**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**Кафедра систем штучного інтелекту**

**Розрахунково-графічна робота**

з дисципліни «Дискретна математика»

**Виконав:**

студент групи КН-112

Тиський Святослав

**Викладач:**

Мельникова Н.І.

Львів-2019

**Варіант №8**

**Розрахунково-графічна робота.**

**Завдання № 1:** Виконати наступні операції над графами:

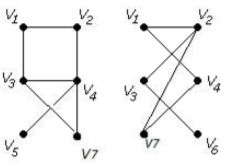
1) знайти доповнення до першого графу   
2) об’єднання графів

3) кільцеву сумму G1 та G2 (G1+G2)

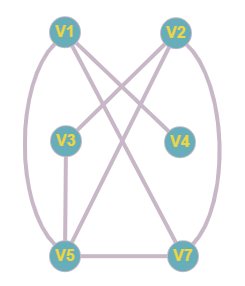
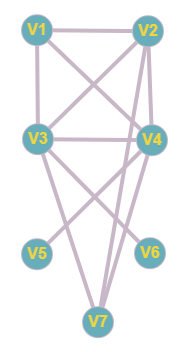
4) розмножити вершину у другому графі

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

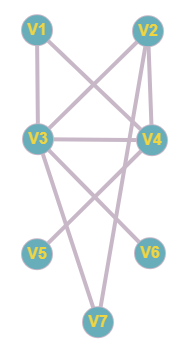
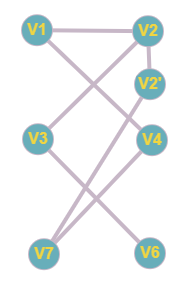
6) добуток графів.



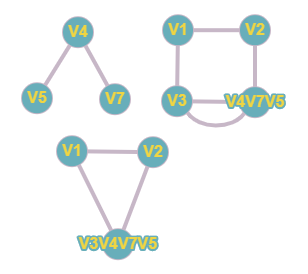
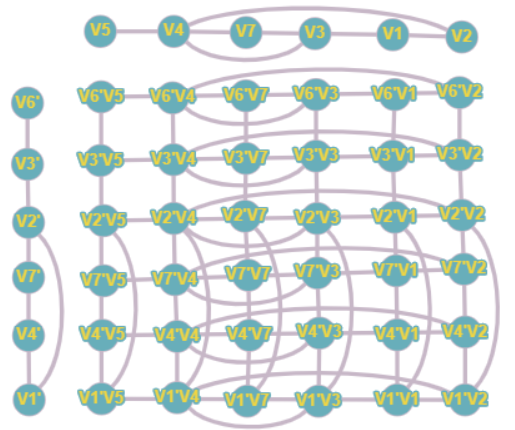
**Розв’язок завдання №1.1: Розв’язок завдання №1.2:**

**Розв’язок завдання №1.3: Розв’язок завдання №1.4:**

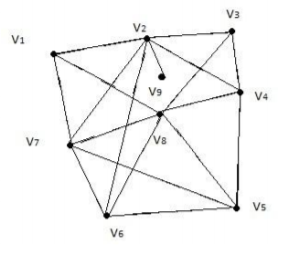
 

**Розв’язок завдання №1.5: Розв’язок задання №1.6:**

 ****

**Завдання №2:**

Скласти таблицю суміжності для орграфа.

****

**Розв’язок завдання №2:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **V1** | **V2** | **V3** | **V4** | **V5** | **V6** | **V7** | **V8** | **V9** |
| **V1** | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| **V2** | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| **V3** | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| **V4** | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| **V5** | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| **V6** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| **V7** | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| **V8** | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| **V9** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Завдання №3:**

Для графа з другого завдання знайти діаметр.

**Розв’язок завдання №3**

Діаметр = 3

**Завдання №4:**

Для графа з другого завдання виконати обхід дерева вшир

**Розв’язок завдання №4:**

|  |  |  |
| --- | --- | --- |
| **Вершина** | **№** | **Черга** |
| V1 | 1 | {V1} |
| V2 | 2 | {V1,V2} |
| V7 | 3 | {V1,V2,V7} |
| V8 | 4 | {V1,V2,V7,V8} |
| - | - | {V2,V7,V8} |
| V3 | 5 | {V2,V7,V8,V3} |
| V4 | 6 | {V2,V7,V8,V3,V4} |
| V6 | 7 | {V2,V7,V8,V3,V4,V6} |
| V9 | 8 | {V2,V7,V8,V3,V4,V6,V9} |
| - | - | {V7,V8,V3,V4,V6,V9} |
| V5 | 9 | {V7,V8,V3,V4,V6,V9,V5} |
| - | - | {V8,V3,V4,V6,V9,V5} |
| - | - | {V3,V4,V6,V9,V5} |
| - | - | {V4,V6,V9,V5} |
| - | - | {V6,V9,V5} |
| - | - | {V9,V5} |
| - | - | {V5} |
| - | - |  |

