МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Розрахункова робота

з дисципліни

«Дискретна математика»

Виконав:

студент групи КН-112

Садовнік Ілля

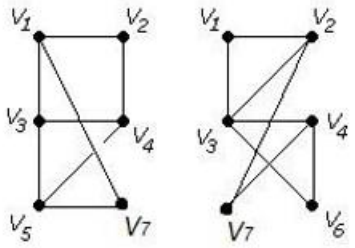
Викладач:

Мельникова Н.І.

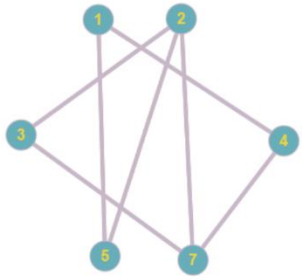
Завдання № 1

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.

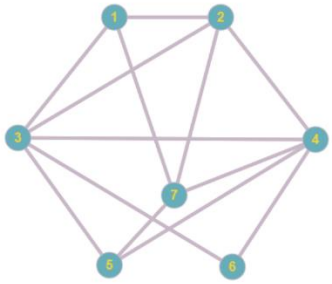
10)



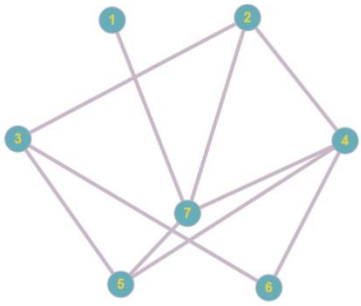
1)) знайти доповнення до першого графу



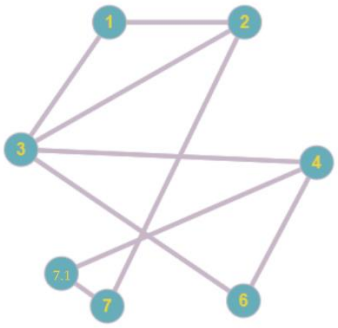
2) об’єднання графів,



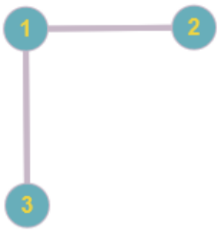
3) кільцеву сумму G1 та G2 (G1+G2),



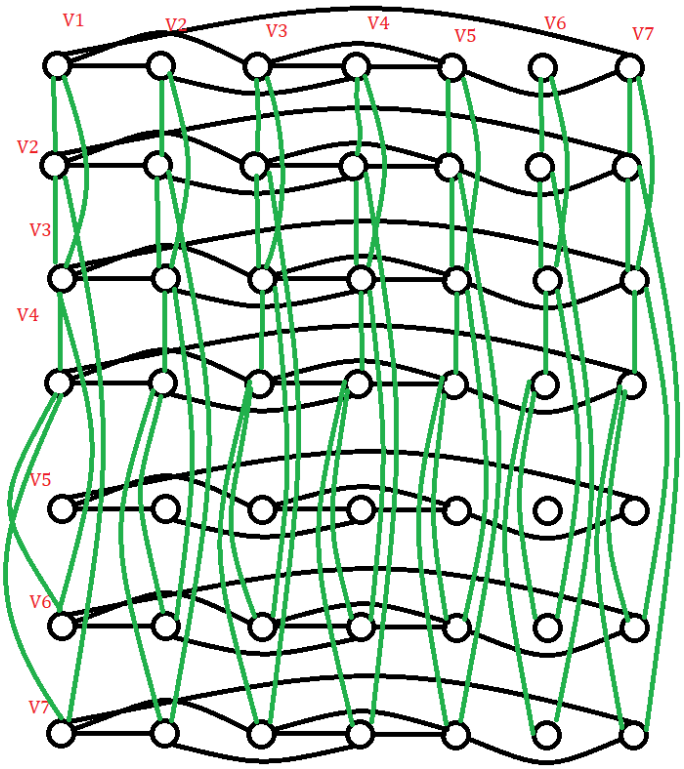
4) розмножити вершину у другому графі,



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



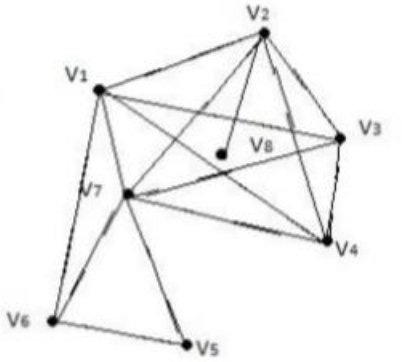
6) добуток графів.



Завдання № 2

Скласти таблицю суміжності для орграфа.

10)



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

Завдання № 3

Для графа з другого завдання знайти діаметр.

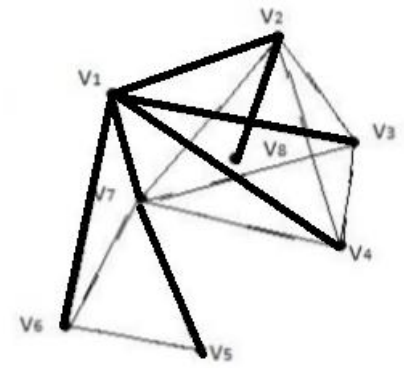
Діаметр = 3.

Завдання № 4

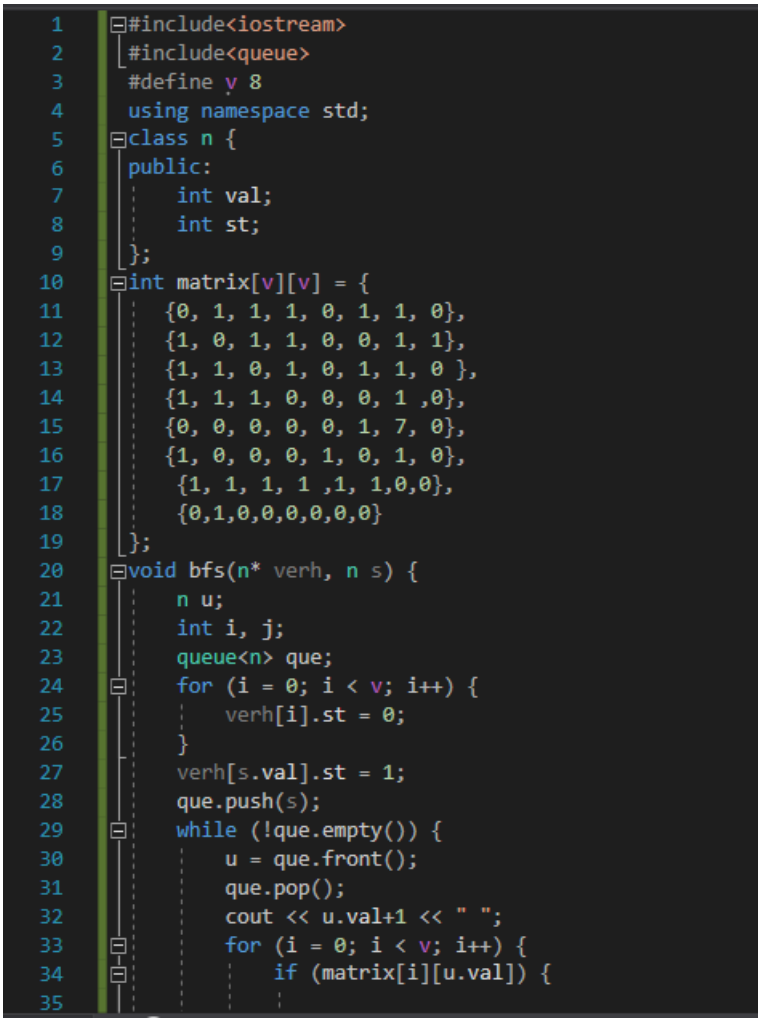
Для графа з другого завдання виконати обхід дерева вглиб (варіант закінчується на непарне число) або вшир (закінчується на парне число).

