	3	4	5	7	1268
3	-	1	7	1	5
4	1	-	6	5	**1**
5	7	6	-	6	5
7	1	5	6	-	7
1268	5	**1**	5	7	-

	3	12684	5	7
3	-	**1**	7	1
12684	**1**	-	6	5
5	7	6	-	6
7	1	5	6	-

	126843	5	7
126843	-	7	**1**
5	7	-	6
7	**1**	6	-

	5	1268437
5	-	**6**
1268437	**6**	-

Отже, вага циклу (1 2 6 8 4 3 7 5) дорівнює 16

2) Почнімо з вершини 2

	1	2	3	4	5	6	7	8
1	-	**1**	6	7	3	5	4	6
2	**1**	-	6	5	4	2	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	1
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	21	3	4	5	6	7	8
21	-	6	7	**3**	5	4	6
3	6	-	1	7	6	1	5
4	7	1	-	6	7	5	1
5	**3**	7	6	-	5	6	5
6	5	6	7	5	-	5	4
7	4	1	5	6	5	-	7
8	6	5	1	5	4	7	-

	3	4	215	6	7	8
3	-	1	7	6	1	5
4	1	-	6	7	5	1
215	7	6	-	**5**	6	5
6	6	7	**5**	-	5	4
7	1	5	6	5	-	7
8	5	1	5	4	7	-

	3	4	2156	7	8
3	-	1	6	1	5
4	1	-	7	5	1
2156	6	7	-	5	**4**
7	1	5	5	-	7
8	5	1	**4**	7	-

	3	4	7	21568
3	-	1	1	5
4	1	-	5	**1**
7	1	5	-	7
21568	5	**1**	7	-

	3	215684	7
3	-	**1**	1
215684	**1**	-	5
7	1	5	-

	2156843	7
2156843	-	**1**
7	**1**	-

Отже, вага циклу (2 1 5 6 8 4 3 7) дорівнює 16

3) Почнімо з вершини 3

	1	2	3	4	5	6	7	8
1	-	1	6	7	3	5	4	6
2	1	-	6	5	4	2	3	5
3	6	6	-	**1**	7	6	1	5
4	7	5	**1**	-	6	7	5	1
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	1	2	34	5	6	7	8
1	-	1	7	3	5	4	6
2	1	-	5	4	2	3	5
34	7	5	-	6	7	5	**1**
5	3	4	6	-	5	6	5
6	5	2	7	5	-	5	4
7	4	3	5	6	5	-	7
8	6	5	**1**	5	4	7	-

	1	2	5	6	7	348
1	-	1	3	5	4	6
2	1	-	4	2	3	5
5	3	4	-	5	6	5
6	5	2	5	-	5	**4**
7	4	3	6	5	-	7
348	6	5	5	**4**	7	-

	1	2	5	3486	7
1	-	1	3	5	4
2	1	-	4	**2**	3
5	3	4	-	5	6
3486	5	**2**	5	-	5
7	4	3	6	5	-

	1	34862	5	7
1	-	**1**	3	4
34862	**1**	-	4	3
5	3	4	-	6
7	4	3	6	-

	348621	5	7
348621	-	**3**	4
5	**3**	-	6
7	4	6	-

	3486215	7
3486215	-	**6**
7	**6**	-

Отже, вага циклу (3 4 8 6 2 1 5 7) дорівнює 18

4) Почнімо з вершини 4

	1	2	3	4	5	6	7	8
1	-	1	6	7	3	5	4	6
2	1	-	6	5	4	2	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	**1**
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	**1**	5	4	7	-

	1	2	3	5	6	7	48
1	-	1	6	3	5	4	6
2	1	-	6	4	2	3	5
3	6	6	-	7	6	1	5
5	3	4	7	-	5	6	5
6	5	2	6	5	-	5	**4**
7	4	3	1	6	5	-	7
48	6	5	5	5	**4**	7	-

	1	2	3	5	486	7
1	-	1	6	3	5	4
2	1	-	6	4	**2**	3
3	6	6	-	7	6	1
5	3	4	7	-	5	6
486	5	**2**	6	5	-	5
7	4	3	1	6	5	-

	1	4862	3	5	7
1	-	1	6	3	4
4862	**1**	-	6	4	3
3	6	6	-	7	1
5	3	4	7	-	6
7	4	3	1	6	-

	48621	3	5	7
48621	-	6	**3**	4
3	6	-	7	1
5	**3**	7	-	6
7	4	1	6	-

	486213	5	7
486213	-	7	**1**
5	7	-	6
7	**1**	6	-

	5	4862137
5	-	**6**
4862137	**6**	-

Отже, вага циклу (4 8 6 2 1 3 7 5) дорівнює 18

5) Почнімо з вершини 5

	1	2	3	4	5	6	7	8
1	-	1	6	7	**3**	5	4	6
2	1	-	6	5	4	2	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	1
5	**3**	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	51	2	3	4	6	7	8
51	-	**1**	6	7	5	4	6
2	**1**	-	6	5	2	3	5
3	6	6	-	1	6	1	5
4	7	5	1	-	7	5	1
6	5	2	6	7	-	5	4
7	4	3	1	5	5	-	7
8	6	5	5	1	4	7	-

	512	3	4	6	7	8
512	-	6	5	**2**	3	5
3	6	-	1	6	1	5
4	5	1	-	7	5	1
6	**2**	6	7	-	5	4
7	3	1	5	5	-	7
8	5	5	1	4	7	-

	3	4	5126	7	8
3	-	1	6	1	5
4	1	-	7	5	1
5126	6	7	-	5	**4**
7	1	5	5	-	7
8	5	1	**4**	7	-

	3	4	7	51268
3	-	1	1	5
4	1	-	5	1
7	1	5	-	7
51268	5	**1**	7	-

	3	512684	7
3	-	**1**	1
512684	**1**	-	5
7	1	5	-

	5126843	7
5126843	-	**1**
7	**1**	-

Отже, вага циклу (5 1 2 6 8 4 7) дорівнює 13

6) Почнімо з вершини 6

	1	2	3	4	5	6	7	8
1	-	1	6	7	3	5	4	6
2	1	-	6	5	4	**2**	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	1
5	3	4	7	6	-	5	6	5
6	5	**2**	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	1	62	3	4	5	7	8
1	-	**1**	6	7	3	4	6
62	**1**	-	6	5	4	3	5
3	6	6	-	1	7	1	5
4	7	5	1	-	6	5	1
5	3	4	7	6	-	6	5
7	4	3	1	5	6	-	7
8	6	5	5	1	5	7	-

