МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

РОЗРАХУНКОВА-ГРАФІЧНА РОБОТА

з дисципліни

“ДИСКРЕТНА МАТЕМАТИКА”

Виконав: Гончаренко Назар

Студент групи КН-115

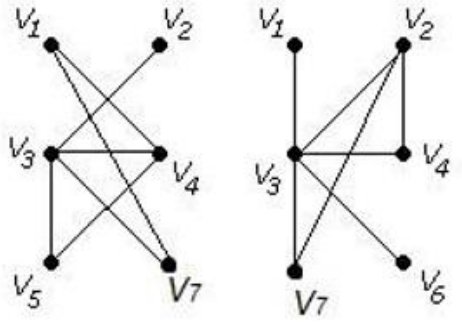
Варіант №12

2017

Варіант №12

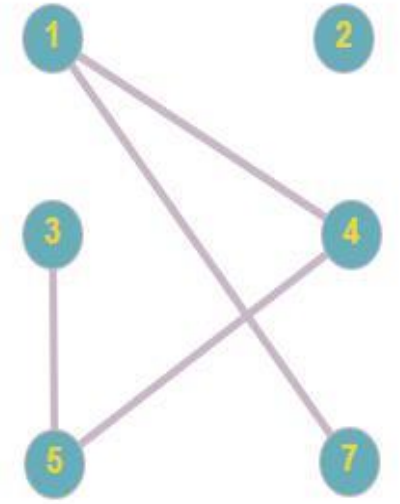
Завдання № 1 Виконати наступні операції над графами: 1) знайти доповнення до першого графу, 2) об'єднання графів, 3) кільцеву сумму G1 та G2 (G1+G2), 4) розмножити вершину у другому графі, 5) виділити підграф А - що скадається з 3-х вершин в G1 6) добуток графів.

12)