Пошук вшир

10)

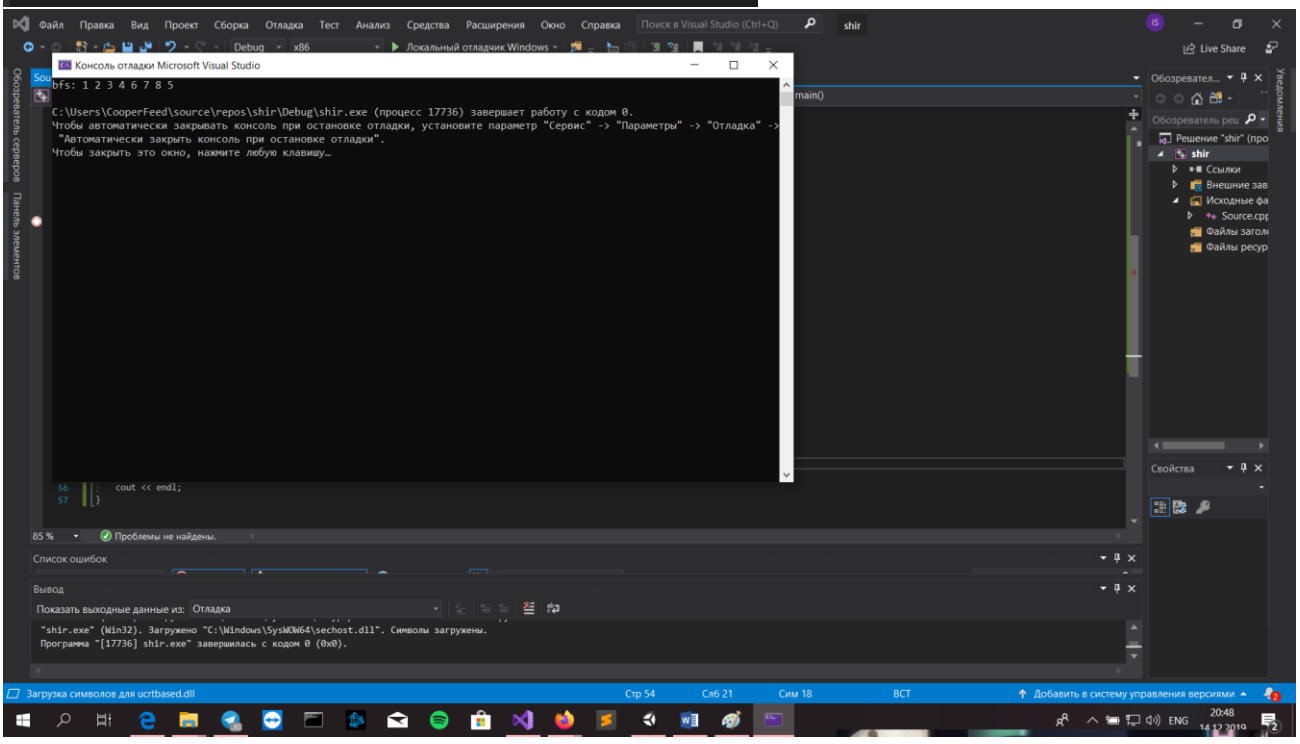


V1	V1
V2	V1V2
V3	V1V2V3
V4	V1V2V3V4
V6	V1V2V3V4V6
V7	V1V2V3V4V6V7
	V2V3V4V7V6
V8	V2V3V4V7V6V8
	V3V4V7V6V8
	V4V7V6V8
	V7V6V8
V5	V7V6V8V5
	V6V8V5
	V8V5
	V5



```cpp
#include<iostream>
#include<queue>
#define v 8
using namespace std;
class n {
public:
    int val;
    int st;
};
int matrix[v][v] = {
   {0, 1, 1, 1, 0, 1, 1, 0},
   {1, 0, 1, 1, 0, 0, 1, 1},
   {1, 1, 0, 1, 0, 1, 1, 0 },
   {1, 1, 1, 0, 0, 0, 1 ,0},
   {0, 0, 0, 0, 0, 1, 7, 0},
   {1, 0, 0, 0, 1, 0, 1, 0},
    {1, 1, 1, 1 ,1, 1,0,0},
    {0,1,0,0,0,0,0,0}
};
void bfs(n* verh, n s) {
    n u;
    int i, j;
    queue<n> que;
    for (i = 0; i < v; i++) {
        verh[i].st = 0;
    }
    verh[s.val].st = 1;
    que.push(s);
    while (!que.empty()) {
        u = que.front();
        que.pop();
        cout << u.val+1 << " ";
        for (i = 0; i < v; i++) {
            if (matrix[i][u.val]) {

```

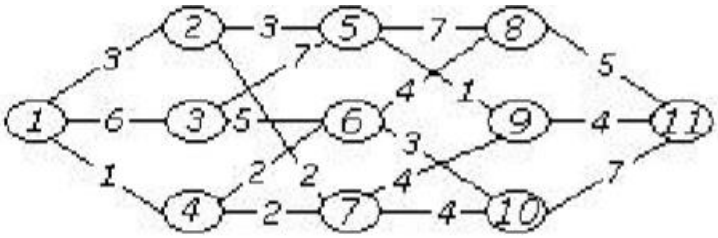
```
            verh[i].st = 0;
        }
        verh[s.val].st = 1;
        que.push(s);
        while (!que.empty()) {
            u = que.front();
            que.pop();
            cout << u.val+1 << " ";
            for (i = 0; i < v; i++) {
                if (matrix[i][u.val]) {

                    if (verh[i].st == 0) {
                        verh[i].st = 1;
                        que.push(verh[i]);
                    }
                }
            }
            u.st = 2;
        }
    }
int main() {
    n verh[v];
    n start;
    int s;
    for (int i = 0; i < v; i++) {
        verh[i].val = i;
    }
    s = 65;
    start.val = s - 65;
    cout << "bfs: ";
    bfs(verh, start);
    cout << endl;
}
```



Завдання № 5

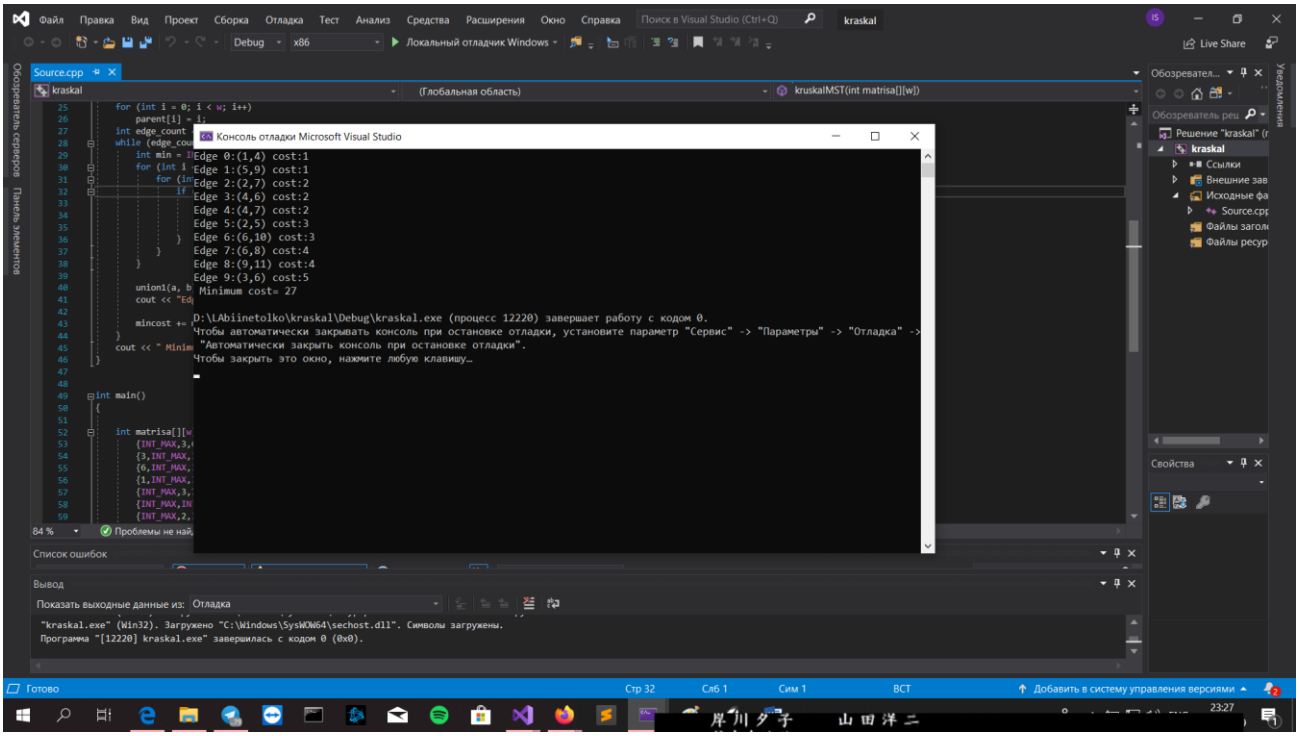
Знайти двома методами (Краскала і Прима) мінімальне остове дерево графа.

