МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Розрахункова робота

з дисципліни

«Дискретна математика»

Виконала:

студентка групи КН-112

Сидір Олена Юріївна

Викладач:

Мельникова Н.І.

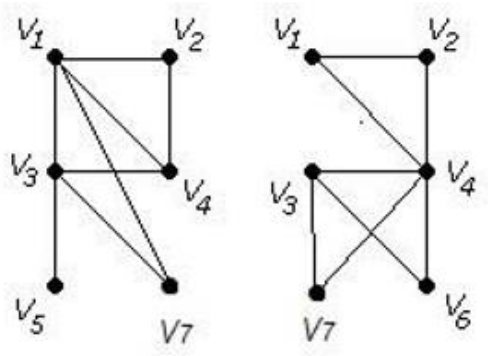
Львів – 2019 р.

Варіант № 21.

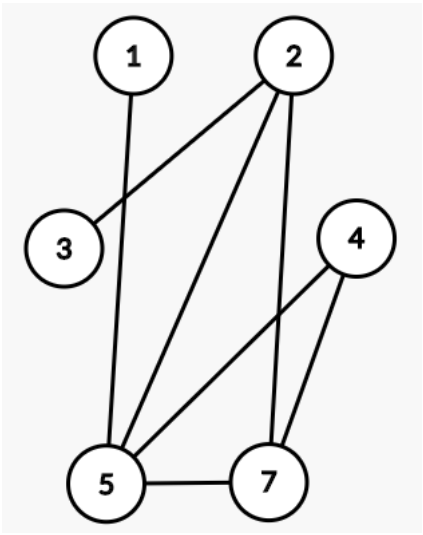
Завдання № 1

Виконати наступні операції над графами:

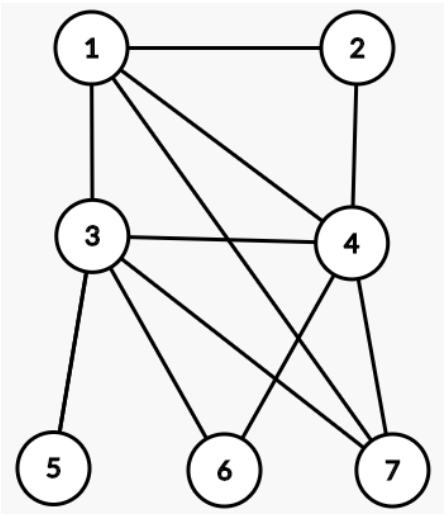
21)



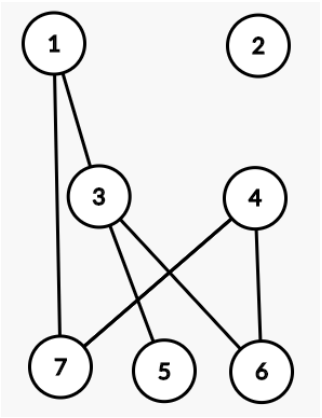
1) знайти доповнення до першого графу



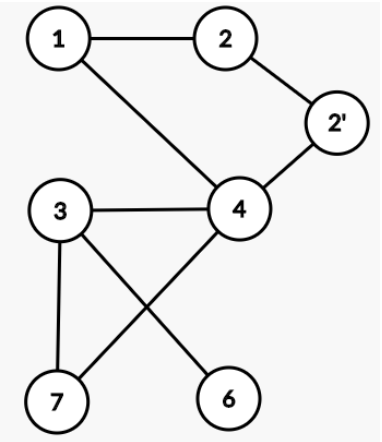
2) об’єднання графів



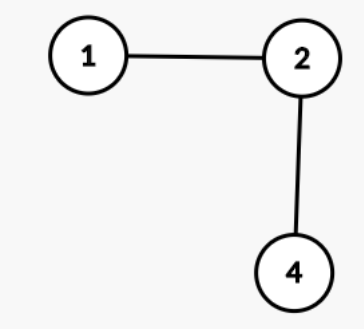
3) кільцеву суму G1 та G2 (G1+G2)