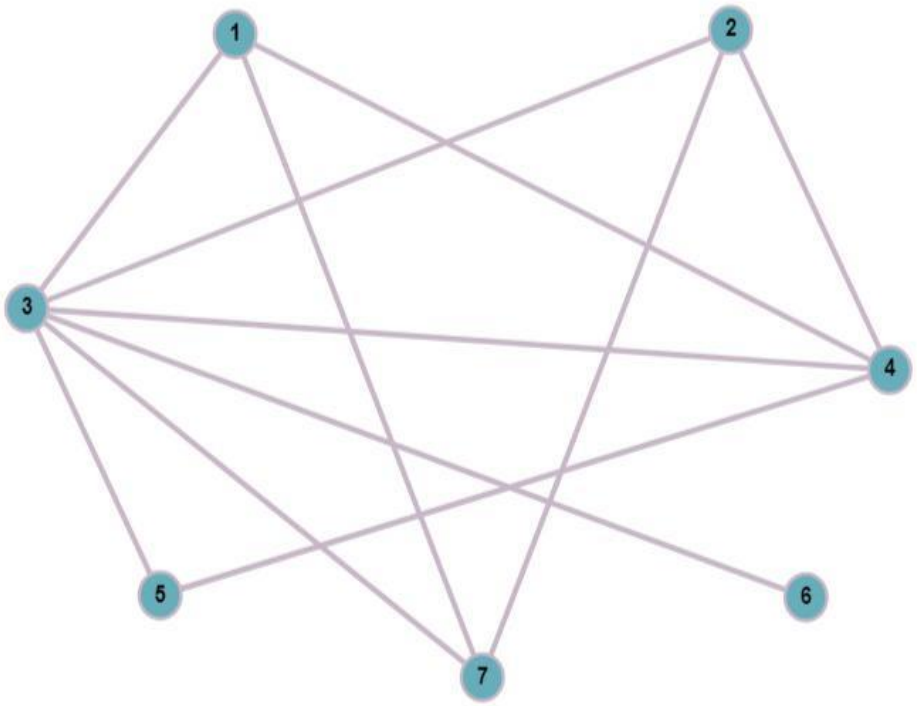


1)Доповнення

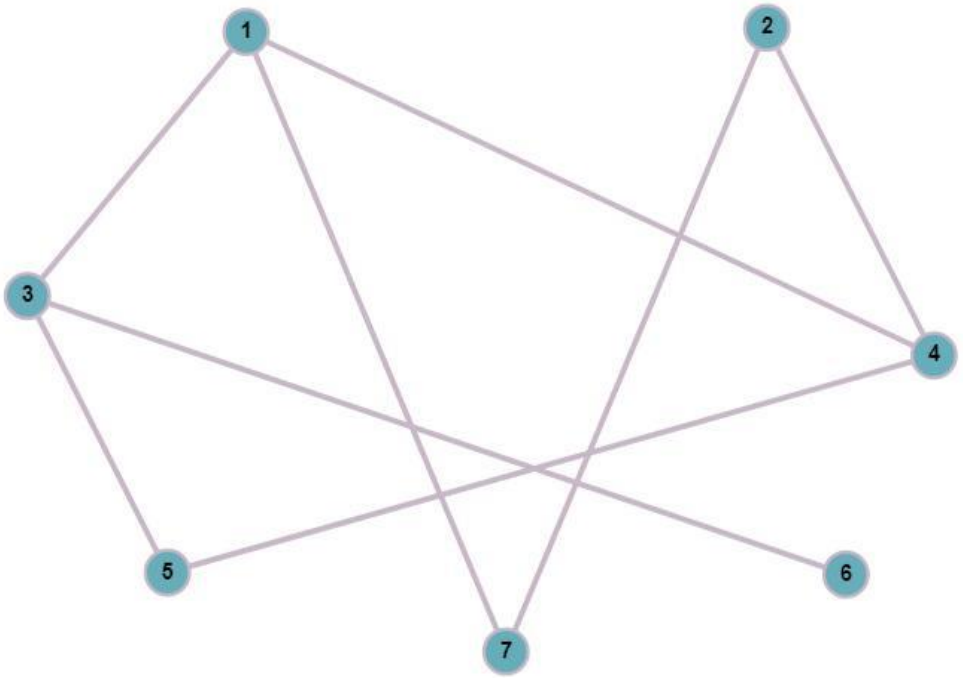
G1\G2:



2)

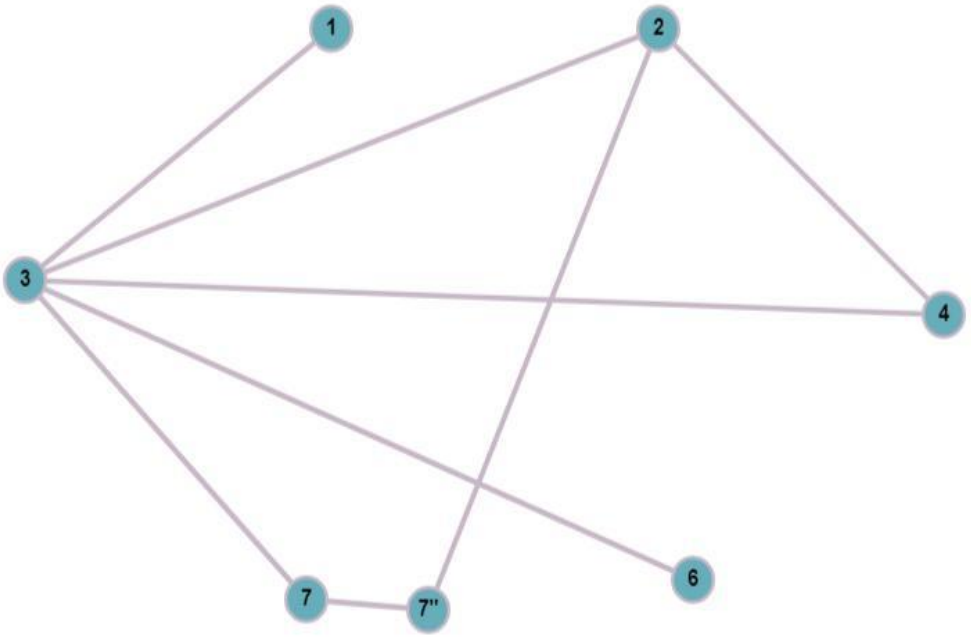


3)Кільцева сума:

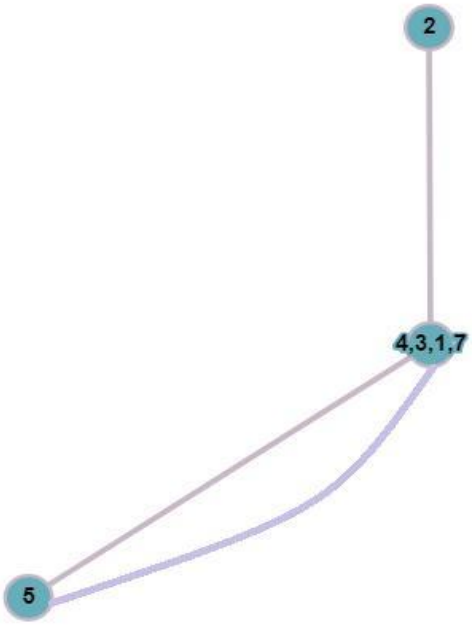


4)Розщепити вершину у другому графі:

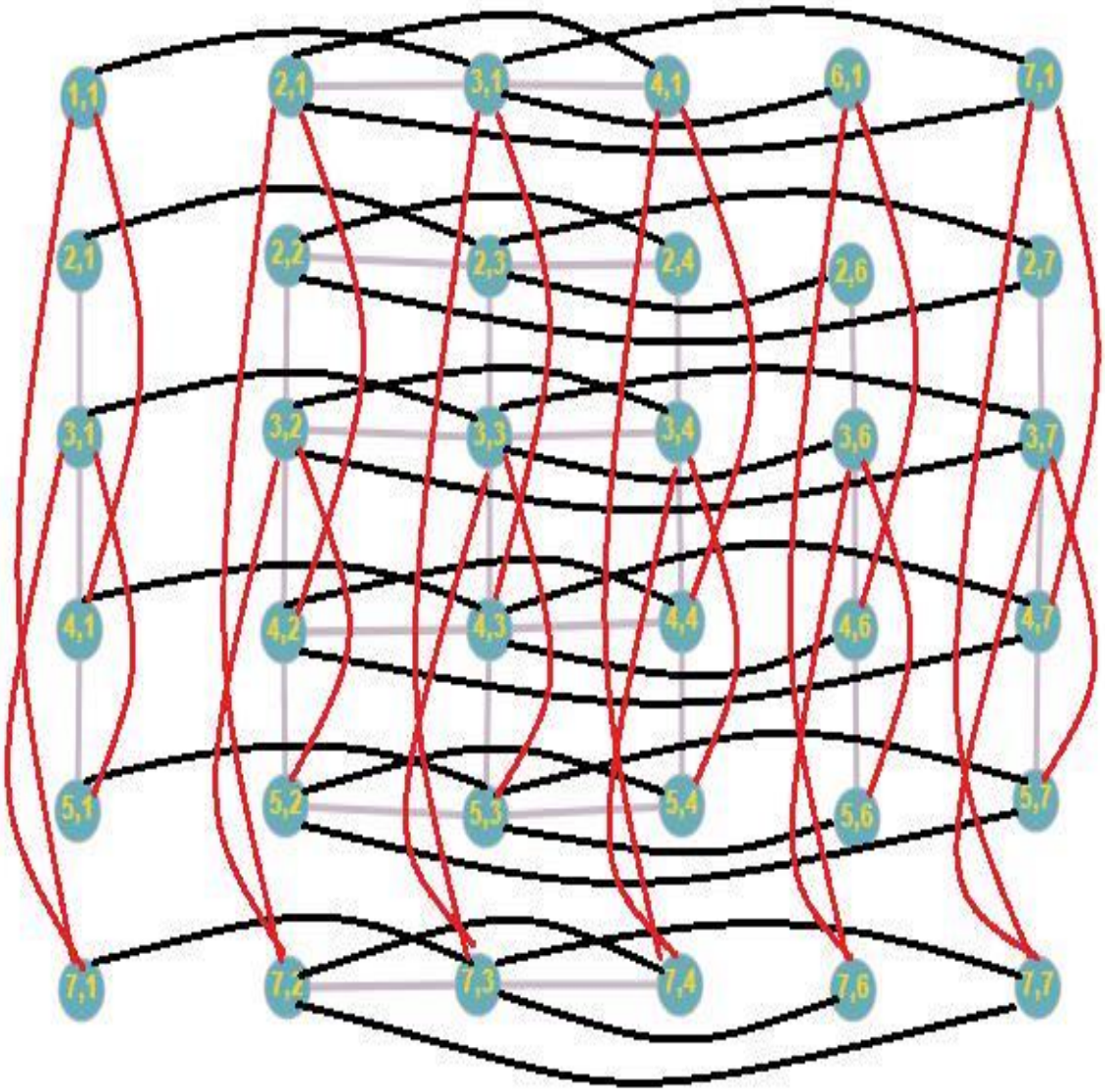
Розщепимо вершину 7



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

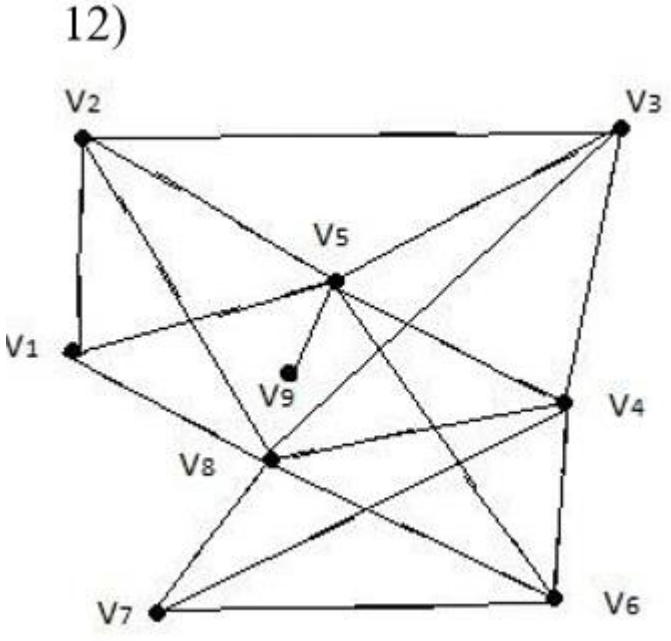


6)Добуток графів



Завдання № 2

Скласти таблицю суміжності для орграфа.



	V1	V2	V3	V4	V5	V6	V7	V8	V9
V1	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**1**	**0**
V2	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**
V3	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**
V4	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**
V5	**1**	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**1**
V6	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**0**
V7	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**0**
V8	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**0**
V9	**0**	**0**	**0**	**0**	**1**	**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 вшир (закінчується на парне число).

№	Додавання вершини	Черга
1	V1	V1
2	V2	V1,V2
3	V5	V1,V2,V5
4	V8	V1,V2,V5,V8
5	-	V2,V5,V8