10)



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

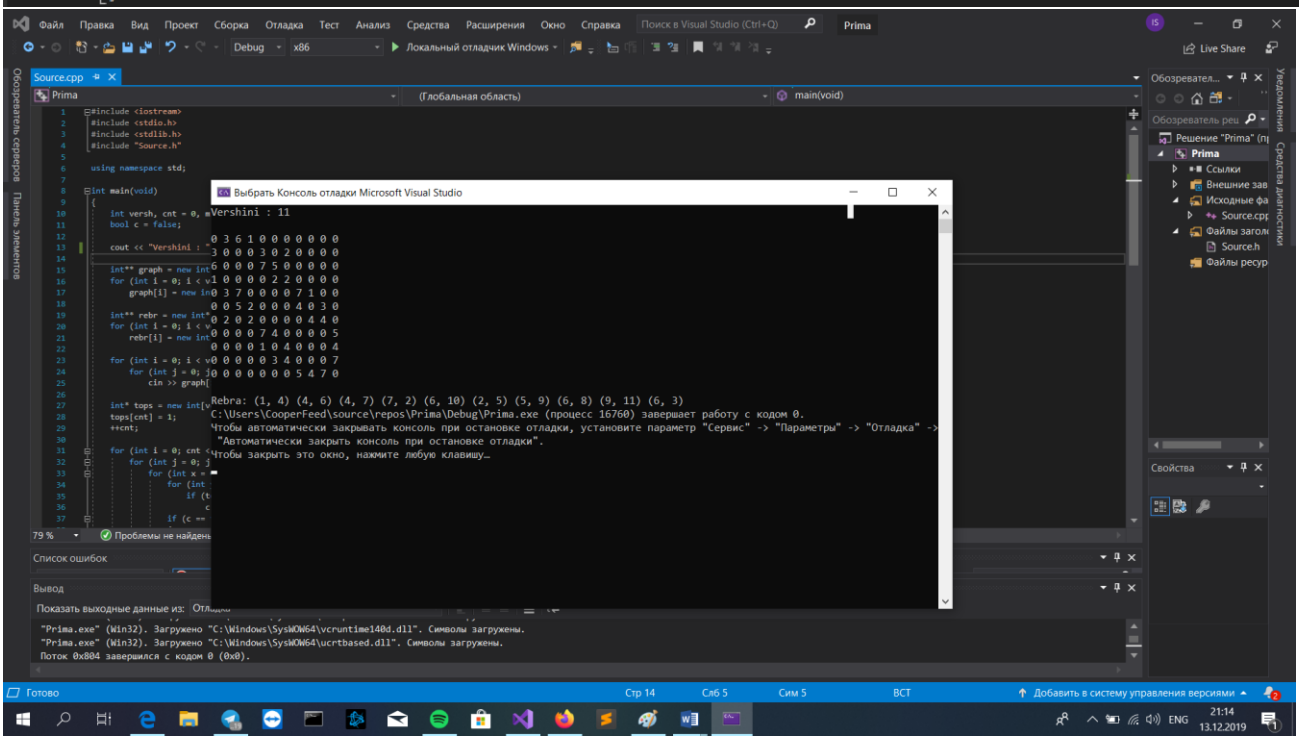
```cpp
#include "iostream"
using namespace std;

#define w 11
int parent[w];

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

void union1(int i, int j)
{
    int a = search(i);
    int b = search(j);
    parent[a] = b;
}
void kruskalMST(int matrisa[][w])
{
    int mincost = 0;

    for (int i = 0; i < w; i++)
        parent[i] = i;
    int edge_count = 0;
    while (edge_count < w - 1) {
        int min = INT_MAX, a = -1, b = -1;
        for (int i = 0; i < w; i++) {
            for (int j = 0; j < w; j++) {
                if (search(i) != search(j) && matrisa[i][j] < min) {
                    min = matrisa[i][j];
                    a = i;
                    a = i;
                    b = j;
                }
            }
        }

        union1(a, b);
        cout << "Edge " << edge_count++ << ":" << "(" << a + 1 << "," << b + 1 << ")" << " cost:" << min << endl;

        mincost += min;
    }
    cout << " Minimum cost= " << mincost << endl;;
}


int main()
{

    int matrisa[][w] = {
        {INT_MAX,3,6,1,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
        {3,INT_MAX,INT_MAX,INT_MAX,3,INT_MAX,2,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
        {6,INT_MAX,INT_MAX,INT_MAX,7,5,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
        {1,INT_MAX,INT_MAX,INT_MAX,INT_MAX,2,2,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
        {INT_MAX,3,7,INT_MAX,INT_MAX,INT_MAX,INT_MAX,7,1,INT_MAX,INT_MAX},
        {INT_MAX,INT_MAX,5,2,INT_MAX,INT_MAX,INT_MAX,4,INT_MAX,3,INT_MAX},
        {INT_MAX,2,INT_MAX,2,INT_MAX,INT_MAX,INT_MAX,INT_MAX,4,4,INT_MAX},
        {INT_MAX,INT_MAX,INT_MAX,INT_MAX,7,4,INT_MAX,INT_MAX,INT_MAX,INT_MAX,5},
        {INT_MAX,INT_MAX,INT_MAX,INT_MAX,1,INT_MAX,4,INT_MAX,INT_MAX,INT_MAX,4},
        {INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,3,4,INT_MAX,INT_MAX,INT_MAX,7},
        {INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,5,4,7,INT_MAX}
    };
    kruskalMST(matrisa);
}
```



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

```cpp
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include "Source.h"

using namespace std;

int main(void)
{
    int versh, cnt = 0, min_ = 0, n, m;
    bool c = false;

    cout << "Vershini : "; cin >> versh; cout << "\n";

    int** graph = new int* [versh];
    for (int i = 0; i < versh; ++i)
        graph[i] = new int[versh];

    int** rebr = new int* [versh - 1];
    for (int i = 0; i < versh - 1; ++i)
        rebr[i] = new int[2];

    for (int i = 0; i < versh; ++i)
        for (int j = 0; j < versh; ++j)
            cin >> graph[i][j];

    int* tops = new int[versh];
    tops[cnt] = 1;
    ++cnt;

    for (int i = 0; i < cnt; ++i) {
        for (int j = 0; j < cnt; ++j) {
            for (int x = 0; x < versh; ++x) {
                for (int y = 0; y < cnt; ++y)
                    if (tops[y] == x + 1)
                        c = true;
                if (c == true)
```

```
                for (int y = 0; y < cnt; ++y)
                    if (tops[y] == x + 1)
                        c = true;
                if (c == true)
                {
                    c = false;
                    continue;
                }

                if (min_ == 0 && graph[tops[j] - 1][x] > 0)
                {
                    min_ = graph[tops[j] - 1][x];
                    n = rebr[cnt - 1][0] = tops[j];
                    m = rebr[cnt - 1][1] = x + 1;
                    continue;
                }

                if (graph[tops[j] - 1][x] > 0 && graph[tops[j] - 1][x] < min_)
                {
                    min_ = graph[tops[j] - 1][x];
                    n = rebr[cnt - 1][0] = tops[j];
                    m = rebr[cnt - 1][1] = x + 1;
                }
            }
        }

        graph[n - 1][m - 1] = 0;
        graph[m - 1][n - 1] = 0;
        tops[cnt] = m;
        ++cnt;
        min_ = 0;
    }

    cout << endl << "Rebra: ";
    for (int i = 0; i < versh - 1; ++i)
        cout << "(" << rebr[i][0] << ", " << rebr[i][1] << ") ";
}
```