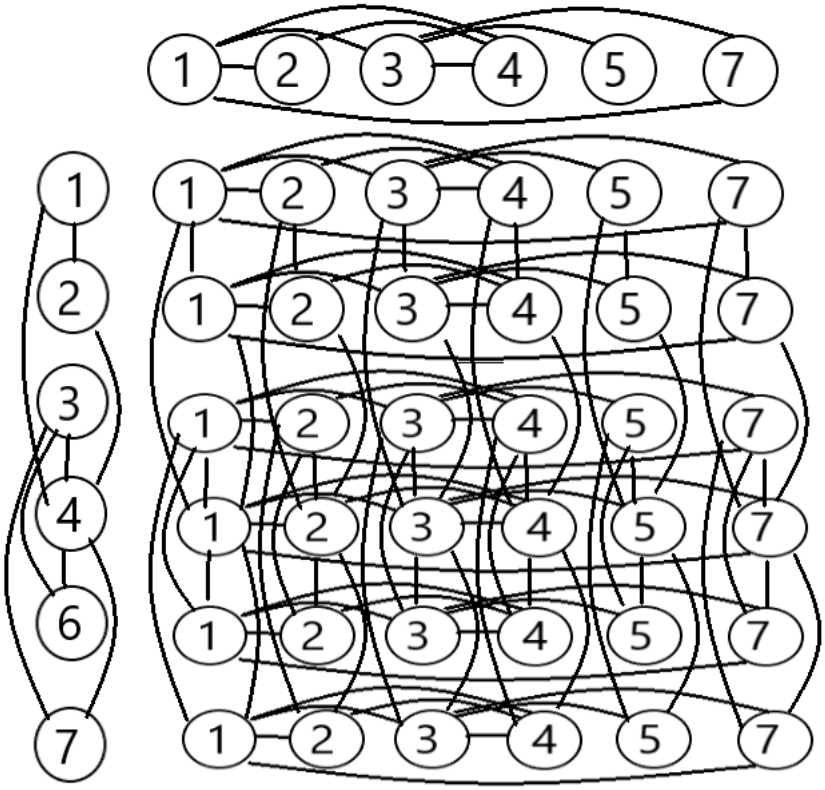
4) розмножити вершину у другому графі



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



6) добуток графів

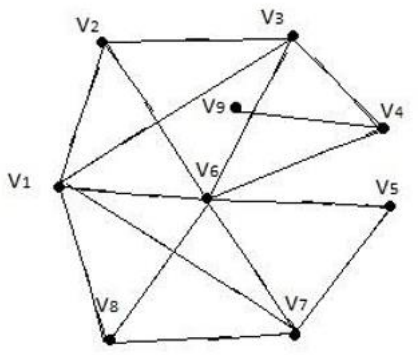


Завдання № 2

Скласти таблицю суміжності для неографа

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

21)



Завдання № 3

Для графа з другого завдання знайти діаметр.

Таблиця інцидентності

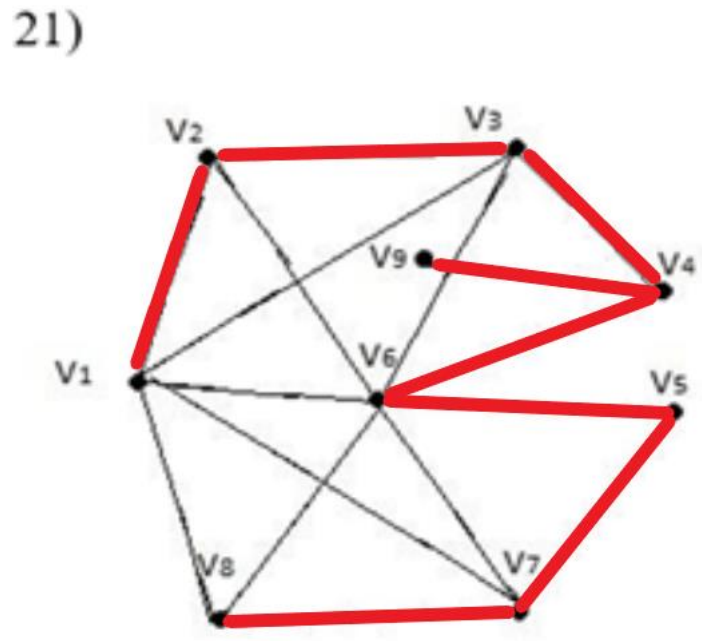
	1	2	3	4	5	6	7	8	9
1	0	1	1	2	2	1	1	1	3
2	1	0	1	2	2	1	2	2	3
3	1	1	0	1	2	1	2	2	2
4	2	2	1	0	2	1	2	2	1
5	2	2	2	2	0	1	1	2	3
6	1	1	1	1	1	0	1	1	2
7	1	2	2	2	1	1	0	1	3
8	1	2	2	2	2	1	1	0	3
9	3	3	2	1	3	2	3	3	0

Діаметр графа: 3.

Завдання № 4

Для графа з другого завдання виконати обхід дерева вглиб

1	1
2	1,2
3	1,2,3
4	1,2,3,4
5	1,2,3,4,9
6	1,2,3,4
7	1,2,3,4,6
8	1,2,3,4,6,5
9	1,2,3,4,6,5,7
10	1,2,3,4,6,5,7,8
11	1,2,3,4,6,5,7
12	1,2,3,4,6,5
13	1,2,3,4,6
14	1,2,3,4
15	1,2,3
16	1,2
17	1
18	