6	V3	V2,V5,V8,V3
7	-	V2,V5,V8
8	V9	V5,V8,V3,V9
9	V4	V5,V8,V3,V9,V4
10	V6	V5,V8,V3,V9,V4,V6
11	-	V8,V3,V9,V4,V6
12	V7	V8,V3,V9,V4,V6,V7
13	-	V3,V9,V4,V6,V7
14	-	V9,V4,V6,V7
15	-	V4,V6,V7
16	-	V6,V7
17	-	V7
18	-	-
19		

```
#include <iostream>
#include <queue>
using namespace std;
int main() {
   int n, v;
   cin >> n >> v;
   int** matrix = new int* [n];
   for (int i = 0; i < n; i++)
     {
       matrix[i] = new int[n];
     }
   for (int i = 0; i < n; i++)
     for (int j = 0; j < n; j++)
       cin >> matrix[i][j];
   queue <int> plan;
   plan.push(--v);
   matrix[v][v] = 1;
   int counter = 1;
   while (!plan.empty()) {
     v = plan.front();
     plan.pop();
     for (int u = 0; u < n; u++) {
```

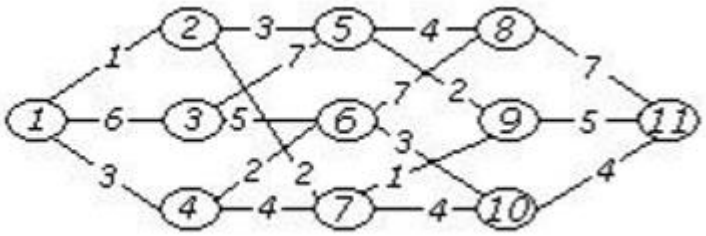
```
        if (matrix[v][u] and !matrix[u][u]) {
          plan.push(u);
          matrix[u][u] = 1;
          counter++;
          cout << plan.back()+1 << endl;
        }
      }
    }
}
```

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

Завдання № 5

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

12)