	621	3	4	5	7	8
621	-	6	7	**3**	4	6
3	6	-	1	7	1	5
4	7	1	-	6	5	1
5	**3**	7	6	-	6	5
7	4	1	5	6	-	7
8	6	5	1	5	7	-

	3	4	6215	7	8
3	-	1	7	1	5
4	1	-	6	5	1
6215	7	6	-	6	**5**
7	1	5	6	-	7
8	5	1	**5**	7	-

	3	4	62157	8
3	-	1	**1**	5
4	1	-	5	1
62157	**1**	5	-	7
8	5	1	7	-

	621573	4	8
621573	-	**1**	5
4	**1**	-	1
8	5	1	-

	6215734	8
6215734	-	**1**
8	**1**	-

Отже, вага циклу (6 2 1 5 7 3 4 8) дорівнює 14

7) Почнімо з вершини 7

	1	2	3	4	5	6	7	8
1	-	1	6	7	3	5	4	6
2	1	-	6	5	4	2	3	5
3	6	6	-	1	7	6	**1**	5
4	7	5	1	-	6	7	5	1
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	**1**	5	6	5	-	7
8	6	5	5	1	5	4	7	-

	1	2	73	4	5	6	8
1	-	1	6	7	3	5	6
2	1	-	6	5	4	2	5
73	6	6	-	**1**	7	6	5
4	7	5	**1**	-	6	7	1
5	3	4	7	6	-	5	5
6	5	2	6	7	5	-	4
8	6	5	5	1	5	4	-

	1	2	734	5	6	8
1	-	1	7	3	5	6
2	1	-	5	4	2	5
734	7	5	-	6	7	**1**
5	3	4	6	-	5	5
6	5	2	7	5	-	4
8	6	5	**1**	5	4	-

	1	2	5	6	7348
1	-	1	3	5	6
2	1	-	4	2	5
5	3	4	-	5	5
6	5	2	5	-	**4**
7348	6	5	5	**4**	-

	1	2	5	73486
1	-	1	3	5
2	1	-	4	**2**
5	3	4	-	5
73486	5	**2**	5	-

	1	734862	5
1	-	**1**	3
734862	**1**	-	4
5	3	4	-

	7348621	5
7348621	-	**3**
5	**3**	-

Отже, вага циклу (7 3 4 8 6 2 1 5) дорівнює 13

8) Почнімо з вершини 8

	1	2	3	4	5	6	7	8
1	-	1	6	7	3	5	4	6
2	1	-	6	5	4	2	3	5
3	6	6	-	1	7	6	1	5
4	7	5	1	-	6	7	5	**1**
5	3	4	7	6	-	5	6	5
6	5	2	6	7	5	-	5	4
7	4	3	1	5	6	5	-	7
8	6	5	5	**1**	5	4	7	-

	1	2	3	84	5	6	7
1	-	1	6	7	3	5	4
2	1	-	6	5	4	2	3
3	6	6	-	**1**	7	6	1
84	7	5	**1**	-	6	7	5
5	3	4	7	6	-	5	6
6	5	2	6	7	5	-	5
7	4	3	1	5	6	5	-

	1	2	843	5	6	7
1	-	1	6	3	5	4
2	1	-	6	4	2	3
843	6	6	-	7	6	**1**
5	3	4	7	-	5	6
6	5	2	6	5	-	5
7	4	3	**1**	6	5	-

	1	2	5	6	8437
1	-	1	3	5	4
2	1	-	4	2	**3**
5	3	4	-	5	6
6	5	2	5	-	5
8437	4	**3**	6	5	-

	1	84372	5	6
1	-	**1**	3	5
84372	**1**	-	4	2
5	3	4	-	5
6	5	2	5	-

	843721	5	6
843721	-	**3**	5
5	**3**	-	5
6	5	5	-

	5	6
8437215	-	**5**
6	**5**	-

Отже, вага циклу (8 4 3 7 2 1 5) дорівнює 15

```cpp
#include <iostream>
#include <cstring>

using namespace std;

string hid(int a) {
  string one = " ", two = "V";
  int b;
  do {
    b = a % 10;
    switch (b) {
    case 0:
      one = "0" + one; break;
    case 1:
      one = "1" + one; break;
    case 2:
      one = "2" + one; break;
    case 3:
      one = "3" + one; break;
    case 4:
      one = "4" + one; break;
    case 5:
      one = "5" + one; break;
    case 6:
      one = "6" + one; break;
    case 7:
      one = "7" + one; break;
    case 8:
      one = "8" + one; break;
    case 9:
      one = "9" + one; break;
    default: cout << "error";
    }
    a /= 10;
  } while (a != 0);
  two += one;
```

