МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Розрахункова робота

з дисципліни
«Дискретна математика »

Виконав:
студент групи КН-114
Брила Ярослав

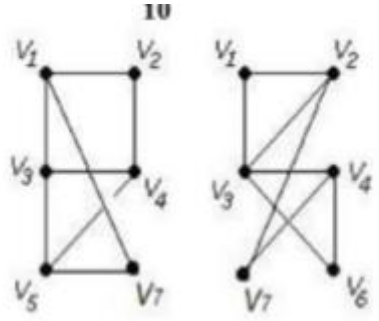
Викладач:
Мельникова Н.І.

Львів – 2019 р

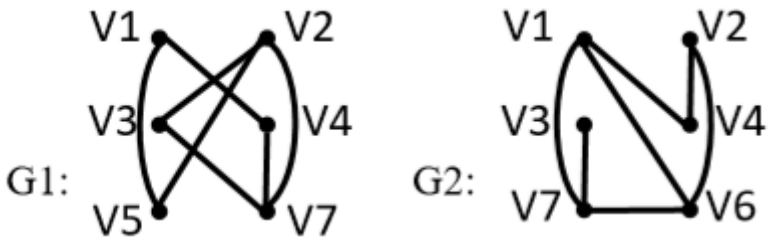
Варіант 10

Завдання №1

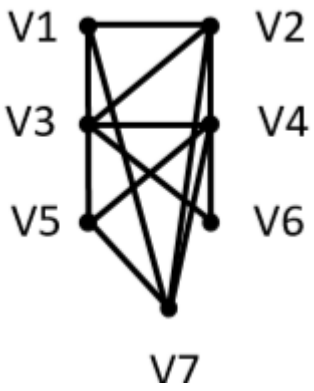
Виконати наступні операції над графами: 1) знайти доповнення до першого графу, 2) об'єднання графів, 3) кільцеву суму G1 та G2 (G1+G2), 4) розщепити вершину у другому графі, 5) виділити підграф А, що складається з 3-х вершин в G1 і знайти стягнення А в G1 (G1\ A), 6) добуток графів.