#include <iostream>

#include <queue>

using namespace std;

bool check(int \*Check,int j,int N) {//перевіряємо чи не було вже такої вершини в черзі

for (int i = 0; i < N; i++) {

if (Check[i] == j) {

return false;

}

}

return true;

}

int main()

{

int Start, N,k=1;

int\*\* Graf;

queue <int> qq;

cin >> N >> Start;

Graf = new int\* [N];

int\* Check = new int[4\*N];//масив з елементами для перевірки(4N бо тут багато зайвих елементів)

qq.push(Start-1);//наш стек починається із заданої вершини

Check[0] = Start-1;

for (int i = 0; i < N; i++) {

Graf[i] = new int[N];

}

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

cin >> Graf[i][j];

}

}

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (Graf[qq.front()][j]) {

if (check(Check, j, 4 \* N)) {

qq.push(j);//якщо вершину ще не проходили додаємо її в чергу

}

Check[k] = j;

k++;

}

}

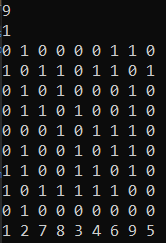
cout << ++qq.front()<<" ";

qq.pop();//після того як закінчились суміжні вершини видаляєм перший елемент черги

}

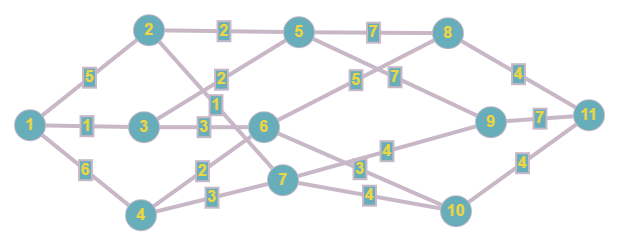
return 0;

}

****

**Завдання №5:**

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

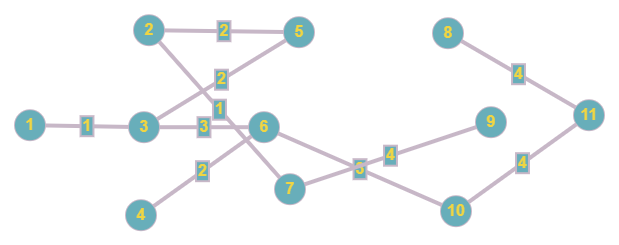


**Розв’язок завдання №5:**

Краскал:

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

E:{(1,3),(2,7),(2,5),(3,5),(4,6),(3,6),(6,10),(7,9),(8,11),(10,11)}.



#include <iostream>

using namespace std;

int main()

{

int rebro[18][4] = {

{1,2,5,0},//1ver 2ver masa

{1,3,1,0},

{1,4,6,0},

{2,5,2,0},

{2,7,1,0},

{3,5,2,0},

{3,6,3,0},

{4,6,2,0},

{4,7,3,0},

{5,8,7,0},

{5,9,7,0},

{6,8,5,0},

{6,10,3,0},

{7,9,4,0},

{7,10,4,0},

{8,11,4,0},

{9,11,7,0},

{10,11,4,0}

};

int top[11][2] = {

{1,0},{2,0},{3,0},{4,0},{5,0},{6,0},{7,0},{8,0},{9,0},{10,0},{11,0}

};

for (int i = 0; i < 18; i++) {

for (int k = 0; k < 17-i; k++) {

if (rebro[k][2] > rebro[k + 1][2]) {

for (int j = 0,a=0; j < 4; j++) {

a = rebro[k + 1][j];

rebro[k + 1][j] = rebro[k][j];

rebro[k][j] = a;

}

}

}

}

int count = 1;

for (int i = 0; i < 18; i++) {

if (top[rebro[i][0] - 1][1] != top[rebro[i][1] - 1][1]) {

rebro[i][3] = 1;

if (top[rebro[i][0] - 1][1] == 0 && top[rebro[i][1] - 1][1]!=0) {

top[rebro[i][0] - 1][1] = top[rebro[i][1] - 1][1];

}

if (top[rebro[i][0] - 1][1] != 0 && top[rebro[i][1] - 1][1] == 0) {

top[rebro[i][1] - 1][1] = top[rebro[i][0] - 1][1];