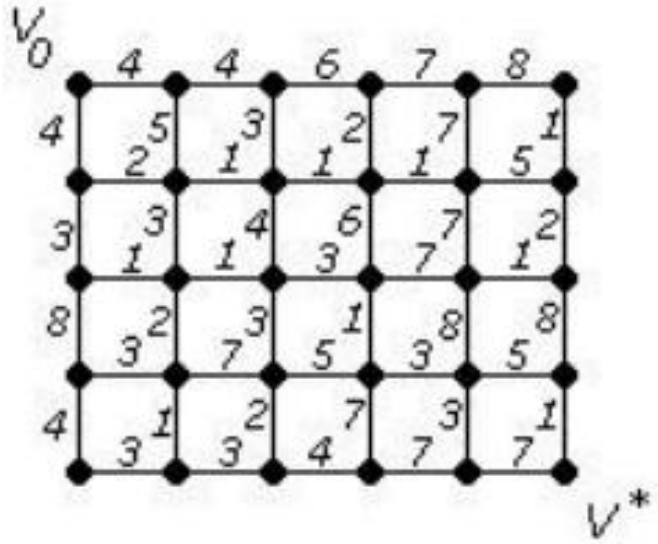
```
36      two += one;
37      return two;
38  }
39
40
41
42  int main()
43  {
44      int** A, n, a, ** B, * C;
45      string* vuvid;
46      cout << "Enter number of vertexes : ";
47      cin >> n;
48      cout << "Enter weight of edges :\n";
49      A = new int* [n];
50      B = new int* [n];
51      C = new int[n];
52      vuvid = new string[n];
53      for (int i = 0; i < n; i++) {
54          A[i] = new int[n];
55          B[i] = new int[n];
56          vuvid[i] = "From " + hid(i + 1) + " :\n\t" + hid(i + 1);
57      }
58      for (int i = 0; i < n; i++) {
59          for (int j = 0; j < n; j++) {
60              cin >> a;
61              A[i][j] = a;
62          }
63      }
64
65      int t = 0, minw = 0, minn = 0, W = 0;
66      for (int Q = 0; Q < n; Q++) {
67
68          for (int i = 0; i < n; i++) {
69              for (int j = 0; j < n; j++) {
70                  B[i][j] = A[i][j];
71              }
```

```
71        }
72      }
73
74      t = Q;
75      W = 0;
76      for (int L = 0; L < n-1; L++) {
77        minw = 0;
78        minn = 0;
79        for (int i = 0; i < n; i++) {
80          if (B[t][i] != 0) {
81            if (minw == 0) {
82              minw = B[t][i];
83              minn = i;
84            }
85            else if (B[t][i] < minw) {
86              minn = i;
87              minw = B[t][i];
88            }
89          }
90
91        }
92              W += minw;
93        for (int i = 0; i < n; i++) {
94          B[t][i] = 0;
95          B[i][t] = 0;
96        }
97        t = minn;
98        vuvid[Q] += hid(t + 1);
99
100     }
101
102     vuvid[Q] += "\n\tweight : ";
103     C[Q] = W;
104
105   }
106
107   cout << "\n\n";
108   for (int i = 0; i < n; i++) {
109     cout << vuvid[i] << C[i] << endl<<endl;
110   }
111
112   return 0;
113 }
```

```
Enter number of vertexes : 8
Enter weight of edges :
0 1 6 7 3 5 4 6
1 0 6 5 4 2 3 5
6 6 0 1 7 6 1 5
7 5 1 0 6 7 5 1
3 4 7 6 0 5 6 5
5 2 6 7 5 0 5 4
4 3 1 5 6 5 0 7
6 5 5 1 5 4 7 0


From V1  :
        V1 V2 V6 V8 V4 V3 V7 V5
        weight : 16

From V2  :
        V2 V1 V5 V6 V8 V4 V3 V7
        weight : 16

From V3  :
        V3 V4 V8 V6 V2 V1 V5 V7
        weight : 18

From V4  :
        V4 V3 V7 V2 V1 V5 V6 V8
        weight : 18

From V5  :
        V5 V1 V2 V6 V8 V4 V3 V7
        weight : 13

From V6  :
        V6 V2 V1 V5 V8 V4 V3 V7
        weight : 14

From V7  :
        V7 V3 V4 V8 V6 V2 V1 V5
        weight : 13

From V8  :
        V8 V4 V3 V7 V2 V1 V5 V6
        weight : 15
```

Завдання № 7

За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі між парою вершин V_0 і V^*.



l(v1) = 4

l(v2) = 4

l(v3) = 6

l(v4) = 7

l(v5) = 7

l(v6) = 8

l(v7) = 8

l(v8) = 8

l(v9) = 9

l(v10) = 9

l(v11) = 10

l(v12) = 10

l(v13) = 11

l(v14) = 12

l(v15) = 12

l(v16) = 13

l(v17) = 13

l(v18) = 14

l(v19) = 14

l(v20) = 14

l(v21) = 15

l(v22) = 16

l(v23) = 16

l(v24) = 17

l(v25) = 17

l(v26) = 18

l(v27) = 19

l(v28) = 21

l(v*) = 22

Найкоротша відстань від вершини V0 до V* дорівнює 22 і проходить через вершини:

v0 -> v2 -> v4 -> v8 -> v10 -> v14 -> v17 -> v22 -> v28 -> v*

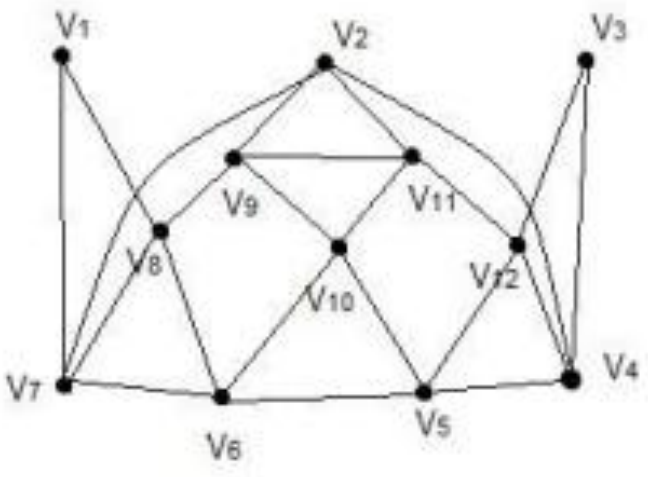
```cpp
#include<iostream>
using namespace std;
int n;
int i, j, q;
int dist[40];
bool visited[40];
int pred[40];
void createGraph(int c[40][40])
{
    int g1, g2;
    cout << "Enter the number of vertices: ";
    cin >> n;
    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++)
        {
            c[i][j] = 0;
        }
    }
    cout << "Enter size of (n*m) : ";
    cin >> g1 >> g2;

    for (i = 0; i < n; i++) {
        for (j = i + 1; j < n; j++)
        {
            if (j == i + 1 || j == i + g1) {
                cout << "Enter the length from x" << i+1 << " to x" << j+1 << ": ";
                cin >> c[i][j];
            }
            else {
                c[i][j] = 0;
            }
        }
    }
}
int minDistance()
{
    int minimum = 10000, minDist;
    for (int v = 0; v < n; v++)
        if (visited[v] == false && dist[v] <= minimum)
        {
            minimum = dist[v];
            minDist = v;
        }
    return minDist;
}
void printPath(int j)
{
    if (pred[j] == -1)
        return;
    printPath(pred[j]);
    cout << "X" << j+1 << " -> ";
}
void dijkstra(int c[40][40])
{
        int start;
        cout << "Enter the first node : ";
        cin >> start;
        for (int i = 0; i < n; i++)
        {
            pred[0] = -1;
            dist[i] = 10000;
```

```
63              visited[i] = false;
64          }
65          dist[start-1] = 0;
66          for (int count = 0; count < n - 1; count++)
67          {
68              int u = minDistance();
69              visited[u] = true;
70              for (int v = 0; v < n; v++)
71                  if (!visited[v] && c[u][v] &&
72                      dist[u] + c[u][v] < dist[v])
73                  {
74                      pred[v] = u;
75                      dist[v] = dist[u] + c[u][v];
76                  }
77          }
78          cout << "The least way is: ";
79          cout << dist[29] << endl;
80          cout << "The way is: ";
81          cout << "X1 -> ";
82              printPath(29);
83              cout << "The end!)" << endl;
84          //}
85      }
86  int main()
87  {
88      int c[40][40];
89      createGraph(c);
90      dijkstra(c);
91      return 0;
92  }
```