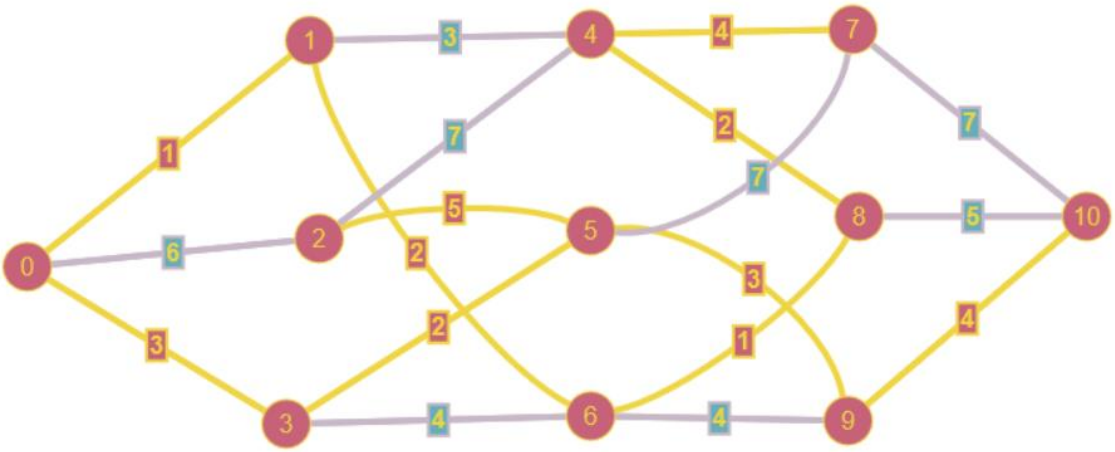


Алгоритм Прима:

```
#include <iostream>
#include <cstring>
using namespace std;

#define INF 9999999

// number of vertices in grapj
#define V 11

// create a 2d array of size 5x5
//for adjacency matrix to represent graph

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

```
  {0, 0, 0, 0, 4, 7, 0, 0, 0, 0, 7},
  {0, 0, 0, 0, 2, 0, 1, 0, 0, 0, 5},
  {0, 0, 0, 0, 0, 3, 4, 0, 0, 0, 4},
  {0, 0, 0, 0, 0, 0, 0, 7, 5, 4, 0},
};

int main() {

   int no_edge;


   int selected[V];


   memset(selected, false, sizeof(selected));

   no_edge = 0;



   selected[0] = true;

   int x;
   int y;


   cout << "Edge" << " : " << "Weight";
   cout << endl;
   while (no_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 << " - " << y << " :  " << G[x][y];
      cout << endl;
      selected[y] = true;
      no_edge++;
   }

   return 0;
}
```

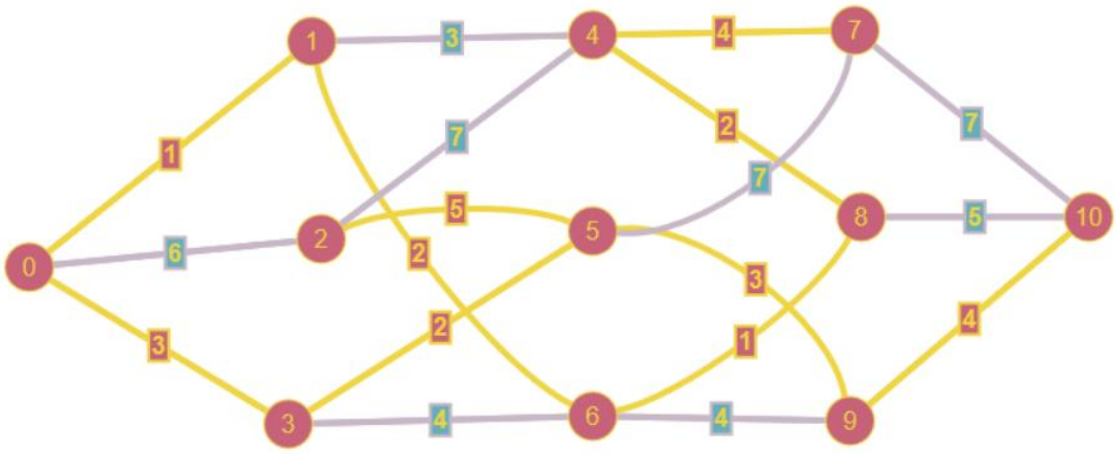
```
Edge : Weight
0 - 1 :  1
1 - 6 :  2
6 - 8 :  1
8 - 4 :  2
0 - 3 :  3
3 - 5 :  2
5 - 9 :  3
4 - 7 :  4
9 - 10 :  4
5 - 2 :  5
```



Алгоритм Крассака:

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

using namespace std;
int main()
{
   int m , n;
   cin >> m >> n;
   vector < pair < int, pair<int, int> > > g;
   for (int i = 0; i < m; i++)
   {
      int a1, a2, v;
      cin >> a1 >> a2 >> v;
      g[i] = make_pair(v, make_pair(a1, a2));
   }

   int cost = 0;
   vector < pair<int, int> > res;

   sort(g.begin(), g.end());
   vector<int> tree_id(n);
   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])
      {
        cost += l;
        res.push_back(make_pair(a, b));
        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;
      }
    }
}
```