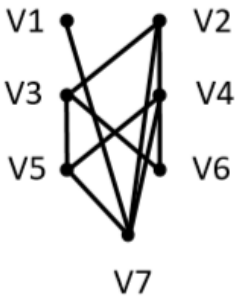
1.Доповнення



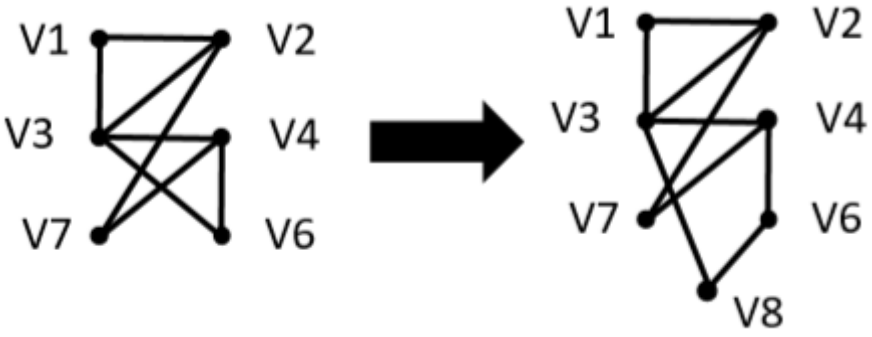
2.Об'єднання



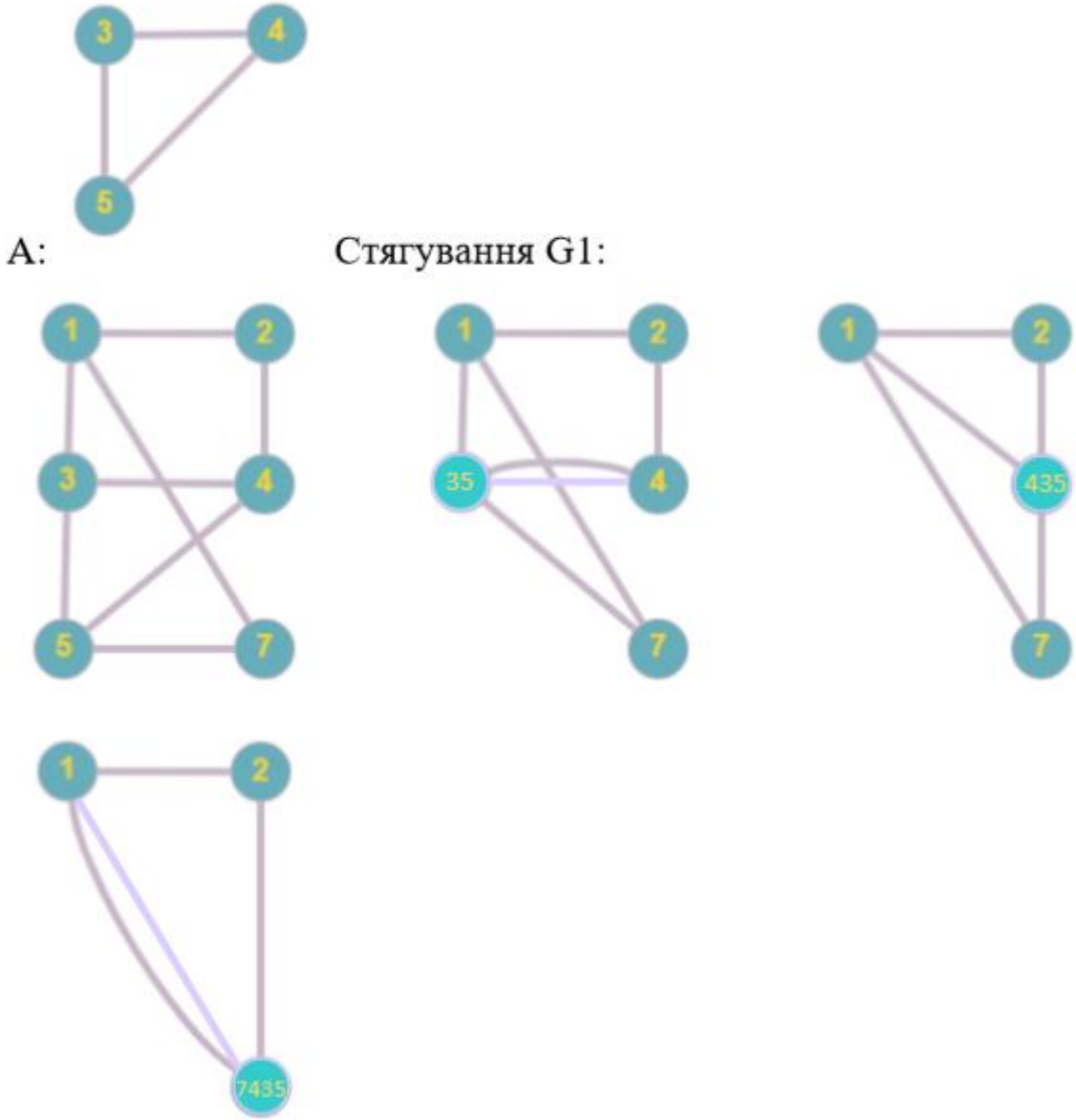
3.Кільцева сума



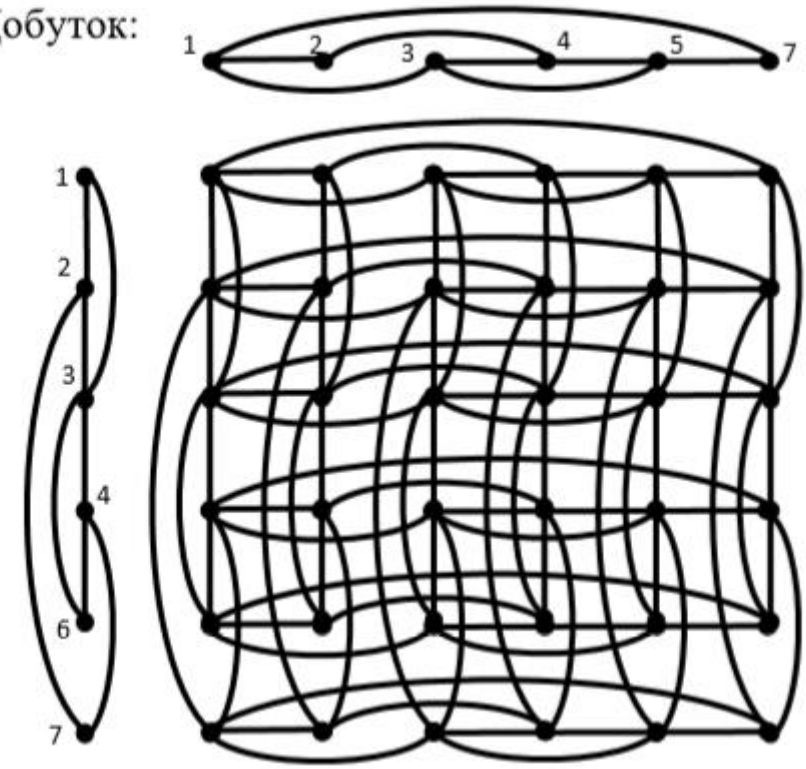
4. Розщеплення вершини V6 у другому графі