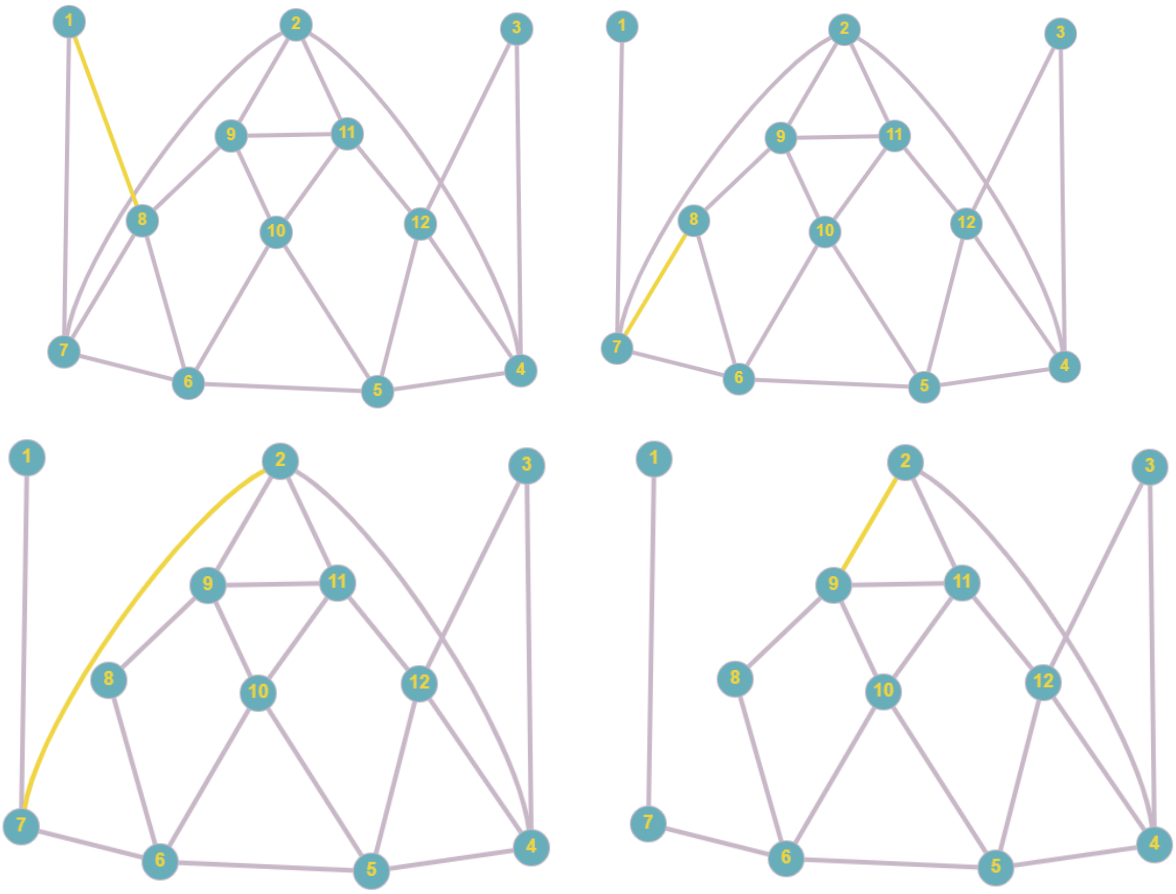
```
Enter the number of vertices: 30
Enter size of (n*m) : 6 5
Enter the length from x1 to x2: 4
Enter the length from x1 to x7: 4
Enter the length from x2 to x3: 4
Enter the length from x2 to x8: 5
Enter the length from x3 to x4: 6
Enter the length from x3 to x9: 3
Enter the length from x4 to x5: 7
Enter the length from x4 to x10: 2
Enter the length from x5 to x6: 8
Enter the length from x5 to x11: 7
Enter the length from x6 to x7: 0
Enter the length from x6 to x12: 1
Enter the length from x7 to x8: 2
Enter the length from x7 to x13: 3
Enter the length from x8 to x9: 1
Enter the length from x8 to x14: 3
Enter the length from x9 to x10: 1
Enter the length from x9 to x15: 4
Enter the length from x10 to x11: 1
Enter the length from x10 to x16: 6
Enter the length from x11 to x12: 5
Enter the length from x11 to x17: 7
Enter the length from x12 to x13: 0
Enter the length from x12 to x18: 2
Enter the length from x13 to x14: 1
Enter the length from x13 to x19: 8
Enter the length from x14 to x15: 1
Enter the length from x14 to x20: 2
Enter the length from x15 to x16: 3
Enter the length from x15 to x21: 3
Enter the length from x16 to x17: 7
Enter the length from x16 to x22: 1
Enter the length from x17 to x18: 1
Enter the length from x17 to x23: 8
Enter the length from x18 to x19: 0
Enter the length from x18 to x24: 8
Enter the length from x19 to x20: 3
Enter the length from x19 to x25: 4
Enter the length from x20 to x21: 7
Enter the length from x20 to x26: 1
Enter the length from x21 to x22: 5
Enter the length from x21 to x27: 2
Enter the length from x22 to x23: 3
Enter the length from x22 to x28: 7
Enter the length from x23 to x24: 5
Enter the length from x23 to x29: 3
Enter the length from x24 to x25: 0
Enter the length from x24 to x30: 1
Enter the length from x25 to x26: 3
Enter the length from x26 to x27: 3
Enter the length from x27 to x28: 4
Enter the length from x28 to x29: 7
Enter the length from x29 to x30: 7
Enter the length from x29 to x30: 7
Enter the first node : 1
The least way is: 22
The way is: X1 -> X7 -> X13 -> X14 -> X15 -> X16 -> X22 -> X23 -> X24 -> X30 ->
The end!)
```