Завдання № 6

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

10)

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

1:

1 2 5 4 7 3 8 6 1 : 17

1 2 8 3 7 4 5 6 1 : 17

1 2 8 6 5 4 7 3 1 : 15

2:

2 1 3 7 4 5 6 8 2 : 15

2 1 3 8 6 5 4 7 2 : 16

2 1 7 4 5 6 8 3 2 : 17

3:

//3 1 2 7 4 5 6 8 3 :

3 1 2 8 6 5 4 7 3 : 15

3 7 4 5 6 8 1 2 3 : 18

3 8 6 5 4 7 1 2 3 : 17

4:

4 5 6 8 3 1 2 7 4 : 16

4 5 6 8 3 7 1 2 4 : 17

4 7 1 2 5 6 8 3 4 : 17

4 7 1 2 8 3 5 6 4 : 16

4 7 1 2 8 6 5 3 4 : 18

4 7 3 1 2 5 6 8 4 : 19

4 7 3 1 2 8 6 5 4 : 15

4 7 3 8 6 5 1 2 4 :20

5:

5 4 7 1 2 8 3 6 5 : 16

5 4 7 1 2 8 6 3 5 : 19

5 4 7 3 1 2 8 6 5 : 15

5 4 7 3 8 6 1 2 5 : 18

6:

6 5 4 7 1 2 8 3 6 : 16

6 5 4 7 3 1 2 8 6 : 15

6 5 4 7 3 8 1 2 6 : 16

6 8 3 7 4 5 2 1 6 : 17

6 8 3 1 2 5 4 7 6 : 18

7:

7 4 5 6 8 3 1 2 7 : 15

8:

8 3 1 2 5 4 6 7 8 : 21

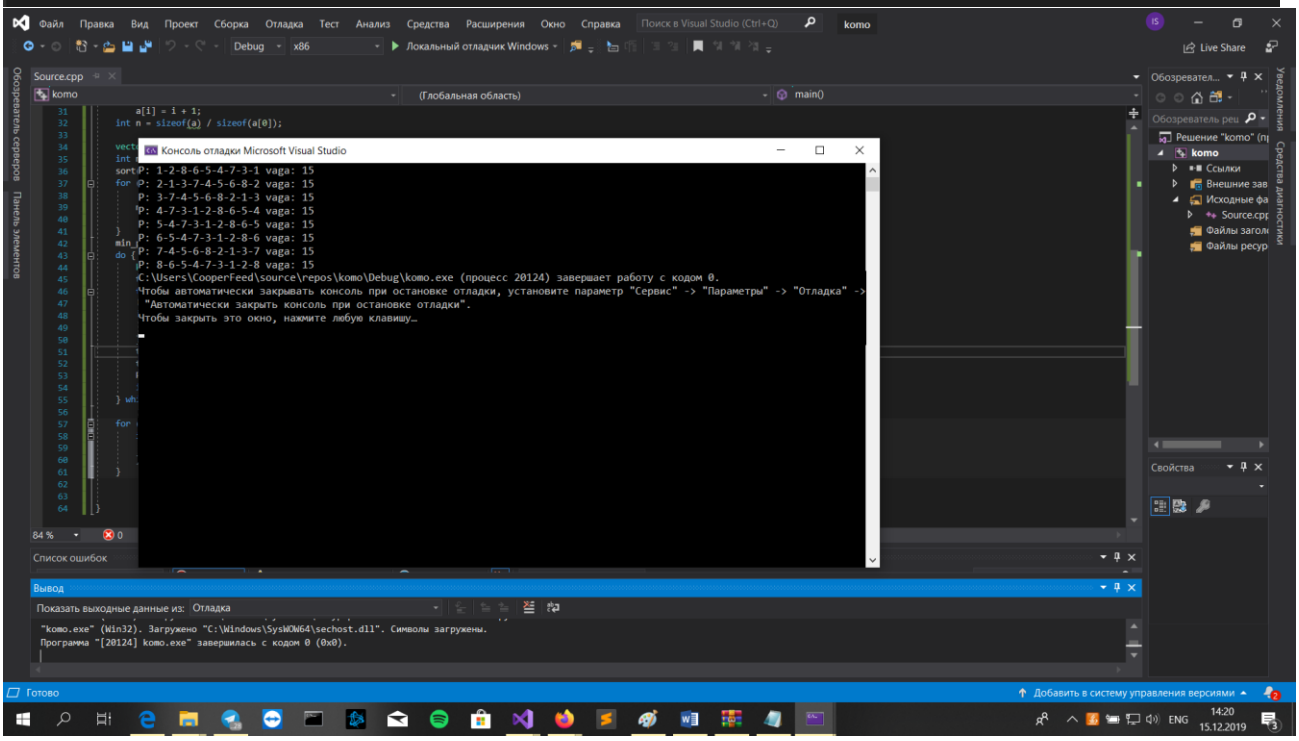
8 3 7 4 5 6 1 2 8 : 18

8 6 5 4 7 1 2 3 8 : 17

8 6 5 4 7 3 1 2 8 : 15

```cpp
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>

using namespace std;

struct k {

    string n;
    int nb;
};
void main()
{
const   int v = 8;
int matrix[v][v] = {
{100,1,2,3,5,4,2,3},
{1,100,6,5,4,6,5,4},
{2,6,100,3,3,5,2,2},
{3,5,2,100,1,6,1,5},
{5,4,3,1,100,2,4,5},
{4,6,5,6,2,100,6,2},
{2,5,2,1,4,6,100,7},
{3,4,2,5,5,2,7,100}
};

    int* a = new int[v];
    for (int i = 0; i < v; i++)
        a[i] = i + 1;
    int n = sizeof(a) / sizeof(a[0]);

    vector<k> Path;
    int min_path = 0;
```

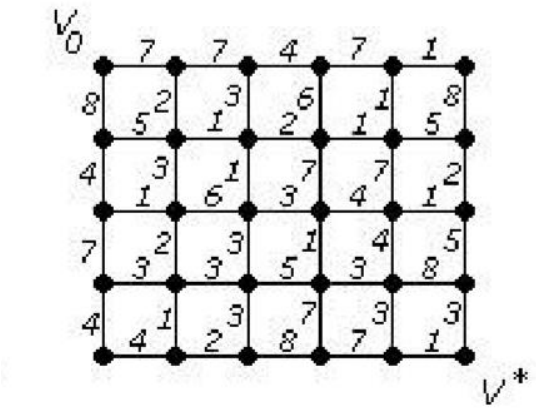
```
        a[i] = i + 1;
    int n = sizeof(a) / sizeof(a[0]);

    vector<k> Path;
    int min_path = 0;
    sort(a, a + v);
    for (int i = 1; i < v; i++) {

        min_path += matrix[a[i - 1] - 1][a[i] - 1];

    }
    min_path += matrix[a[v - 1] - 1][a[0] - 1];
    do {
        k t;
        t.n = to_string(a[0]); t.nb = 0;
        for (int i = 1; i < v; i++) {
            t.n+= "-" + to_string(a[i]);
            t.nb += matrix[a[i - 1] - 1][a[i] - 1];

        }
        t.n += "-" + to_string(a[0]);
        t.nb += matrix[a[v - 1] - 1][a[0] - 1];
        Path.push_back(t);
        if (min_path > t.nb) min_path = t.nb;
    } while (next_permutation(a, a + v));

    for (int i = 0; i < Path.size(); i++) {
        if (Path[i].nb == min_path) {
            cout << "P: " << Path[i].n << " " << "vaga: " << Path[i].nb << endl;
        }
    }


}
```