5. Виділення підграфа A з 3-х вершин в G1 і стягнення G1 в A: Стягуємо вершини G1, які належать A. Граф A складається з вершин V1, V2, V3.

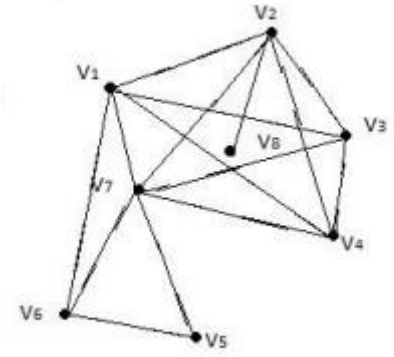


6) Добуток:



Завдання№2

10)

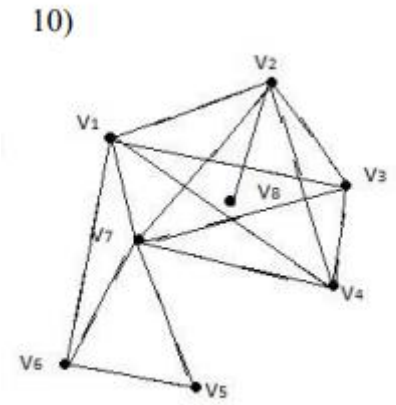


Побудувати таблицю суміжності

0	1	1	1	0	1	1	0
1	0	1	1	0	0	1	1
1	1	0	1	0	0	1	0
1	1	1	0	0	0	1	0
0	0	0	0	0	1	1	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

Завдання№3

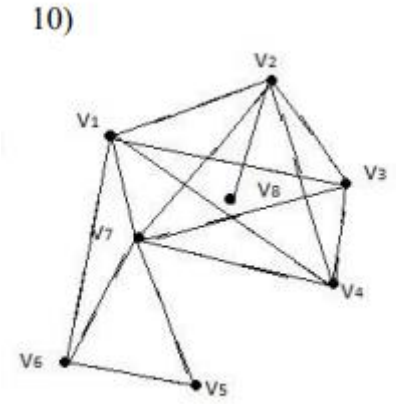
Для графа з другого завдання знайти діаметр.



Діаметр графа 4 (5-6-1-2-8)

Завдання№4

Обхід в ширину



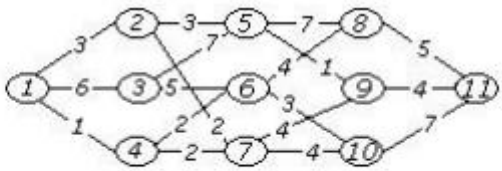
Вершина	BFS-номер	Вміст черги
V6	1	V6
V5	2	V6V5
V7	3	V6V5V7
V1	4	V6V5V7V1
-	-	V5V7V1
-	-	V7V1
V2	5	V7V1V2
V3	6	V7V1V2V3
V4	7	V7V1V2V3V4