Завдання № 6

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

12)

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

```
1.  #include <iostream>
```

```
#define min(x, y) x < y ? x : y
using namespace std;
const int inf = 1E9, NMAX = 16;
int n, i, j, k, m, temp, ans, d[NMAX][NMAX], t[1 << NMAX][NMAX];
bool get(int nmb, int x)
{
  return (x & (1 << nmb)) != 0;
}
int main()
{
  cin >> n;
  for (i = 0; i < n; ++i)
    for (j = 0; j < n; ++j) cin >> d[i][j];
  t[1][0] = 0; m = 1 << n;
  for (i = 1; i < m; i += 2)
    for (j = (i == 1) ? 1 : 0; j < n; ++j)
    {
      t[i][j] = inf;
      if (j > 0 && get(j, i))
      {
        temp = i ^ (1 << j);
        for (k = 0; k < n; ++k)
          if (get(k, i) && d[k][j] > 0) t[i][j] = min(t[i][j], t[temp][k] + d[k][j]);
      }
    }
  for (j = 1, ans = inf; j < n; ++j)
    if (d[j][0] > 0) ans = min(ans, t[m - 1][j] + d[j][0]);
  if (ans == inf) cout << -1; else cout << ans;
}
```

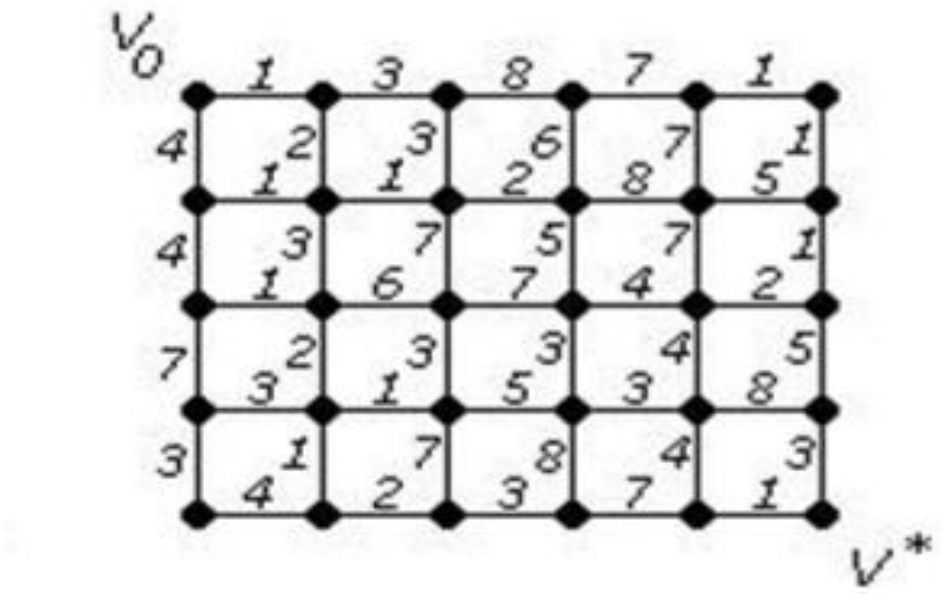
```
8
100000 5 6 5 4 4 5 5
5 100000 1 5 1 1 1 1
6 1 100000 1 1 3 2 1
5 5 1 100000 5 5 7 5
4 1 1 5 100000 3 2 5
4 1 3 5 3 100000 5 6
5 1 2 7 2 5 100000 1
5 1 1 5 5 6 1 100000
16
```

Відповідь 16.