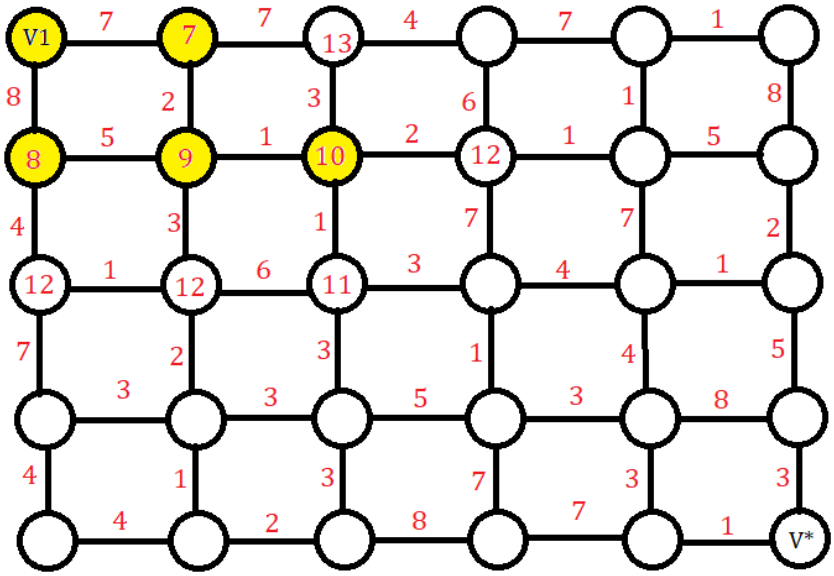


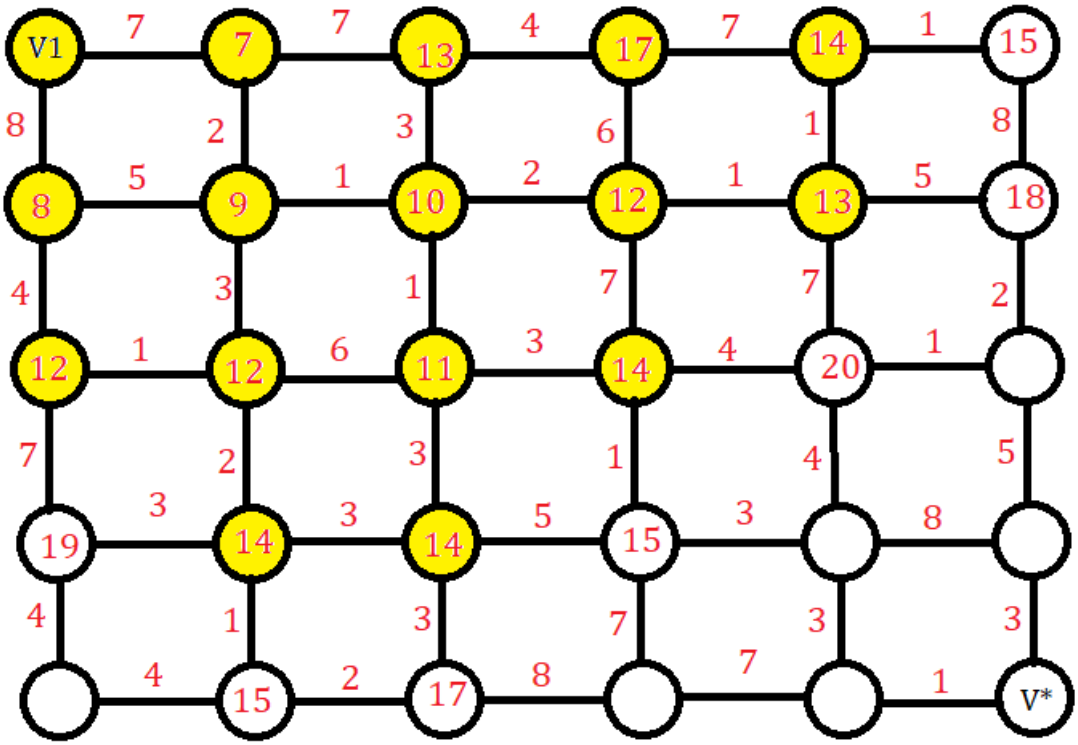
Завдання № 7

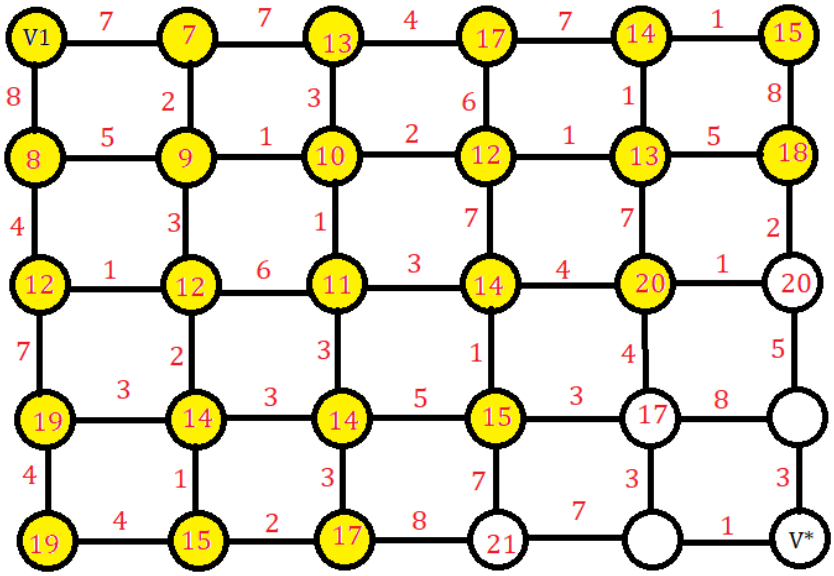
За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі між парою вершин V0 і V* .

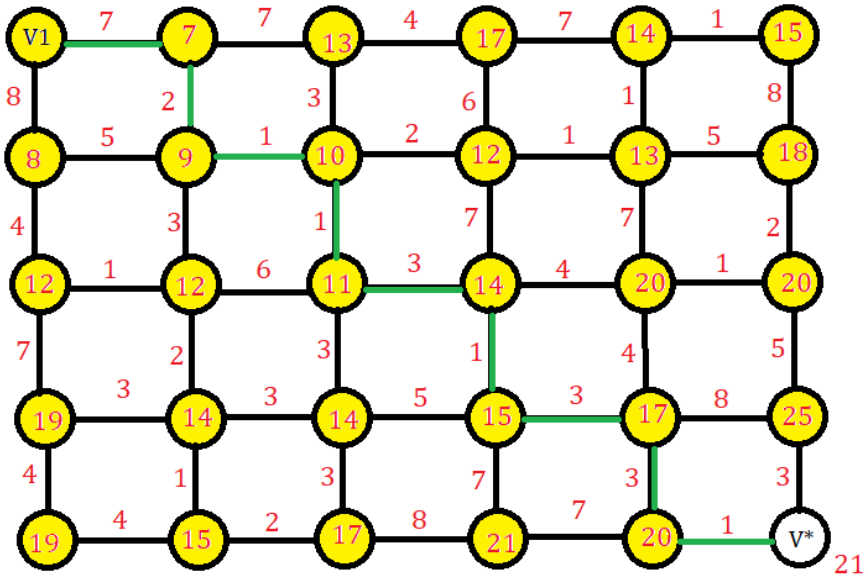
10)