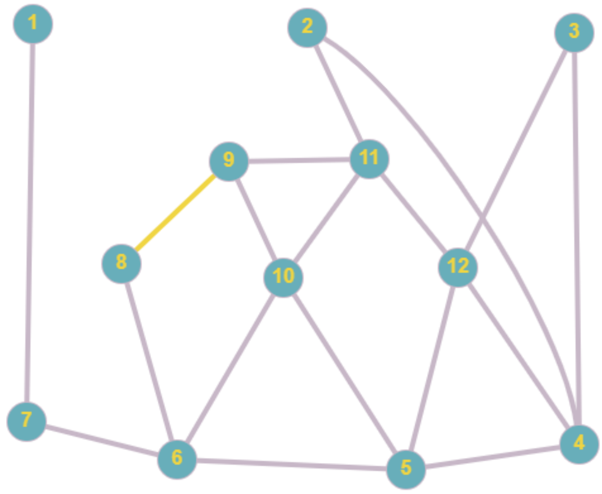
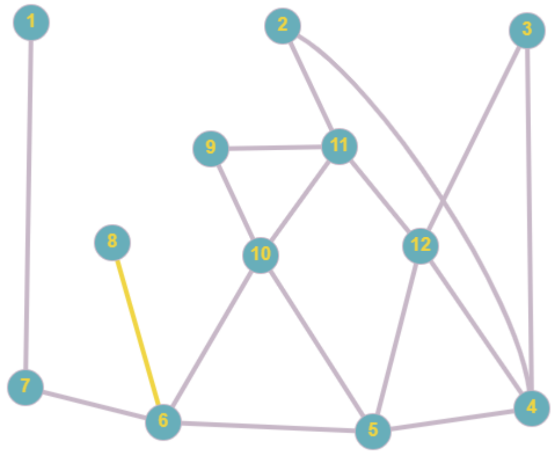
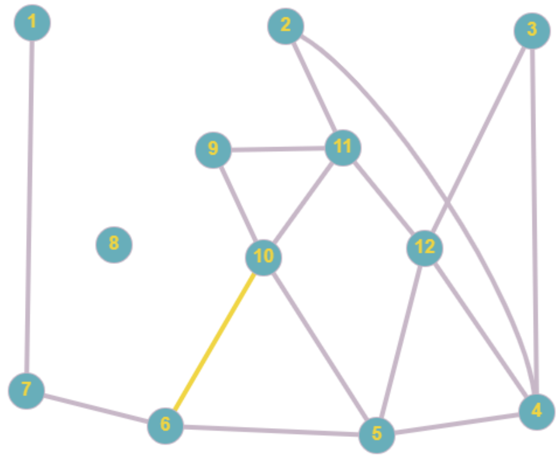
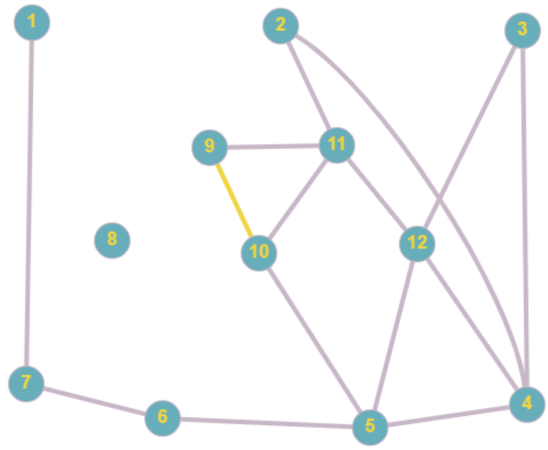
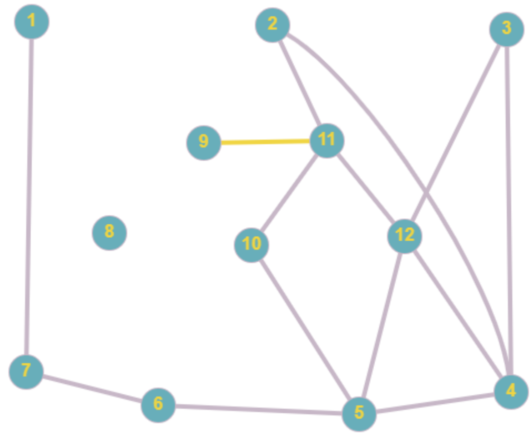
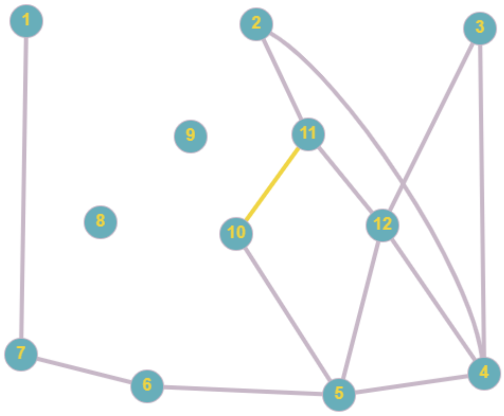
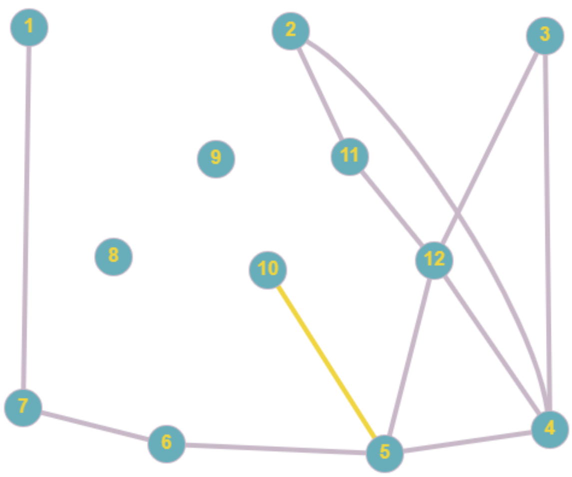
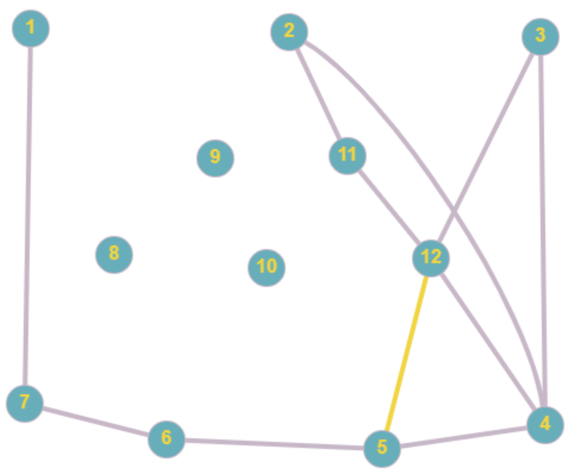
Завдання № 8