Завдання № 7

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

12)



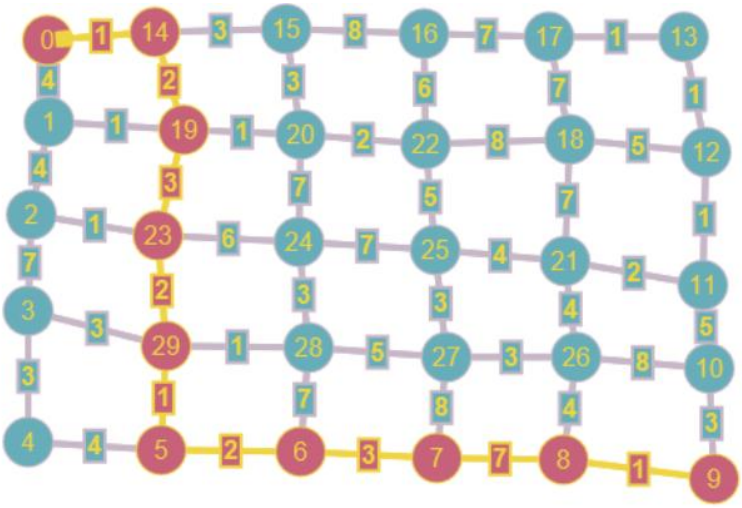

```
#include <cmath>
#include <iostream>

using namespace std;


void output(int i, int* par, int k);


const int N = 30;


int matrix[N][N] = {
0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
4, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 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, 1, 0, 0, 0, 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,
3,
0, 0, 0, 3, 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, 0, 0, 0, 4, 0, 2, 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, 0, 2, 0, 3, 0, 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, 0, 3, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0,
0,
0, 0, 0, 0, 0, 0, 0, 7, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 0, 0, 0, 3, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0,
0,
```

```
0, 0, 0, 0, 0, 0, 0, 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, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 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, 0, 3, 0, 8, 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, 8, 0, 7, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 7, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 7, 0, 0, 0, 7, 8, 0, 0, 0, 0, 0, 0,
0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 1, 0, 0, 2, 0, 7, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 4, 4, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 8, 0, 2, 0, 0, 0, 0, 5, 0, 0, 0,
0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 6, 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, 7, 0, 0, 6, 0, 7, 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, 4, 5, 0, 7, 0, 0, 3, 0,
0,
0, 0, 0, 0, 0, 0, 0, 4, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 3, 0,
0,
0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 5,
0,
0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 5, 0,
1,
0, 0, 0, 3, 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,
};

int  cost[N][N] = {
0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
4, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
0, 4, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 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,
3,
0, 0, 0, 3, 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, 0, 0, 0, 4, 0, 2, 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, 0, 2, 0, 3, 0, 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, 0, 3, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0,
0,
0, 0, 0, 0, 0, 0, 0, 7, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 0, 0, 0, 3, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 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, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 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, 3, 0, 8, 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, 8, 0, 7, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 7, 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, 5, 0, 0, 0, 0, 7, 0, 0, 0, 7, 8, 0, 0, 0, 0, 0, 0,
0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 0, 3, 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, 1, 0, 0, 2, 0, 7, 0, 0, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 4, 4, 0, 0,
0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 8, 0, 2, 0, 0, 0, 0, 5, 0, 0, 0,
0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 6, 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, 7, 0, 0, 6, 0, 7, 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, 4, 5, 0, 7, 0, 0, 3, 0,
0,
0, 0, 0, 0, 0, 0, 0, 4, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 3, 0,
0,
0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 5,
0,
0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 5, 0,
1,
0, 0, 0, 3, 0, 1, 0, 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,


};


int  dist[N];

int  parent[N];
void deicstra(int start, int end)
{

    bool in_tree[N] = { false };

    for (int i = 0; i < N; i++)
        dist[i] = INT_MAX;


    dist[start] = 0;

    int cur = start;


    while (!in_tree[cur])
    {
        in_tree[cur] = true;

        for (int i = 0; i < N; i++)
        {
            if (matrix[cur][i] != 0)
            {

                int d = dist[cur] + cost[cur][i];

                if (d < dist[i])
                {
                    dist[i] = d;
                    parent[i] = cur;
                }
            }
```