-	-	V1V2V3V4
-	-	V2V3V4
V8	8	V2V3V4V8
-	-	V3V4V8
-	-	V4V8
-	-	V8
-	-	-

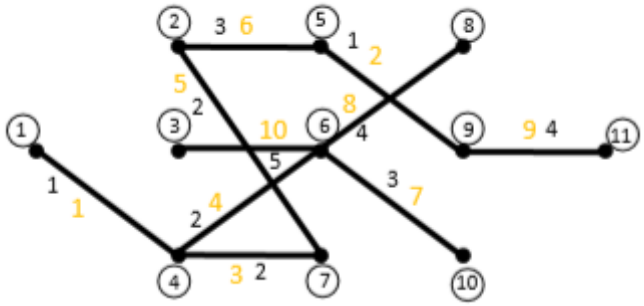
Завдання№5

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

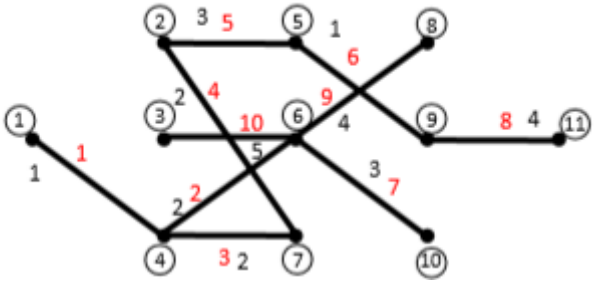
10)



Краскала:



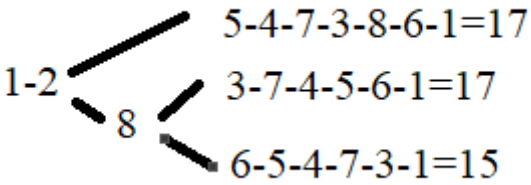
Прима



Завдання№6

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

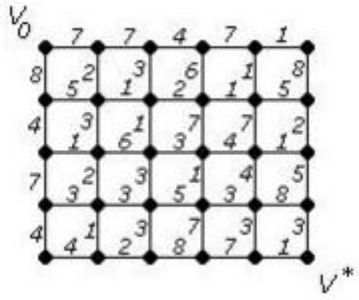
	1	2	3	4	5	6	7	8
1	∞	1	2	3	5	4	2	3
2	1	∞	6	5	4	6	5	4
3	2	6	∞	3	3	5	2	2
4	3	5	3	∞	1	6	1	5
5	5	4	3	1	∞	2	4	5
6	4	6	5	6	2	∞	6	2
7	2	5	2	1	4	6	∞	7
8	3	4	2	5	5	2	7	∞

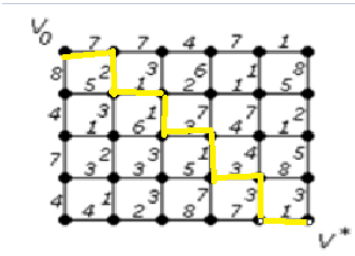


Мінімальна відстань:15

Завдання№7

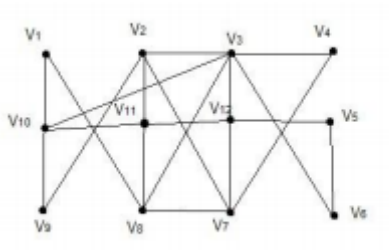
За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі між парою вершин V0 і V * .





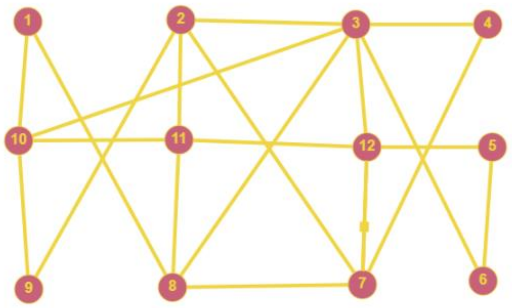
Довжина 22

Завдання№8 Знайти ейлеровий цикл в ейлеровому графі двома методами: а) Флері; б) елементарних циклів.

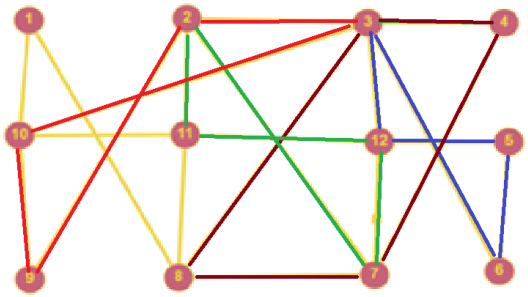


а)Флері

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



б)Елементарні цикли



Завдання№9

Спростити формули (привести їх до скороченої ДНФ).