Програма:

```
#include <iostream>

using namespace std;
const int n=9;
int i, j;
bool *visited=new bool[n];
```

```
int graph[n][n] =
        {
                {0, 1, 1, 0, 0, 1, 1, 1, 0},
                {1, 0, 1, 0, 0, 1, 0, 0, 0},
                {1, 1, 0, 1, 0, 1, 0, 0, 0},
                {0, 0, 1, 0, 0, 1, 1, 0, 1},
                {0, 0, 0, 0, 0, 1, 1, 0, 0},
                {1, 1, 1, 1, 1, 0, 1, 1, 0},
                {1, 0, 0, 0, 1, 1, 0, 1, 0},
                {1, 0, 0, 0, 0, 1, 1, 0, 0},
                {0, 0, 0, 1, 0, 0, 0, 0, 0}
        };

void DFS(int st)
{
    int r;
    cout<<st+1<<" ";
    visited[st]=true;
    for (r=0; r<n; r++)
        if ((graph[st][r]!=0) && (!visited[r]))
            DFS(r);
}

int main()
{
    int start;
    cout<<"Матриця суміжності графа: "<<endl;
    for (i=0; i<n; i++)
    {
        visited[i]=false;
        for (j=0; j<n; j++)
            cout<<" "<<graph[i][j];
        cout<<endl;
    }
    cout<<"Стартова вершина: ";
    cin>>start;
    bool *vis=new bool[n];
    cout<<"Порядок обходу: ";
    DFS(start-1);
    delete []visited;
}
```

Результат:

```
Стартова вершина: 1
Порядок обходу: 1 2 3 4 6 5 7 8 9
```

Завдання № 5

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

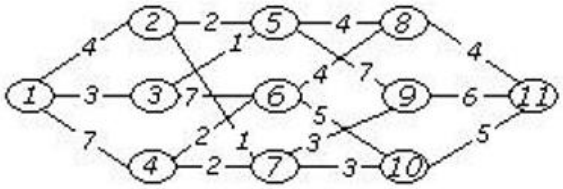
Краскала

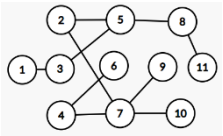
E	вага
3,5	1
2,7	1
2,5	2
4,6	2
4,7	2
1,3	3
7,9	3
7,10	3
5,8	4
8,11	4

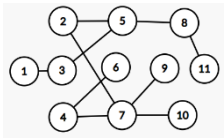
Прима

E	вага
1,3	3
3,5	1
5,2	2
2,7	1
7,4	2
4,6	2
7,9	3
7,10	3
5,8	4
8,11	4

21)







Програма для алгоритму Прима:

```
#include <bits/stdc++.h>
using namespace std;
int minKey(int key[], bool used[])
#define V 11


{

    int min = INT_MAX, min_index;

    for (int v = 0; v < V; v++)
        if (!used[v] && key[v] < min)
            min = key[v], min_index = v;

    return min_index;
}


void print(int parent[], int graph[V][V])
{
    cout<<"Edge \tWeight\n";
    for (int i = 1; i < V; i++)
        cout<<parent[i]+1<<" - "<<i+1<<" \t"<<graph[i][parent[i]]<<" \n";
}


void prims(int graph[V][V])
{
    int parent[V];
    int key[V];
    bool used[V];
    for (int i = 0; i < V; i++)
        key[i] = INT_MAX, used[i] = false;

    key[0] = 0;
```

```
    parent[0] = -1;

    for (int count = 0; count < V - 1; count++)
    {
        int u = minKey(key, used);
        used[u] = true;
        for (int v = 0; v < V; v++)
            if (graph[u][v] && !used[v] && graph[u][v] < key[v])
                parent[v] = u, key[v] = graph[u][v];
    }

    print(parent, graph);
}


int main()
{
    int graph[V][V] = {{0,4,3,7,0,0,0,0,0,0,0},
                       {4,0,0,0,2,0,1,0,0,0,0},
                       {3,0,0,0,1,7,0,0,0,0,0},
                       {7,0,0,0,0,2,2,0,0,0,0},
                       {0,2,1,0,0,0,0,4,7,0,0},
                       {0,0,7,2,0,0,0,4,0,5,0},
                       {0,1,0,2,0,0,0,0,3,3,0},
                       {0,0,0,0,4,4,0,0,0,0,4},
                       {0,0,0,0,7,0,3,0,0,0,6},
                       {0,0,0,0,0,5,3,0,0,0,5},
                       {0,0,0,0,0,0,0,4,6,5,0}
    };

    prims(graph);

    return 0;
}
```

Результат:

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

Програма для алгоритму Краскала:

```
#include <bits/stdc++.h>
using namespace std;

#define V 11
int parent[V];

int find(int i)
{
    while (parent[i] != i)
        i = parent[i];
    return i;
}
```

```
void union1(int i, int j)
{
    int a = find(i);
    int b = find(j);
    parent[a] = b;
}

void kruskal(int cost[][V])
{
    int mincost = 0;

    for (int i = 0; i < V; i++)
        parent[i] = i;

    int edge_count = 0;
    while (edge_count < V - 1) {
        int min = INT_MAX, a = -1, b = -1;
        for (int i = 0; i < V; i++) {
            for (int j = 0; j < V; j++) {
                if (find(i) != find(j) && cost[i][j] < min) {
                    min = cost[i][j];
                    a = i;
                    b = j;
                }
            }
        }

        union1(a, b);
        printf("Вершина %d:(%d, %d) вага:%d \n",
               edge_count+++1, a+1, b+1, min);
        mincost += min;
    }
    printf("\n Мінімальна вага= %d \n", mincost);
}

int main()
{
    int cost[V][V] = {

{INT_MAX,4,3,7,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX},

{4,INT_MAX,INT_MAX,INT_MAX,2,INT_MAX,1,INT_MAX,INT_MAX,INT_MAX,INT_MAX},

{3,INT_MAX,INT_MAX,INT_MAX,1,7,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX},

{7,INT_MAX,INT_MAX,INT_MAX,INT_MAX,2,2,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
            {INT_MAX,2,1,INT_MAX,INT_MAX,INT_MAX,INT_MAX,4,7,INT_MAX,INT_MAX},
            {INT_MAX,INT_MAX,7,2,INT_MAX,INT_MAX,INT_MAX,4,INT_MAX,5,INT_MAX},
            {INT_MAX,1,INT_MAX,2,INT_MAX,INT_MAX,INT_MAX,INT_MAX,3,3,INT_MAX},

{INT_MAX,INT_MAX,INT_MAX,INT_MAX,4,4,INT_MAX,INT_MAX,INT_MAX,INT_MAX,4},

{INT_MAX,INT_MAX,INT_MAX,INT_MAX,7,INT_MAX,3,INT_MAX,INT_MAX,INT_MAX,6},

{INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,5,3,INT_MAX,INT_MAX,INT_MAX,5},

{INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,4,6,5,INT_MAX}
    };

    kruskal(cost);

    return 0;
}
```

Результат:

```
Вершина 1:(2, 7) вага:1
Вершина 2:(3, 5) вага:1
Вершина 3:(2, 5) вага:2
Вершина 4:(4, 6) вага:2
Вершина 5:(4, 7) вага:2
Вершина 6:(1, 3) вага:3
Вершина 7:(7, 9) вага:3
Вершина 8:(7, 10) вага:3
Вершина 9:(5, 8) вага:4
Вершина 10:(8, 11) вага:4

 Мінімальна вага= 25
```

Завдання № 6

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

21)

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

1) (1,2)>(2,6)>(6,8)>(8,4)>(4,7)>(7,3)>(3,5)>(5,1)=1+1+2+1+1+4+4+1=18.
2) (1,5)>(5,3)>(3,8)>(8,4)>(4,6)>(6,2)>(2,7)>(7,1)=1+3+1+1+5+1+5+3=20.

Отже, раціональнішим буде шлях №1

Програма:

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

int counter =0, Miminal_way=9999;


bool check(vector<int> V, int Node) {
    for (auto i = V.begin(); i != V.end(); i++)
        if (*i == Node)return false;
    return true;
}

int minimum(vector<int>* V, int** arr, int n,int i) {
    int min = 999;
    for (int j = 0; j < n; j++) {
        if (arr[i][j] < min && arr[i][j] != 0 && check((*V), j))min = arr[i][j];
```

```
    }
    return min;
}



void vhlub(vector<int>* V, int** arr, int n, int pos, vector<int> * Vcon) {
    int min;
    for (int i = pos, k = 0; k < 1; i++, k++) {
        min = minimum(V, arr, n, i);
        for (int j = 0; j < n; j++)
            if (arr[i][j] == min && check((*V), j)) {
                (*V).push_back(j);
                vhlub(V, arr, n, j,Vcon);
            }
        if (V->size() == n) {
            (*V).push_back((*V)[0]);
            counter = 0;
            for (int l = 1; l <= n; l++)
                counter += arr[(*V)[l - 1]][(*V)[l]];
            for (auto it = (*V).begin(); it != (*V).end(); it++)
                cout << *it + 1 << " ";
            cout <<"Довжина: "<<counter<< endl;
            if (Miminal_way == counter) {
                for (int op = 0; op <= n; op++)
                    (*Vcon).push_back((*V)[op]);
                (*Vcon).push_back(counter);
            }
            else if (Miminal_way > counter) {
                (*Vcon).clear();
                for (int op = 0; op <= n; op++)
                    (*Vcon).push_back((*V)[op]);
                (*Vcon).push_back(counter);
                Miminal_way = counter;
            }
            V->pop_back();

        }
    }
    V->pop_back();

}

int main()
{
    int n;
    cout << "Введіть кількість вершин: ";
    cin >> n;
    int** arr = new  int* [n];
    for (int i = 0; i < n; i++)
        arr[i] = new int[n];

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            cin >> arr[i][j];
    vector<int> V;
    vector<int> Vcon;

    cout << endl;
    for (int i = 0; i < n; i++) {
        V.clear();
        V.push_back(i);
        vhlub(&V, arr, n, i, &Vcon);
    }

    cout << endl << endl << endl << "Найкоротші шляхи:" << endl << endl;
```

```
    for (int i = 1; i <= Vcon.size(); i++) {

        if (i != 0 && i%(n+2) == 0 )
            cout << "Довжина: " << Vcon[i-1] << endl;
        else
            cout << Vcon[i-1] + 1 << " ";
    }
}