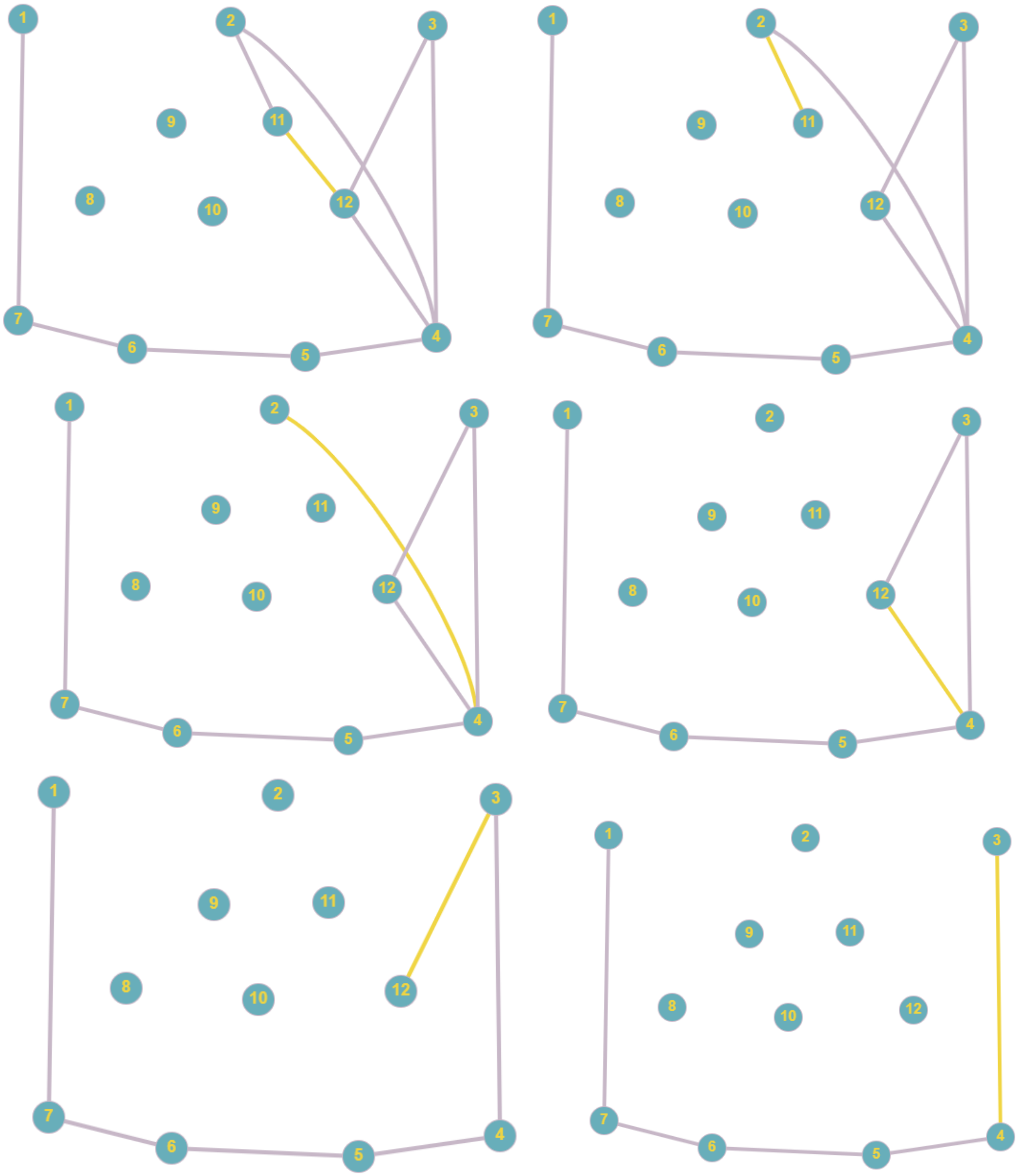
Знайти ейлеровий цикл в ейлеровому графі двома методами: а) Флері; б) елементарних циклів.