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

Xz ∨¬xz ∨ yz ∨ ¬xyz=(x ∧ z) ∨(y ∧ z) ∨(z ∧ ¬x) ∨(y ∧ z ∧ ¬x)=z

Алгоритм Прима

```
#include <iostream>

using namespace std;
struct edge {
    int firstPoint;
    int secondPoint;
    int weight;
};
void sortEdges(edge* a, int n);
bool isInclude(int* arr, int n, int k);
void findTheTree(edge* a, int* points, edge *tree, int &n, int &k, int &i, int &j);
bool isMinimum(int w, edge* a, int k, int* points, int n);
int main() {
    int n;
    int k;
    cout << "Enter a number of points: ";
    cin >> n;
    cout << "Enter a number of edges: ";
    cin >> k;
    edge* edges = new edge[k];
    cout << "Enter edges (first point | second point | weight) " << endl;
    for (int i = 0; i < k; i++) {
        cin >> edges[i].firstPoint >> edges[i].secondPoint >> edges[i].weight;

    }
    sortEdges(edges, k);
    cout << endl;
    int* points = new int[n];
    points[0] = edges[0].firstPoint;
    points[1] = edges[0].secondPoint;
    edge* tree = new edge[n-1];
    tree[0].firstPoint = points[0];
    tree[0].secondPoint = points[1];
    int i = 2;
    int j = 1;



    findTheTree(edges, points, tree, &n, &k, &i, &j);
    cout << "V(G) = { ";
    for (int i = 0; i < n; i++) {
        cout << points[i] << ", ";
    }
    cout << "}" << endl;
    cout << "E(G) = { ";
    for (int i = 0; i < n-1; i++) {
        cout <<"("<< tree[i].firstPoint << "," << tree[i].secondPoint << ")"<<" ";
```

```
    }
    cout << " }" << endl;


    return 0;
}

void findTheTree(edge* a, int* points, edge *tree, int &n, int &k, int &i, int &j) {
    if(i == n){
        return;
    }
    else if (j == k) {
        j = 1;
    }

    if(isInclude(points, n, a[j].firstPoint) && isInclude(points, n, a[j].secondPoint)){
        j++;
        findTheTree(a, points, tree, &n, &k, &i, &j);
    }
    else if (!isInclude(points, n, a[j].firstPoint) && isInclude(points, n, a[j].secondPoint) && isMinimum(a[j].weight, a, k, points, n))
        tree[i - 1].firstPoint = a[j].secondPoint;
        tree[i - 1].secondPoint = a[j].firstPoint;
        points[i] = a[j].firstPoint;
        tree[i - 1].weight = a[j].weight;

        j++;
        i++;
        findTheTree(a, points, tree, &n, &k, &i, &j);

    }
    else if (isInclude(points, n, a[j].firstPoint) && !isInclude(points, n, a[j].secondPoint) && isMinimum(a[j].weight, a, k, points, n))
        tree[i - 1].firstPoint = a[j].firstPoint;
        tree[i - 1].secondPoint = a[j].secondPoint;
        points[i] = a[j].secondPoint;
        tree[i - 1].weight = a[j].weight;

        j++;
        i++;
        findTheTree(a, points, tree, &n, &k, &i, &j);
    }
    else {
        j++;
        findTheTree(a, points, tree, &n, &k, &i, &j);
```

```
        findTheTree(a, points, tree, &n, &k, &i, &j);
    }
}
void sortEdges(edge* a, int n) {
    edge temp;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (a[j].weight > a[j+1].weight) {
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
}

bool isInclude(int* arr, int n, int k) {
    for (int i = 0; i < n; i++) {
        if (k == arr[i]) {
            return true;
        }

    }
    return false;
}
bool isMinimum(int w, edge* a,int k, int* points, int n) {
    for (int j = 1; j < k; j++) {
        if (((!isInclude(points, n, a[j].firstPoint) && isInclude(points, n, a[j].secondPoint)) || (isInclude(points, n, a[j].firstPoint) && !isInclude(points, n, a[j].secondPoint))) && a[j].weight < w) {
            return false;
        }
    }
    return true;
}
```