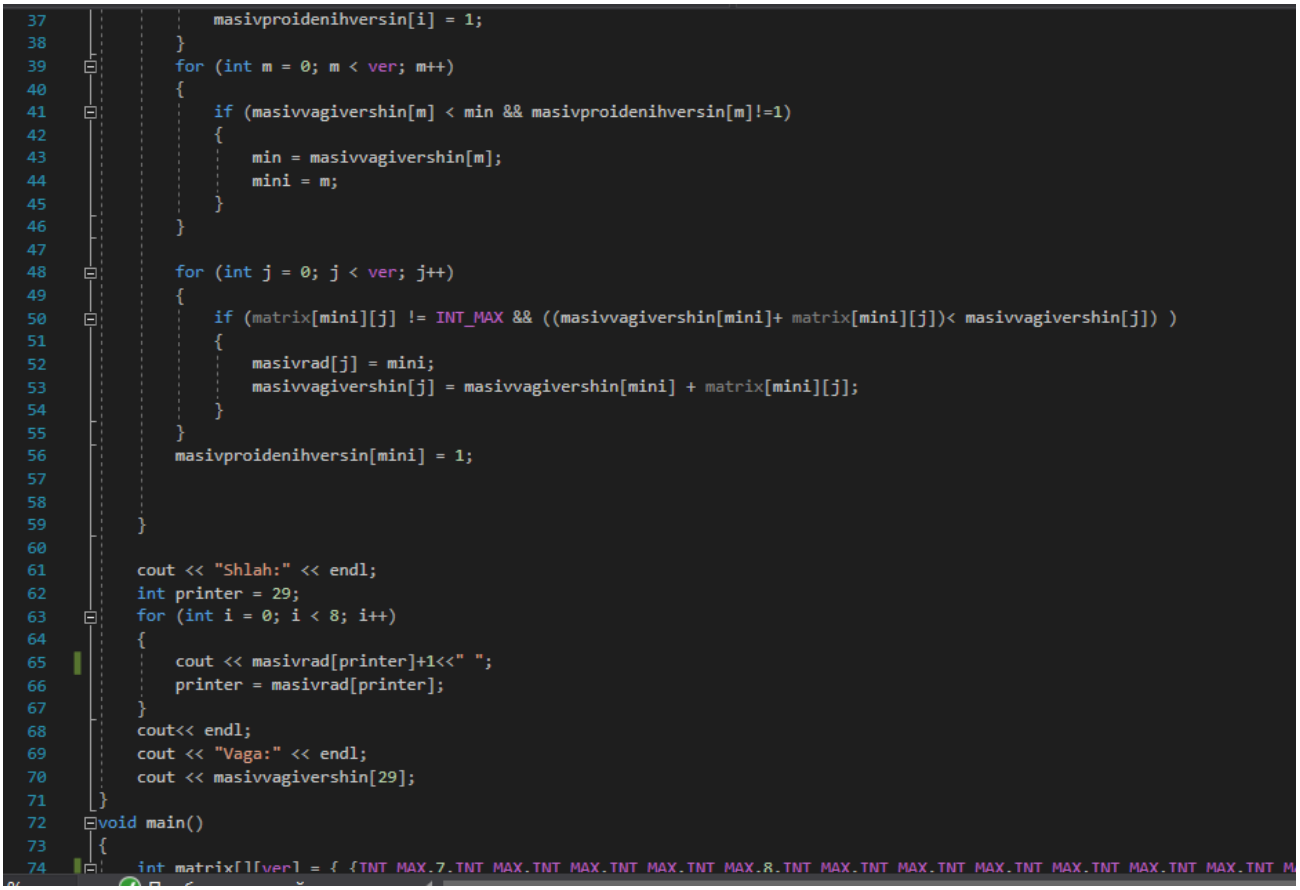

V1
7
7
13
8
9
10
12
12
12
11
V*


V1
7
7
13
17
14
15
8
9
10
12
13
18
12
12
11
14
20
19
14
14
15
15
17
V*

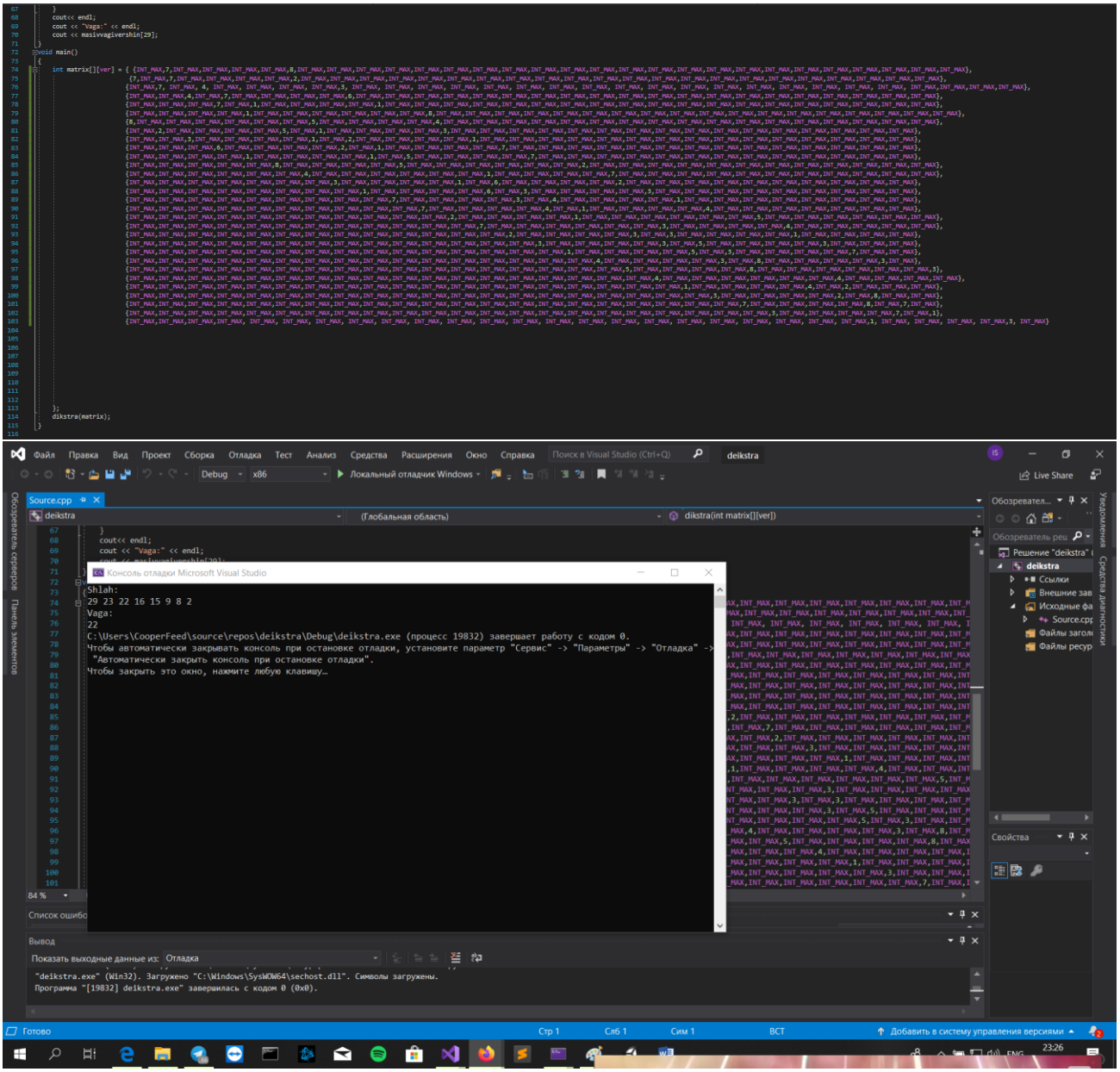

V1
7
7
13
17
14
15
8
9
10
12
13
18
12
12
11
14
20
20
19
14
14
15
17
19
15
17
21
V*


V1
7
7
13
17
14
15
8
9
10
12
13
18
12
12
11
14
20
20
19
14
14
15
17
25
19
15
17
21
20
V*
21

```cpp
#include "iostream"
#define ver 30
using namespace std;

void dikstra(int matrix[][ver])
{
    int masivproidenihversin[ver];
    int masivvagivershin[ver];
    int masivrad[ver];
    masivvagivershin[0] = 1000;
    for (int i = 1; i < ver; i++)
    {
        masivvagivershin[i] = 99999;
    }
    for (int i = 0; i < ver; i++)
    {
        masivproidenihversin[i] = 0;
    }
    for (int i = 0; i < ver; i++)
    {
        masivrad[i] = 99;
    }
    int min, mini;
    for (int i = 0; i < ver; i++)
    {
        min = 2000;
        if (i == 0)
        {
            for (int j = 0; j < ver; j++)
            {
                if (matrix[i][j] != INT_MAX)
                {
                    masivrad[j] = i;
                    masivvagivershin[j] = matrix[i][j];
                }
            }
            masivproidenihversin[i] = 1;
        }
```



```
            masivproidenihversin[i] = 1;
        }
        for (int m = 0; m < ver; m++)
        {