/*

 0 1 3 5 1 5 3 2
 1 0 6 6 6 1 5 5
 3 6 0 7 3 5 4 1
 5 6 7 0 5 5 5 1
 1 6 3 5 0 6 6 6
 5 1 5 5 6 0 5 2
 3 5 4 5 6 5 0 2
 2 5 1 1 6 2 2 0
 */
```

Результат:

Введіть кількість вершин: 8

0 1 3 5 1 5 3 2

1 0 6 6 6 1 5 5

3 6 0 7 3 5 4 1

5 6 7 0 5 5 5 1

1 6 3 5 0 6 6 6

5 1 5 5 6 0 5 2

3 5 4 5 6 5 0 2

2 5 1 1 6 2 2 0

1 2 6 8 3 5 4 7 1 Довжина: 21

1 2 6 8 4 5 3 7 1 Довжина: 20

1 2 6 8 4 7 3 5 1 Довжина: 18

1 5 3 8 4 6 2 7 1 Довжина: 20

1 5 3 8 4 7 2 6 1 Довжина: 22

1 5 3 8 4 7 6 2 1 Довжина: 18

2 1 5 3 8 4 6 7 2 Довжина: 22

2 1 5 3 8 4 7 6 2 Довжина: 18

2 6 8 3 1 5 4 7 2 Довжина: 23

2 6 8 3 5 1 7 4 2 Довжина: 22

2 6 8 4 1 5 3 7 2 Довжина: 22

2 6 8 4 5 1 3 7 2 Довжина: 22

2 6 8 4 5 1 7 3 2 Довжина: 23

2 6 8 4 7 1 5 3 2 Довжина: 22

3 8 4 1 2 6 7 5 3 Довжина: 23

3 8 4 1 5 2 6 7 3 Довжина: 24

3 8 4 1 5 6 2 7 3 Довжина: 24

3 8 4 1 5 7 2 6 3 Довжина: 25

3 8 4 1 5 7 6 2 3 Довжина: 26

3 8 4 5 1 2 6 7 3 Довжина: 19

3 8 4 6 2 1 5 7 3 Довжина: 20

3 8 4 7 1 2 6 5 3 Довжина: 21

3 8 4 7 1 5 2 6 3 Довжина: 23

3 8 4 7 1 5 6 2 3 Довжина: 24

4 8 3 1 2 6 7 5 4 Довжина: 23

4 8 3 1 5 2 6 7 4 Довжина: 23

4 8 3 1 5 6 2 7 4 Довжина: 23

4 8 3 1 5 7 2 6 4 Довжина: 23

4 8 3 1 5 7 6 2 4 Довжина: 24

4 8 3 5 1 2 6 7 4 Довжина: 18

5 1 2 6 8 3 7 4 5 Довжина: 20

5 1 2 6 8 4 7 3 5 Довжина: 18

6 2 1 5 3 8 4 7 6 Довжина: 18

7 8 3 1 2 6 4 5 7 Довжина: 24

7 8 3 1 5 4 6 2 7 Довжина: 23

7 8 3 5 1 2 6 4 7 Довжина: 19

7 8 4 1 2 6 3 5 7 Довжина: 24

7 8 4 1 5 3 6 2 7 Довжина: 23

7 8 4 5 1 2 6 3 7 Довжина: 20

7 8 4 6 2 1 5 3 7 Довжина: 18

8 3 1 2 6 4 5 7 8 Довжина: 24

8 3 1 2 6 4 7 5 8 Довжина: 28

8 3 1 2 6 7 4 5 8 Довжина: 27

8 3 1 5 4 6 2 7 8 Довжина: 23

8 3 1 5 4 7 2 6 8 Довжина: 23

8 3 1 5 4 7 6 2 8 Довжина: 26

8 3 5 1 2 6 4 7 8 Довжина: 19

8 3 5 1 2 6 7 4 8 Довжина: 18

8 4 1 2 6 3 5 7 8 Довжина: 24

8 4 1 2 6 7 3 5 8 Довжина: 26

8 4 1 5 3 7 2 6 8 Довжина: 22

8 4 1 5 3 7 6 2 8 Довжина: 25

8 4 5 1 2 6 3 7 8 Довжина: 20

8 4 5 1 2 6 7 3 8 Довжина: 19

8 4 6 2 1 5 3 7 8 Довжина: 18

8 4 7 1 2 6 3 5 8 Довжина: 25

8 4 7 1 5 3 6 2 8 Довжина: 24

Найкоротші шляхи:

1 2 6 8 4 7 3 5 1 Довжина: 18

1 5 3 8 4 7 6 2 1 Довжина: 18

2 1 5 3 8 4 7 6 2 Довжина: 18

4 8 3 5 1 2 6 7 4 Довжина: 18

5 1 2 6 8 4 7 3 5 Довжина: 18

6 2 1 5 3 8 4 7 6 Довжина: 18

7 8 4 6 2 1 5 3 7 Довжина: 18

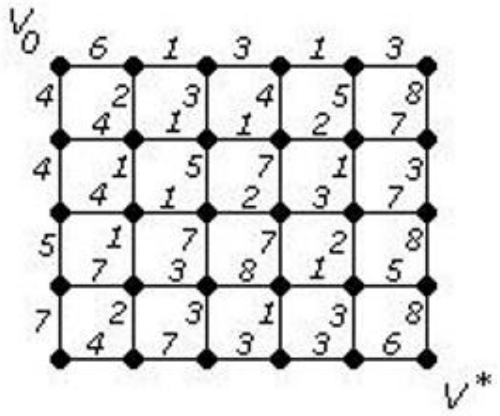
8 3 5 1 2 6 7 4 8 Довжина: 18

8 4 6 2 1 5 3 7 8 Довжина: 18

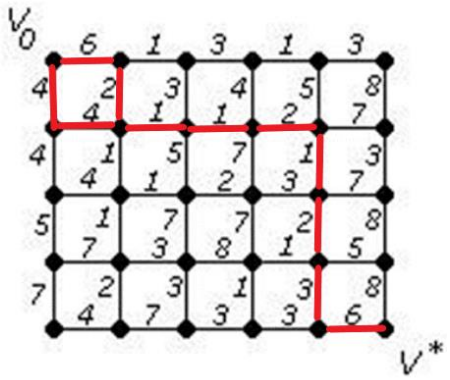
Завдання № 7

За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі між парою вершин V0 і V* .

21)



21)



Програма:

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#define SIZE 30

#pragma warning(disable : 4996);
using namespace std;
int main()
{
    int a[SIZE][SIZE];
    int d[SIZE];
    int v[SIZE];
    int t, minindex, min;
    int first = 0;
    cout << "enter matrix :" << endl;
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            cin >> a[i][j];
        }
    }
    for (int i = 0; i < SIZE; i++)
    {
        d[i] = 10000;
        v[i] = 1;
    }
    d[first] = 0;
    do {
        minindex = 10000;
        min = 10000;
        for (int i = 0; i < SIZE; i++)
        {
            if ((v[i] == 1) && (d[i] < min))
            {
                min = d[i];
                minindex = i;
            }
```

```
        }
        if (minindex != 10000)
        {
            for (int i = 0; i < SIZE; i++)
            {
                if (a[minindex][i] > 0)
                {
                    t = min + a[minindex][i];
                    if (t < d[i])
                    {
                        d[i] = t;
                    }
                }
            }
            v[minindex] = 0;
        }
    } while (minindex < 10000);
    cout << "the shortest distance to the vertexes: " << endl;
    for (int i = 0; i < SIZE; i++)
        cout << d[i] << "  ";
    int ver[SIZE];
    int end = 29;
    ver[0] = end + 1;
    int k = 1;
    int weight = d[end];

    while (end != first)
    {
        for (int i = 0; i < SIZE; i++)
            if (a[end][i] != 0)
            {
                int t = weight - a[end][i];
                if (t == d[i])
                {
                    weight = t;
                    end = i;
                    ver[k] = i + 1;
                    k++;
                }
            }
    }
    cout << endl;
    cout << "the shortest way:" << endl;
    for (int i = k - 1; i >= 0; i--)
        cout<< " -> " << "V" << ver[i];
    return 0;
}
/*

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

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

0 0 0 0 0 0 4 0 0 0 0 0 0 4 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 4 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 5 0 0 0 0 1 0 2 0 0 0 0 7 0 0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0 7 0 0 0 0 2 0 3 0 0 0 0 7 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 3 0 7 0 0 0 0 2 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 7 0 0 0 0 0 0 8 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 7 0 0 0 0 7 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 7 0 3 0 0 0 0 2 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 0 0 0 0 3 0 8 0 0 0 0 3 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 0 0 0 0 8 0 1 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 0 5 0 0 0 0 3 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 0 0 0 0 5 0 0 0 0 0 0 8

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

*/
```