Вивід програми

```
Enter a number of points:11
 Enter a number of edges:18
 Enter edges (first point | second point | weight)
1 2 7
1 3 3
1 4 1
2 7 1
2 5 2
3 6 4
3 5 7
4 6 2
4 7 5
5 8 4
5 9 4
6 8 6
6 10 2
7 9 3
7 10 3
8 11 7
9 11 4
10 11 5

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

Алгоритм Краскала

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

using namespace std;
vector<pair<int, pair<int, int> > > g; // вес - вершина 1 - вершина 2
vector<pair<int, int> > res;
int main() {
    int m,n,x,y,w,cost = 0;
    cin>>n>>m;
    vector<int> tree_id(n);

    for (int i = 0; i < m; ++i) {
        cin>>x>>y>>w; //точки та вага
        x--;y--;
        g.push_back({w,{x,y}});//вносимо в вектор
    }
    sort(g.begin(), g.end());

    for (int i = 0; i < n; ++i)
        tree_id[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_id[a] != tree_id[b])//порівнюємо дерева
        {
          cost += l;
            res.emplace_back(a+1, b+1);
            int old_id = tree_id[b], new_id = tree_id[a];
            for (int j = 0; j < n; ++j)
                if (tree_id[j] == old_id)
                    tree_id[j] = new_id;
        }
    }
    for(auto i:res) {
        cout<<i.first<<' '<<i.second<<endl;

    }
}
```

```
enter the number of points and the number of edges
11 18
enter points and edge weight
1 2 7
1 3 3
1 4 2
2 5 7
2 7 1
3 5 7
3 6 4
4 6 5
4 7 5
5 8 2
5 9 4
6 8 4
6 10 2
7 9 3
7 10 4
8 11 3
9 11 1
10 11 6
2 7
9 11
1 4
5 8
6 10
1 3
7 9
8 11
3 6
6 8

Weight: 25
```

Алгоритм Дейкстри

```
#include <iostream>
using namespace std;
int n,g[50][50]={0},dist[50],pred[50];
bool visit[50];

void way(int j)
{
    if (pred[j] == -1)
        return;
    way(pred[j]);
    cout << "Top" << j+1 << " -> ";
}
int distance()
{
    int minimum = 10000, minD;
    for (int z = 0; z < n; z++)
        if (visit[z] == false && dist[z] <= minimum)
        {
            minimum = dist[z];
            minD = z;
        }
    return minD;
}
void algo(int g[50][50])
{
    for (int i = 0; i < n; i++)
    {
        pred[0] = -1;
        dist[i] = 10000;
        visit[i] = false;
    }
    dist[0] = 0;
    for (int j = 0; j < n - 1; j++)
    {
        int u = distance();
        visit[u] = true;
        for (int z = 0; z < n; z++)
            if (!visit[z] && g[u][z] && dist[u] + g[u][z] < dist[z])
```

```
            {
                pred[z] = u;
                dist[z] = dist[u] + g[u][z];
            }
    }
    cout << "The least way is: ";
    cout << dist[29] << endl;
    cout << "The way is: ";
    cout << "Top -> ";
    way(29);
    cout << "finish" << endl;
}
int main()
{
n=30;
int v1, v2;
cout<<"Number of rows and columns ";
cin>>v1>>v2;
for (int i=0; i<n; i++){
for(int j=i+1; j<n; j++){
if(j==i+1 || j==i+v1){
cout<<"top "<<i+1<<" to top "<<j+1<<" :";
cin>>g[i][j];
}
else
g[i][j]=0;
}
}
algo(g);
}
```

```
Number of rows and columns6 5
top 1 to top 2 :7
top 1 to top 7 :4
top 2 to top 3 :5
top 2 to top 8 :3
top 3 to top 4 :3
top 3 to top 9 :3
top 4 to top 5 :8
top 4 to top 10 :4
top 5 to top 6 :4
top 5 to top 11 :4
top 6 to top 7 :0
top 6 to top 12 :1
top 7 to top 8 :7
top 7 to top 13 :2
top 8 to top 9 :3
top 8 to top 14 :1
top 9 to top 10 :1
top 9 to top 15 :7
top 10 to top 11 :2
top 10 to top 16 :4
top 11 to top 12 :2
top 11 to top 17 :1
top 12 to top 13 :0
top 12 to top 18 :7
top 13 to top 14 :3
top 13 to top 19 :1
top 14 to top 15 :8
top 14 to top 20 :5
top 15 to top 16 :8
top 15 to top 21 :6
top 16 to top 17 :7
top 16 to top 22 :6
top 17 to top 18 :7
top 17 to top 23 :7
top 18 to top 19 :0
top 18 to top 24 :7
top 19 to top 20 :1
top 19 to top 25 :4
top 20 to top 21 :3
top 20 to top 26 :2
top 21 to top 22 :1
top 21 to top 27 :3
top 22 to top 23 :1
top 22 to top 28 :8
top 23 to top 24 :5
top 23 to top 29 :1
top 24 to top 25 :0
top 24 to top 30 :2
top 25 to top 26 :3
top 26 to top 27 :3
top 27 to top 28 :4
top 28 to top 29 :3
top 29 to top 30 :1
The way is: Top1 -> Top7 -> Top13 -> Top19 -> Top20 -> Top21 -> Top22 -> Top23 -> Top29 -> Top30 -> Return 0;
The least way is: 15
```