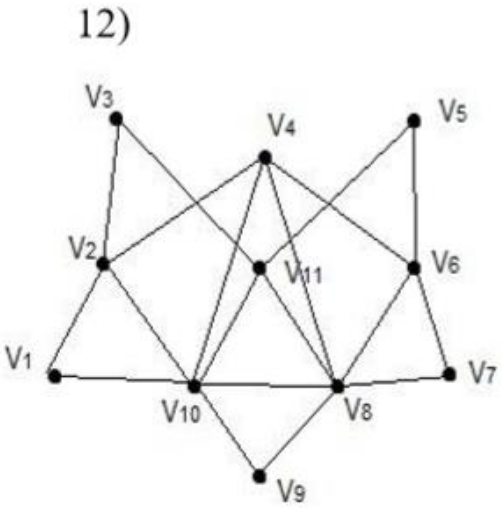
```
        }


        int min_dist = INT_MAX;
        for (int i = 0; i < N; i++)
        {
            if (!in_tree[i] && dist[i] < min_dist)
            {
                cur = i;
                min_dist = dist[i];
            }
        }
    }
    output(start, parent, end);
    cout << end << "\nWeight: " << dist[end];

}


void output(int k, int* par, int i)
{
    if (k == i)
    {
        return;
    }
    else
    {
        output(k, par, par[i]);
        cout << par[i] << " --> ";
        return;
    }
}

int main()
{
    deicstra(0 , 9);


}
```

```
0 --> 14 --> 19 --> 23 --> 29 --> 5 --> 6 --> 7 --> 8 --> 9
Weight: 22
```



Завдання № 8

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



Эйлеровий цикл:

```
#include<iostream>
#include<vector>
#define NODE 11
using namespace std;
int graph[NODE][NODE] = {
      {0,1,0,0,0,0,0,0,0,1,0},
      {1,0,1,1,0,0,0,0,0,1,0},
      {0,1,0,0,0,0,0,0,0,0,1},
      {0,1,0,0,0,1,0,1,0,1,0},
      {0,0,0,0,0,1,0,0,0,0,1},
      {0,0,0,1,1,0,1,1,0,0,0},
      {0,0,0,0,0,1,0,1,0,0,0},
      {0,0,0,1,0,1,1,0,1,1,1},
      {0,0,0,0,0,0,0,1,0,1,0},
      {1,1,0,1,0,0,0,1,1,0,1},
      {0,0,1,0,1,0,0,1,0,1,0},
};
int tempGraph[NODE][NODE];
int findStartVert() {
   for (int i = 1; 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 isBridge(int u, int v) {
   int deg = 0;
   for (int i = 0; i < NODE; i++)
      if (tempGraph[v][i])
         deg++;
   if (deg > 1) {
```

```
37.       return false;
38.    }
39.    return true;
40. }
41. int edgeCount() {
42.    int count = 0;
43.    for (int i = 0; i < NODE; i++)
44.       for (int j = i; j < NODE; j++)
45.          if (tempGraph[i][j])
46.             count++;
47.    return count;
48. }
49. void fleuryAlgorithm(int start) {
50.    static int edge = edgeCount();
51.    for (int v = 0; v < NODE; v++) {
52.       if (tempGraph[start][v]) {
53.          if (edge <= 1 || !isBridge(start, v)) {
54.             cout << start + 1 << "--" << v + 1 << " ";
55.             tempGraph[start][v] = tempGraph[v][start] = 0;
56.             edge--;
57.             fleuryAlgorithm(v);
58.          }
59.       }
60.    }
61. }
62. int main() {
63.    for (int i = 0; i < NODE; i++)
64.       for (int j = 0; j < NODE; j++)
65.          tempGraph[i][j] = graph[i][j];
66.    cout << "Euler Path Or Circuit: ";
67.    fleuryAlgorithm(findStartVert());
68. }
```

```
Euler Path Or Circuit: 1--2 2--3 3--11 11--5 5--6 6--4 4--2 2--10 10--4 4--8 8--6 6--7 7--8 8--9 9--10 10--8 8--11 11--1
0 10--1
```

1—2, 2—3, 3—11, 11—5, 5—6, 6—4, 4—2, 2—10, 10—4, 4—8, 8—6, 6—7, 7—8, 8—9, 9—10, 10—8, 8—11, 11—10, 10—1.

Завдання №9

Спростити формули (привести їх до скороченої ДНФ).

12. $\bar{x}y \vee x\bar{y}\bar{z}$

Відповідь: $\bar{x}y \vee x\bar{y}\bar{z}$.