            if (masivvagivershin[m] < min && masivproidenihversin[m]!=1)
            {
                min = masivvagivershin[m];
                mini = m;
            }
        }

        for (int j = 0; j < ver; j++)
        {
            if (matrix[mini][j] != INT_MAX && ((masivvagivershin[mini]+ matrix[mini][j])< masivvagivershin[j]) )
            {
                masivrad[j] = mini;
                masivvagivershin[j] = masivvagivershin[mini] + matrix[mini][j];
            }
        }
        masivproidenihversin[mini] = 1;


    }

    cout << "Shlah:" << endl;
    int printer = 29;
    for (int i = 0; i < 8; i++)
    {
        cout << masivrad[printer]+1<<" ";
        printer = masivrad[printer];
    }
    cout<< endl;
    cout << "Vaga:" << endl;
    cout << masivvagivershin[29];
}
void main()
{
    int matrix[][ver] = { {INT_MAX,7,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,8,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX,INT_MAX},
```

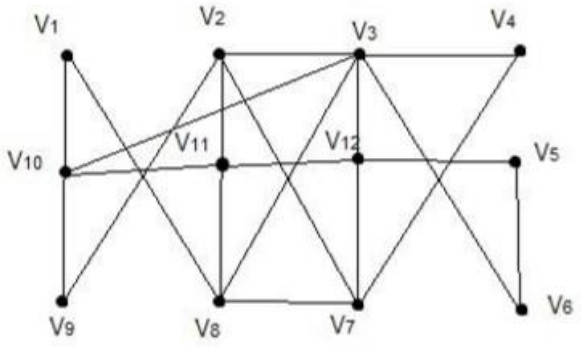


Завдання № 8

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

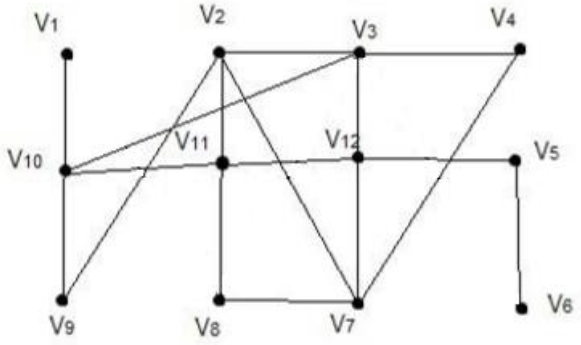
а) Флері

10)



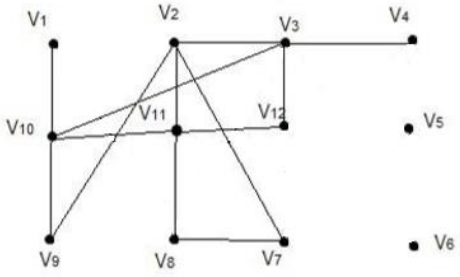
1-8-3-6

10)



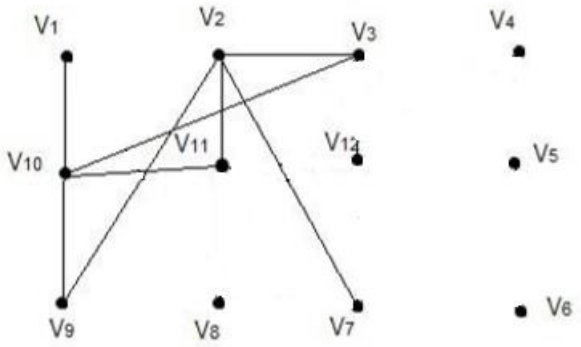
6-5-12-7-4

10)



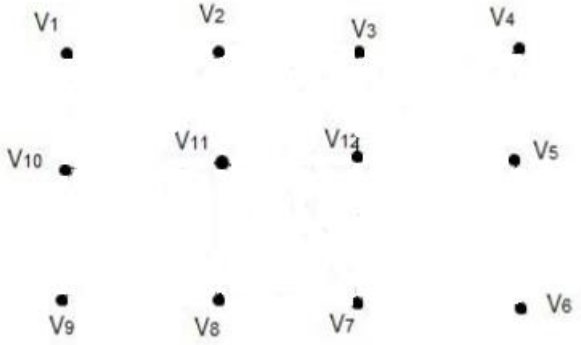
4-3-12-11-8-7

10)



7-2-3-10-9-2-11-10-1

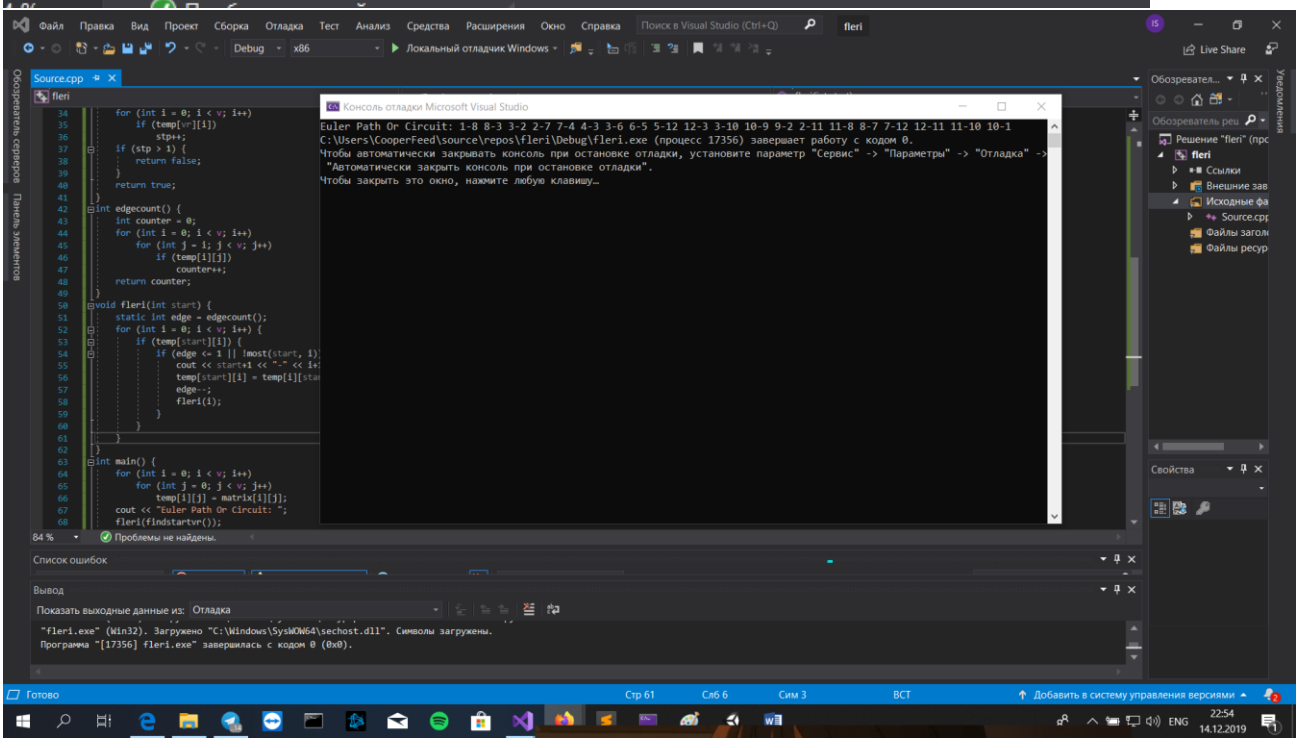
10)