Задача Комівояжера

```cpp
#include <iostream>
#include <utility>
#include <vector>
#include <algorithm>
using namespace std;
int matrix[100][100] = {0};
int controlLine[100] = {0};
pair<int, int> cpair[100];
vector<int> ind;

void insertIndex(int size, int ins){
    int d = INT_MAX;
    int nd;
    int current_position = -1;

    for(int i = 0; i < size - 1; i++){
        nd = matrix[ind[i]][ins] + matrix[ins][ind[i + 1]] - matrix[ind[i]][ind[i + 1]];
        if(nd < d){
            d = nd;
            current_position = i + 1;
        }
    }
    nd = matrix[ind[size - 1]][ins] + matrix[ins][ind[0]] - matrix[ind[size - 1]][ind[0]];
    if(nd < d){
        current_position = size;
    }
    ind.insert(ind.begin() + current_position, ins);

}
int main(int argc, const char * argv[]) {
    cout << "Enter points : " << endl;
    int n;
    cin >> n;

    for(int  i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            cin >> matrix[i][j];
            controlLine[j] += matrix[i][j];
        }
    }

    for(int i = 0; i < n; i++){
        cpair[i] = make_pair(controlLine[i], i);
    }
```

```
for(int i = 0; i < n; i++){
    cpair[i] = make_pair(controlLine[i], i);
}

sort(cpair, _Last: cpair + n);
reverse(cpair, _Last: cpair + n);

for(int i = 0; i < 3; i++)
    ind.push_back(cpair[i].second);


for(int i = 3; i < n; i++){
    insertIndex(i, cpair[i].second);
}
cout << "answer" << endl;
for(int i = 0; i < n ; i++){
    cout << ind[i] + 1 << ' ';
}
```

```
8
0 1 2 3 5 4 2 3
1 0 6 5 4 6 5 4
2 6 0 3 3 5 2 2
3 5 3 0 1 6 1 5
5 4 3 1 0 2 4 5
4 6 5 6 2 0 6 2
2 5 2 1 4 6 0 7
3 4 2 5 5 2 7 0
5 1 7
L = 17
6 7 2 8
L = 17
6 5 7 2 8
L = 15
6 5 4 7 2 8
L = 18
6 5 4 7 3 2 8
L = 15
6 5 4 7 3 1 2 8
answer
6 5 4 7 3 1 2 8
```

Пошук вшир

```cpp
#include <iostream>
#include <queue>
using namespace std;
queue <int> q;
int mas[9][9],p[9];
int main()
{
    int n;
    cin>>n;
    for (int i = 0; i <n; ++i) {
        for (int j = 0; j < n; ++j) {
        cin>>mas[i][j];
        }
    }
    q.push( _Val: 0);
    while (!q.empty())
    {
        int node = q.front();
        q.pop();
        p[node] = 2;
        for (int j = 0; j < 9; j++)
        {
            if (mas[node][j] == 1 && p[j] == 0)
            {
                q.push(j);
                p[j] = 1;
            }
        }
        cout << node + 1 << ' ';
    }
}
```

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

Алгоритм Флері

```cpp
#include <iostream>
using namespace std;
int k,st[100],g[100][100];
void Search(int v,int n)
{
    int i;
    for(i = 0; i < n; i++)
        if(g[v][i])
        {
            g[v][i] = g[i][v] = 0;
            Search(i,n);
        }
    st[++k] = v+1;
}
int main()
{
    int T, p, q, s,n;
    int j, vv;
    cin>>n;
    for (int i = 0; i < n; ++i) {
        for (int l = 0; l < n; ++l) {
            cin>>g[i][l];
        }
    }
    T = 1;
    for(p = 0; p < n; p++)
    {
        s = 0;
        for(q = 0; q < n; q++)
        {
            s += g[p][q];
        }
        if(s%2) T = 0;
    }
    k = -1;
    cout << "Start: ";
    cin >> vv;
    vv-=1;
    if(T)
    {
        Search (vv,n);
        for(j = 0; j <= k; j++)
            cout << st[j] << " ";
    }
    else
        cout << "it is not Eulerian graph\n";
```

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