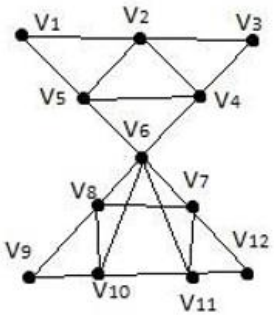
Результат:

```
the shortest distance to the vertexes:
0  6  7  10  11  14  4  8  9  10  12  19  8  9  10  12  13  20  12  10  13  16  15  20  16  12  16  17  18  24
the shortest way:
 -> V1 -> V2 -> V8 -> V9 -> V10 -> V11 -> V17 -> V23 -> V29 -> V30
```

Завдання № 8

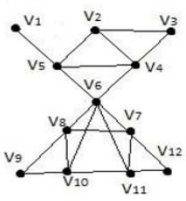
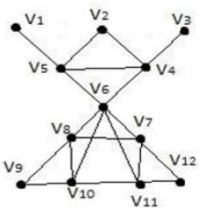
Знайти ейлеровий цикл в ейлеровому графі двома методами: а) Флері; б) елементарних циклів.

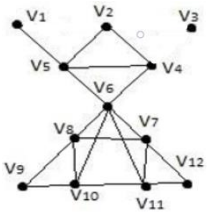
21)

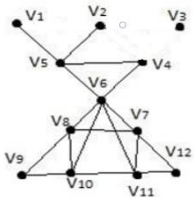


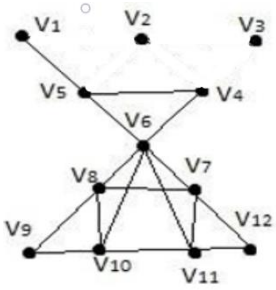
1) Флері

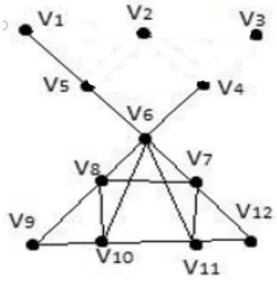
Оскільки кожна вершина має парний степінь, можна стверджувати, шо в цьому графі існує ейлеровий цикл.