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

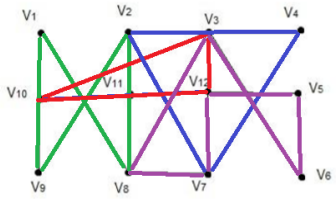
```cpp
#include<iostream>
#include<vector>
#define v 12
using namespace std;
int matrix[v][v] = {
    {0,0,0,0,0,0,0,1,0,1,0,0},
    {0,0,1,0,0,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1,0,1,0,1},
    {0,0,1,0,0,0,1,0,0,0,0,0},
    {0,0,0,0,0,1,0,0,0,0,0,1},
    {0,0,1,0,1,0,0,0,0,0,0,0},
    {0,1,0,1,0,0,0,1,0,0,0,1},
    {1,0,1,0,0,0,1,0,0,0,1,0},
    {0,1,0,0,0,0,0,0,0,1,0,0},
    {1,0,1,0,0,0,0,0,1,0,1,0},
    {0,1,0,0,0,0,0,1,0,1,0,1},
    {0,0,1,0,1,0,1,0,0,0,1,0}
};
int temp[v][v];
int findstartvr() {
    for (int i = 0; i < v; i++) {
        int stp = 0;
        for (int j = 0; j < v; j++) {
            if (temp[i][j])
                stp++;
        }
        if (stp % 2 != 0)
            return i;
    }
    return 0;
}
bool most(int u, int vr) {
    int stp = 0;
    for (int i = 0; i < v; i++)
        if (temp[vr][i])
```

```
    for (int i = 0; i < v; i++)
        if (temp[vr][i])
            stp++;
    if (stp > 1) {
        return false;
    }
    return true;
}
int edgecount() {
    int counter = 0;
    for (int i = 0; i < v; i++)
        for (int j = i; j < v; j++)
            if (temp[i][j])
                counter++;
    return counter;
}
void fleri(int start) {
    static int edge = edgecount();
    for (int i = 0; i < v; i++) {
        if (temp[start][i]) {
            if (edge <= 1 || !most(start, i)) {
                cout << start+1 << "-" << i+1 << " ";
                temp[start][i] = temp[i][start] = 0;
                edge--;
                fleri(i);
            }
        }
    }
}
int main() {
    for (int i = 0; i < v; i++)
        for (int j = 0; j < v; j++)
            temp[i][j] = matrix[i][j];
    cout << "Euler Path Or Circuit: ";
    fleri(findstartvr());
```



б) елементарних циклів.

10)



Цикли:

1-10-9-2-11-8-1

10-3-12-11-10

2-3-4-7-2

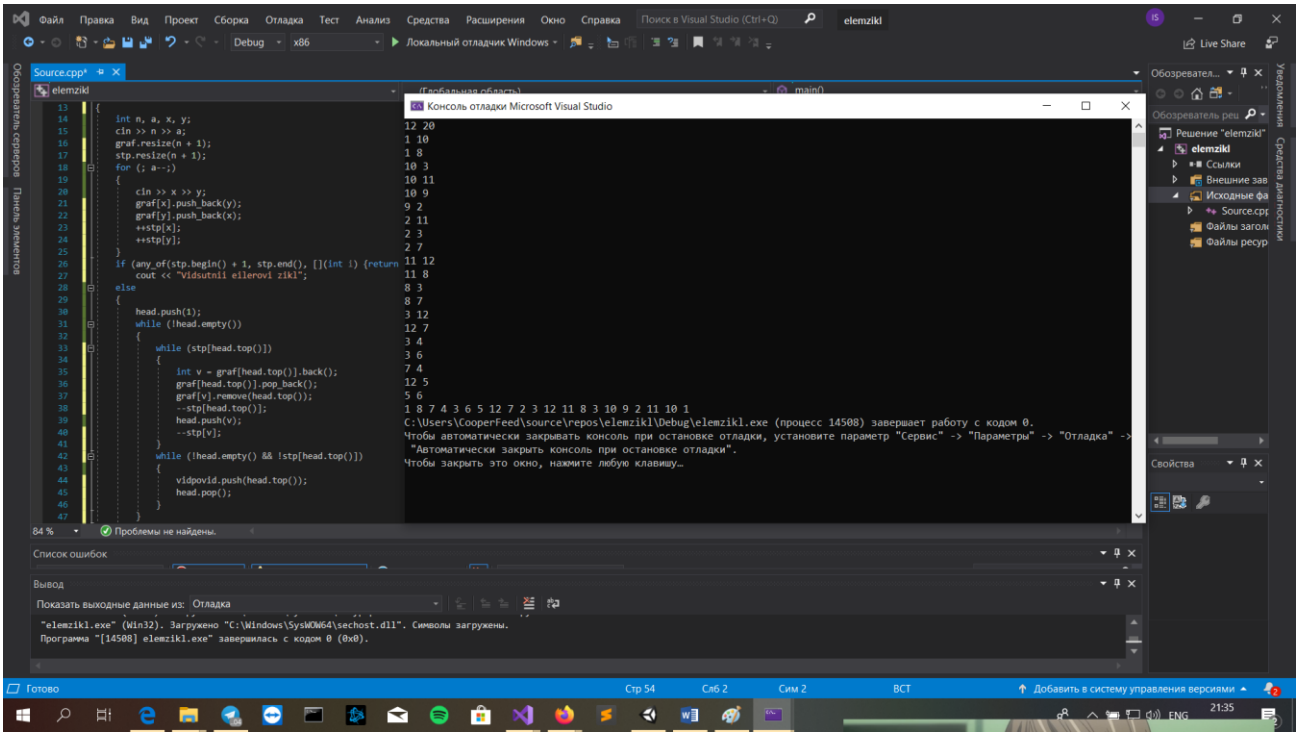
3-8-7-12-5-6-3

Ейлеровий цикл:

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

```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <algorithm>
#include <list>
using namespace std;
stack<int> head ;
stack<int>vidpovid;
vector <int> stp;
vector < list<int> > graf;

int main()
{
    int n, a, x, y;
    cin >> n >> a;
    graf.resize(n + 1);
    stp.resize(n + 1);
    for (; a--;)
    {
        cin >> x >> y;
        graf[x].push_back(y);
        graf[y].push_back(x);
        ++stp[x];
        ++stp[y];
    }
    if (any_of(stp.begin() + 1, stp.end(), [](int i) {return i & 1; }))
```

```
        graf[y].push_back(x);
        ++stp[x];
        ++stp[y];
    }
    if (any_of(stp.begin() + 1, stp.end(), [](int i) {return i & 1; }))
        cout << "Vidsutnii eilerovi zikl";
    else
    {
        head.push(1);
        while (!head.empty())
        {
            while (stp[head.top()])
            {
                int v = graf[head.top()].back();
                graf[head.top()].pop_back();
                graf[v].remove(head.top());
                --stp[head.top()];
                head.push(v);
                --stp[v];
            }
            while (!head.empty() && !stp[head.top()])
            {
                vidpovid.push(head.top());
                head.pop();
            }
        }
        while (!vidpovid.empty())
        {
            cout << vidpovid.top() << ' ';
            vidpovid.pop();
        }
    }
}
```



Завдання №9

Спростити формули (привести їх до скороченої ДНФ).

10. $xz \vee x\bar{z} \vee yz \vee \bar{x}yz$

xzVx¬zVyzV¬xyz

(xzV¬xyz) V (x¬zVyz)

yzVxy