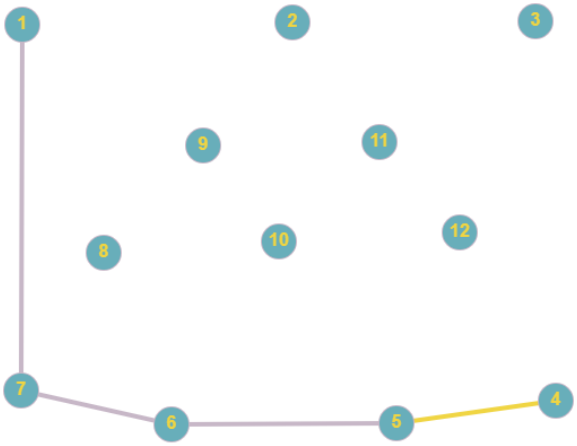


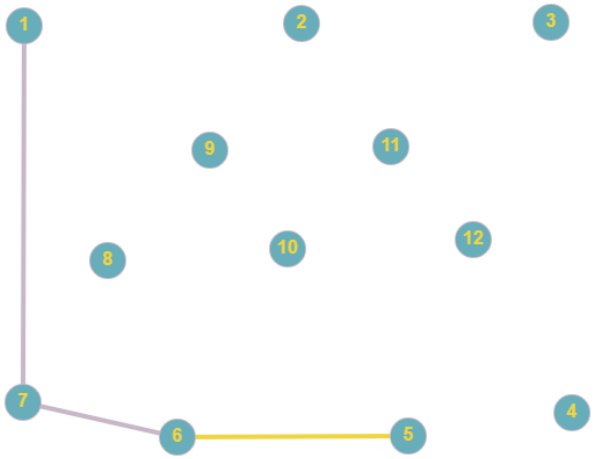
1) Алгоритм Флері

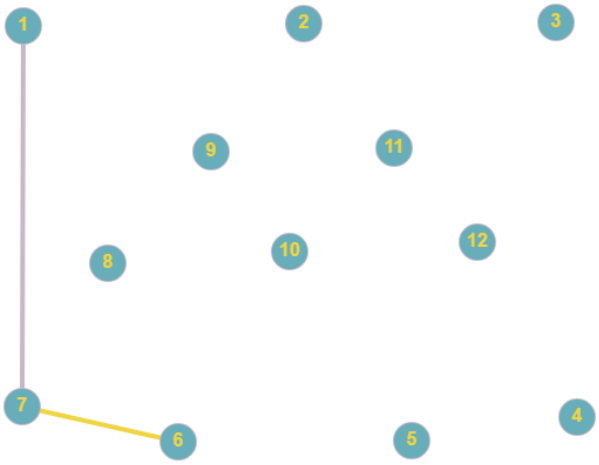


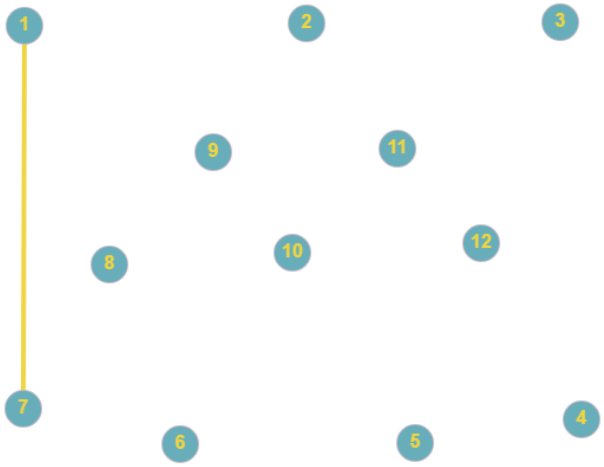

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12

1
2
3
4
5
6
7
8
9
10
11
12


1 2 3
9 11
8 10 12
7 6 5 4
1 2 3
9 11
8 10 12
7 6 5 4
1 2 3
9 11
8 10 12
7 6 5 4
1 2 3
9 11
8 10 12
7 6 5 4
1 2 3
9 11
8 10 12
7 6 5 4
1 2 3
9 11
8 10 12
7 6 5 4


1
2
3
9
11
8
10
12
7
6
5
4


1
2
3
9
11
8
10
12
7
6
5
4


1
2
3
9
11
8
10
12
7
6
5
4


1
2
3
9
11
8
10
12
7
6
5
4

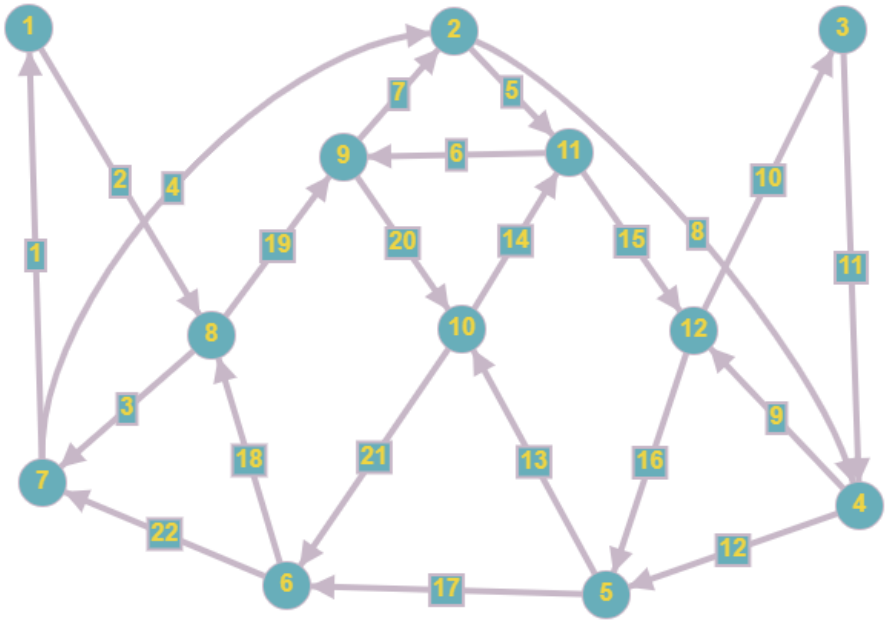
```cpp
#include <iostream>
#include <cstdio>

#define N 12
#define STACK_SIZE 100
using namespace std;

int G[N][N] {
{0,0,0,0,0,0,1,1,0,0,0,0},
{0,0,0,1,0,0,1,0,1,0,1,0},
{0,0,0,1,0,0,0,0,0,0,0,1},
{0,1,1,0,1,0,0,0,0,0,0,1},
{0,0,0,1,0,1,0,0,0,1,0,1},
{0,0,0,0,1,0,1,1,0,1,0,0},
{1,1,0,0,0,1,0,1,0,0,0,0},
{1,0,0,0,0,1,1,0,1,0,0,0},
{0,1,0,0,0,0,0,1,0,1,1,0},
{0,0,0,0,1,1,0,0,1,0,1,0},
{0,1,0,0,0,0,0,0,1,1,0,1},
{0,0,1,1,1,0,0,0,0,0,1,0},
};

int k;
int Stack[STACK_SIZE];

void Search(int v)
{
    int i;
    for(i = 0; i < N; i++)
        if(G[v][i])
        {
            G[v][i] = G[i][v] = 0;
            Search(i);
        }
    Stack[++k] = v+1;
}
```

```
36  }
37
38
39
40  int main()
41  {
42      int T, p, q, s;
43      int j, vv;
44
45      T = 1;
46      for(p = 0; p < N; p++)
47      {
48          s = 0;
49          for(q = 0; q < N; q++)
50          {
51              s += G[p][q];
52          }
53          if(s%2) T = 0;
54      }
55      k = -1;
56      cout << "Start vertex: ";
57      cin >> vv;
58      vv-=1;
59      if(T)
60      {
61          Search (vv);
62          for(j = 0; j <= k; j++)
63              cout << Stack[j] << " ";
64      }
65      else
66          cout << "it is not Eulerian graph\n";
67      return 0;
```

```
Start vertex: 7
7 8 9 11 10 9 2 11 12 4 3 12 5 10 6 7 2 4 5 6 8 1 7
Process returned 0 (0x0)   execution time : 1.777 s
Press any key to continue.
```

2) Алгоритм на основі циклів

Починаємо з вершини 7



Ось цикли, які утворюють наш ейлеровий цикл:

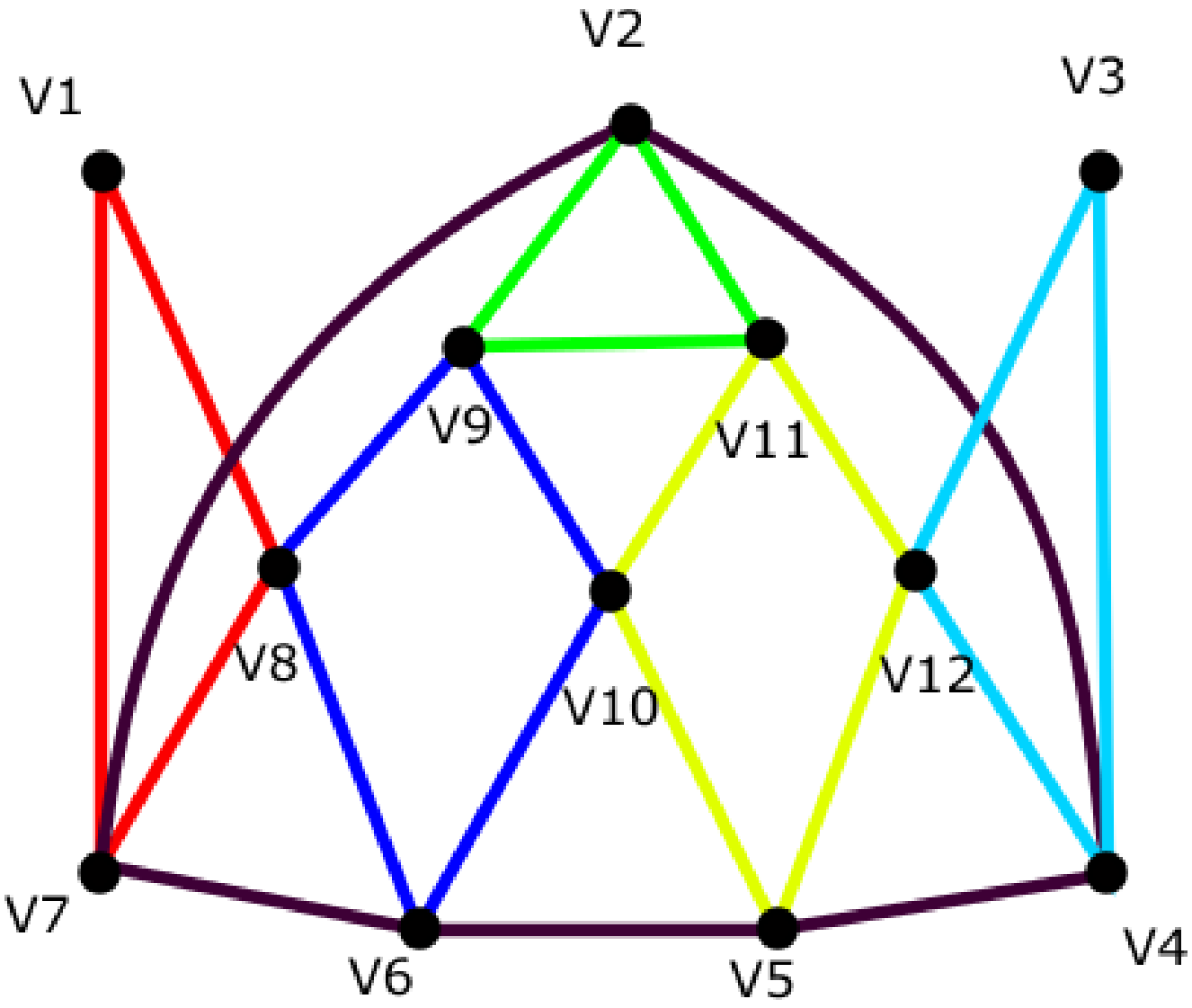
1) V7 - V2 - V4 - V5 - V6

2) V1 - V8 - V7

3) V6 - V8 - V9 - V10

4) V5 - V10 - V11 - V12

5) V3 - V4 - V12


V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12

```cpp
#include <iostream>
#include <iostream>
#include <vector>
using namespace std;
vector <int> graph;
int inf = 99;

bool check(vector<int> V, int vertix) {
  for (auto i = V.begin(); i != V.end(); i++) {
    if (*i == vertix) return false;
  }
  return true;
}
void Find(vector<int> * V, int** B, int n, int position, int f_vertix) {
  for (int i = position, k = 0; k < 1; i++, k++) {
    for (int j = 0; j < n; j++)
      if (B[i][j] == 1 && check((*V), j)) {
        if (j == f_vertix && (*V).size() > 2) {
          if (inf > V->size()) {
            graph.clear();
            graph.push_back(f_vertix + 1);
            for (auto it = (*V).begin(); it != (*V).end();
              it++)
              graph.push_back(*it + 1);
            graph.push_back(f_vertix + 1);
            inf = V->size();
            break;
          }
        }
        else {
          (*V).push_back(j);
          Find(V, B, n, j, f_vertix);
        }
      }
  }
  if (V->size() != 0)
    V->pop_back();
}
```

```
    V->pop_back();
  }
int main() {
  int n;
  cout << "Enter number of vertices: ";
  cin >> n;
  int** A = new int* [n];
  for (int i = 0; i < n; i++) {
    A[i] = new int[n];
  }
  for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
      cin >> A[i][j];
    }
  }
  vector<int> C;
  vector<int> B;
  cout << endl;
  int counted, p, j, sum;
  counted = 1;
  for (p = 0; p < n; p++)
  {
    sum = 0;
    for (j = 0; j < n; j++)
    {
      sum += A[p][j];
    }
    if (sum % 2) counted = 0;
  }
  cout << endl;
  cout << "So, the cycles are: " << endl;
  if (counted) {
    for (int j = 0; j < n; j++) {
      inf = 99;
      Find(&C, A, n, j, j);
      for (int i = 1; i <= graph.size(); i++) {
        cout << graph[i - 1] << " ";
      }
      cout << endl;
      graph.clear();
    }
  }
  else
    cout << "It is not correct \n";
  cout << endl;
  return 0;
}
```

```
Enter number of vertices: 12
0 0 0 0 0 0 1 1 0 0 0 0
0 0 0 1 0 0 1 0 1 0 1 0
0 0 0 1 0 0 0 0 0 0 0 1
0 1 1 0 1 0 0 0 0 0 0 1
0 0 0 1 0 1 0 0 0 1 0 1
0 0 0 0 1 0 1 1 0 1 0 0
1 1 0 0 0 1 0 1 0 0 0 0
1 0 0 0 0 1 1 0 1 0 0 0
0 1 0 0 0 0 0 1 0 1 1 0
0 0 0 0 1 1 0 0 1 0 1 0
0 1 0 0 0 0 0 0 1 1 0 1
0 0 1 1 1 0 0 0 0 0 1 0


So, the cycles are:
1 7 6 8 1
2 4 12 11 2
3 4 5 12 3
4 2 11 12 4
5 4 3 12 5
6 7 1 8 6
7 1 8 6 7
8 1 7 6 8
9 2 7 8 9
10 5 12 11 10
11 2 4 12 11
12 3 4 5 12
```

Завдання №9

Спростити формули (привести їх до скороченої ДНФ).

6. $(x \vee \bar{y} \to (z \to y \vee \bar{y} \vee x))$

$$\left(x \vee \bar{y} \to (z \to T \vee x)\right) = \left(x \vee \bar{y} \to (z \to T)\right) =$$

$$= (x \vee \bar{y} \to T) = \bar{x}y \vee T = T$$