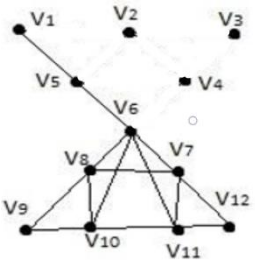
Цикл: V1,V2  V1,V2,V3 

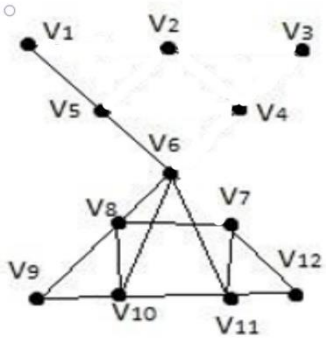
V1,V2,V3,V4 

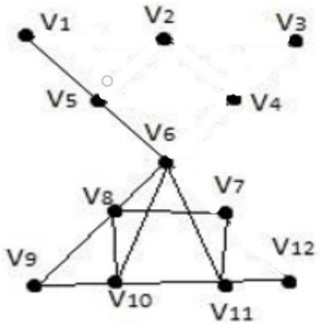
V1,V2,V3,V4,V2 

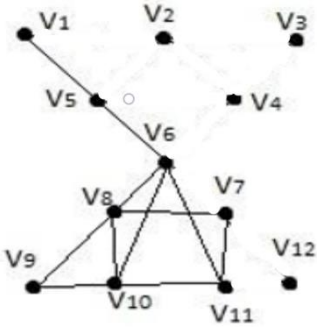
V1,V2,V3,V4.V2,V5 

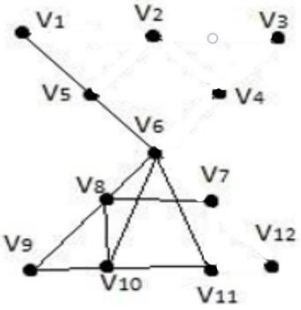
V1,V2,V3,V4,V2,V5, V4 

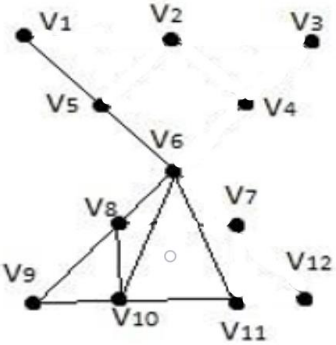
V1,V2,V3,V4,V2,V5,V4,V6 

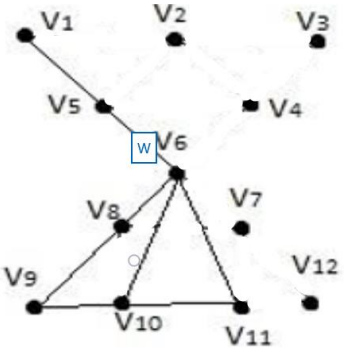
V1,V2,V3,V4.V2,V5,V6,V7 

V1,V2,V3,V4.V2,V5,V6,V7,V12 

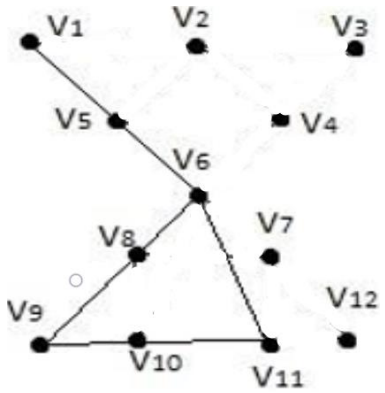
V1,V2,V3,V4.V2,V5,V6,V7,V12,V11 

V1,V2,V3,V4.V2,V5,V6,V7,V12,V11,V7 

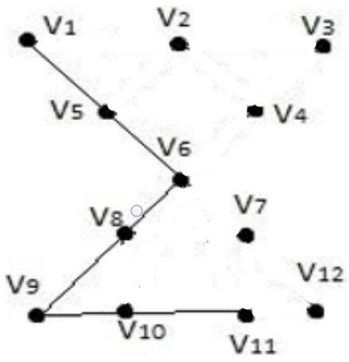
V1,V2,V3,V4.V2,V5,V6,V7,V12,V11,V7,V8 

V1,V2,V3,V4.V2,V5,V6,V7,V12,V11,V7,V8,V10 

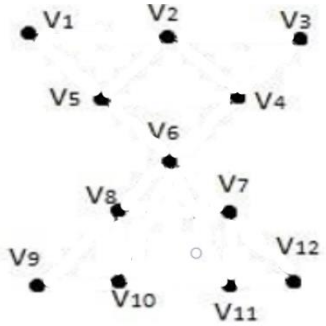
V1,V2,V3,V4.V2,V5,V6,V7,V12,V11,V7,V8,V10,V6



V1,V2,V3,V4.V2,V5,V6,V7,V12,V11,V7,V8,V10,V6,V11



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



Програма:

```
#include<iostream>
#include<vector>
#define NODE 12
using namespace std;
int graph[NODE][NODE] = {{0,1,0,0,1,0,0,0,0,0,0,0},
                         {1,0,1,1,1,0,0,0,0,0,0,0},
                         {0,1,0,1,0,0,0,0,0,0,0,0},
                         {0,1,1,0,1,1,0,0,0,0,0,0},
                         {1,1,0,1,0,1,0,0,0,0,0,0},
                         {0,0,0,1,1,0,1,1,0,1,1,0},
                         {0,0,0,0,0,1,0,1,0,0,1,1},
                         {0,0,0,0,0,1,1,0,1,1,0,0},
                         {0,0,0,0,0,0,0,1,0,1,0,0},
                         {0,0,0,0,0,1,0,1,1,0,1,0},
                         {0,0,0,0,0,1,1,0,0,1,0,1},
                         {0,0,0,0,0,0,1,0,0,0,1,0}
};
int tempGraph[NODE][NODE];
int findStartVert(){
   for(int i = 0; i<NODE; i++){
```

```
      int deg = 0;
      for(int j = 0; j<NODE; j++){
         if(tempGraph[i][j])
         deg++;
      }
      if(deg % 2 != 0)
      return i;
   }
   return 0;
}
bool Bridge(int u, int v){
   int deg = 0;
   for(int i = 0; i<NODE; i++)
      if(tempGraph[v][i])
         deg++;
      if(deg>1){
         return false;
      }
   return true;
}
int edgeCount(){
   int count = 0;
   for(int i = 0; i<NODE; i++)
      for(int j = i; j<NODE; j++)
         if(tempGraph[i][j])
            count++;
   return count;
}
void fleury(int start){
   static int edge = edgeCount();
   for(int v = 0; v<NODE; v++){
      if(tempGraph[start][v]){
         if(edge <= 1 || !Bridge(start, v)){
            cout << start+1 << "->" << v+1 << " ";
            tempGraph[start][v] = tempGraph[v][start] = 0;
            edge--;
            fleury(v);
         }
      }
   }
}
int main(){
   for(int i = 0; i<NODE; i++) //copy main graph to tempGraph
   for(int j = 0; j<NODE; j++)
   tempGraph[i][j] = graph[i][j];
   cout << "Ейлерів шлях: ";
   fleury(findStartVert());
}
```

Результат:

```
Ейлерів шлях: 1->2 2->3 3->4 4->2 2->5 5->4 4->6 6->5 6->7 7->8 8->6 6->10 10->8 8->9 9->10 10->11 11->7 7->12 12->11
```

2)Елементарні цикли C1=v1,v2,v5,v1

C2=v2,v3,v4,v2

C`1=v1,v2,v3,v4,v2,v5,v1

C3=v5,v4,v6,v5

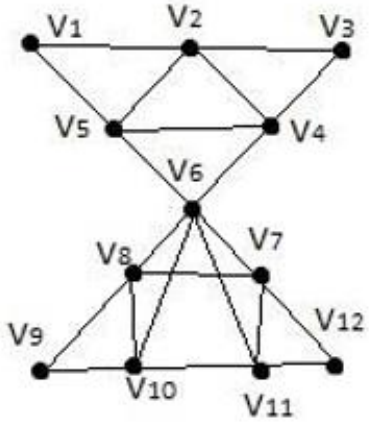
C``1=v1,v2,v3,v4,v2,v5,v4,v6,v5,v1

C4=v6,v8,v9,v10,v11,v12,v7,v6

C5=v10,v8,v7,v11,v10

C6=v11,v6,v10,v11

21)



C=v1,v2,v3,v4,v2,v5,v4,v6,v8,v9,v10,v8,v7,v11,v10,v11,v6,v10,v11,v12,v7,v6,v5,v1

Завдання №9

Спростити формули (привести їх до скороченої ДНФ).

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

$X\bar{X} \vee XY \vee \bar{Y}\bar{X} \vee \bar{Y}Y \vee \bar{Z}\bar{X} \vee \bar{Z}Y$

$0 \vee XY \vee \bar{Y}\bar{X} \vee \bar{Y}Y \vee \bar{Z}\bar{X} \vee \bar{Z}Y$

$XY \vee \bar{Y}\bar{X} \vee \bar{Y}Y \vee \bar{Z}\bar{X} \vee \bar{Z}Y$

$XY \vee \bar{Y}\bar{X} \vee 0 \vee \bar{Z}\bar{X} \vee \bar{Z}Y$

$XY \vee \bar{Y}\bar{X} \vee \bar{Z}\bar{X} \vee \bar{Z}Y$

$\bar{X}\bar{Y} \vee XY \vee Y\bar{Z} \vee \bar{Z}Y$

$\bar{X}\bar{Y} \vee XY \vee Y\bar{Z}$