}

if (top[rebro[i][0] - 1][1] != 0 && top[rebro[i][1] - 1][1] != 0) {

for (int j = 0; j < 11; j++) {

if (top[j][1] == top[rebro[i][1] - 1][1]) {

top[j][1] = top[rebro[i][0] - 1][1];

}

}

}

}

if (top[rebro[i][0] - 1][1] == 0 && top[rebro[i][1] - 1][1] == 0) {

rebro[i][3] = 1;

top[rebro[i][0] - 1][1] = count;

top[rebro[i][1] - 1][1] = count;

count++;

}

}

for (int i = 0; i < 18; i++) {

if (rebro[i][3] == 1) {

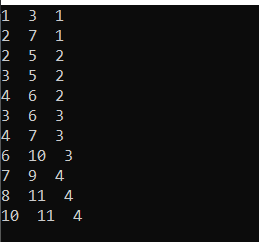
cout << rebro[i][0] << " " << rebro[i][1] << " " << rebro[i][2]<<endl;

}

}

return 0;

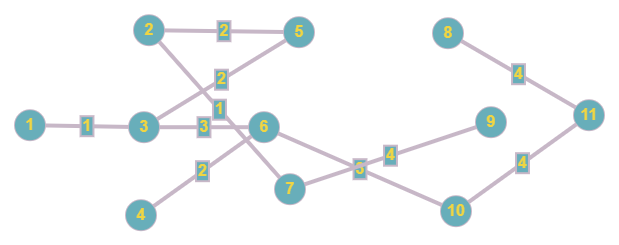
}



Прима:

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

E:{(1,3),(3,5),(5,2),(2,7),(3,6),(6,4),(6,10),(7,9),(10,11),(11,8)}.



#include <iostream>

using namespace std;

int main()

{

int v, count = 0, min = 0, k,t;

bool check = false;

cin >> v;

int\* tops = new int[v];

int\*\* matrix = new int\* [v];

for (int i = 0; i < v; i++) {

matrix[i] = new int[v];

}

int\*\* rebra = new int\* [v - 1];

for (int i = 0; i < v - 1; i++) {

rebra[i] = new int[2];

}

for (int i = 0; i < v; i++) {

for (int j = 0; j < v; j++) {

cin >> matrix[i][j];

}

}

tops[count] = 1;

count++;

for (int i = 0; count < v; i++) {

for (int j = 0; j < count; j++) {

for (int a = 0; a < v; a++) {

for (int m = 0; m < count; m++) {

if (tops[m] == a + 1) {

check = true;

}

}

if (check) { check = false; continue; }

if (min == 0 && matrix[tops[j] - 1][a] > 0) {

min = matrix[tops[j] - 1][a];

k = rebra[count - 1][0] = tops[j]; t = rebra[count - 1][1] = a + 1;

continue;

}

if (matrix[tops[j] - 1][a] > 0 && matrix[tops[j] - 1][a] < min) {

min = matrix[tops[j] - 1][a];

k = rebra[count - 1][0] = tops[j]; t = rebra[count - 1][1] = a + 1;

}

}

}

matrix[k - 1][t - 1] = 0; matrix[t - 1][k - 1] = 0;

tops[count] = t;

count++;

min = 0;

}

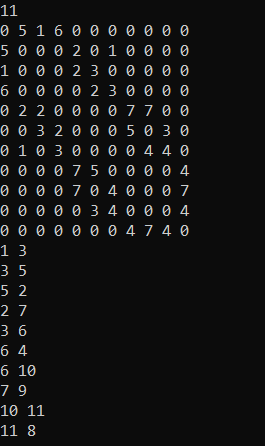
for (int j = 0; j < v - 1; j++) {

cout << rebra[j][0] <<" "<< rebra[j][1] <<endl;

}

return 0;

}



**Завдання №6:**

Розв’язати задачу комiвояжера для повного 8-ми вершинного графа методом «iди у найближчий», матриця вагiв якого має вигляд:

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

**Для вершини 1:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **2** | **3** | **4** | **5** | **6** | **17** | **8** |
| **2** |  | 6 | 1 | 5 | 1 | 1 | 2 |
| **3** | 6 |  | 5 | 1 | 7 | 5 | 5 |
| **4** | 1 | 5 |  | 3 | 3 | 2 | 3 |
| **5** | 5 | 1 | 3 |  | 2 | 2 | 3 |
| **6** | 1 | 7 | 3 | 2 |  | 5 | 7 |
| **17** | 1 | 5 | 2 | 2 | 5 |  | 5 |
| **8** | 2 | 5 | 3 | 3 | 7 | 5 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **172** | **3** | **4** | **5** | **6** | **8** |
| **172** |  | 6 | 1 | 5 | 1 | 2 |
| **3** | 6 |  | 5 | 1 | 7 | 5 |
| **4** | 1 | 5 |  | 3 | 3 | 3 |
| **5** | 5 | 1 | 3 |  | 2 | 3 |
| **6** | 1 | 7 | 3 | 2 |  | 7 |
| **8** | 2 | 5 | 3 | 3 | 7 |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **3** | **1724** | **5** | **6** | **8** |
| **3** |  | 5 | 1 | 7 | 5 |
| **1724** | 5 |  | 3 | 3 | 3 |
| **5** | 1 | 3 |  | 2 | 3 |
| **6** | 7 | 3 | 2 |  | 7 |
| **8** | 5 | 3 | 3 | 7 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **3** | **17245** | **6** | **8** |
| **3** |  | 1 | 7 | 5 |
| **17245** | 1 |  | 2 | 3 |
| **6** | 7 | 2 |  | 7 |
| **8** | 5 | 3 | 7 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **172453** | **6** | **8** |
| **172453** |  | 7 | 5 |
| **6** | 7 |  | 7 |
| **8** | 5 | 7 |  |

|  |  |  |
| --- | --- | --- |
|  | **6** | **1724538** |
| **6** |  | 7 |
| **1724538** | 7 |  |

2+1+1+1+1+5+7=18

#include<iostream>

#include<vector>

using namespace std;

int counter = 0, Inf = 9999;

bool check(vector<int> q, int Node);

int F\_Min(vector<int>\* q, int\*\* arr, int n, int i);

void Find(vector<int>\* q, int\*\* arr, int n, int pos, vector<int>\* qq);

int main()

{

int n;

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> q;

vector<int> qq;

cout << endl;

for (int i = 0; i < n; i++) {

q.clear();

q.push\_back(i);

Find(&q, arr, n, i, &qq);

}

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

if (i != 0 && i % (n + 2) == 0)

cout << " {" << qq[i - 1] << "}" << endl;

else

cout << qq[i - 1] + 1 << " ";

}

return 0;

}

bool check(vector<int> q, int Node) {

for (auto i = q.begin(); i != q.end(); i++)

if (\*i == Node)return false;

return true;

}

int F\_Min(vector<int>\* q, 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((\*q), j))min = arr[i][j];

}

return min;

}

void Find(vector<int>\* q, int\*\* arr, int n, int pos, vector<int>\* qq) {

int min;

for (int i = pos, k = 0; k < 1; i++, k++) {

min = F\_Min(q, arr, n, i);

for (int j = 0; j < n; j++) {

if (arr[i][j] == min && check((\*q), j)) {

(\*q).push\_back(j);

Find(q, arr, n, j, qq);

}

}

if (q->size() == n) {

(\*q).push\_back((\*q)[0]);

counter = 0;

for (int l = 1; l <= n; l++) {

counter += arr[(\*q)[l - 1]][(\*q)[l]];

}

if (Inf == counter) {

for (int b = 0; b <= n; b++) {

(\*qq).push\_back((\*q)[b]);

}

(\*qq).push\_back(counter);

}

else if (Inf > counter) {

(\*qq).clear();

for (int b = 0; b <= n; b++) {

(\*qq).push\_back((\*q)[b]);

}

(\*qq).push\_back(counter);

Inf = counter;

}

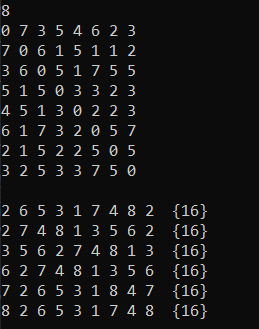
q->pop\_back();

}

}

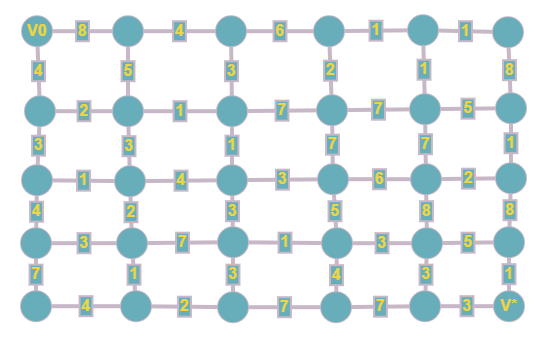
q->pop\_back();

}

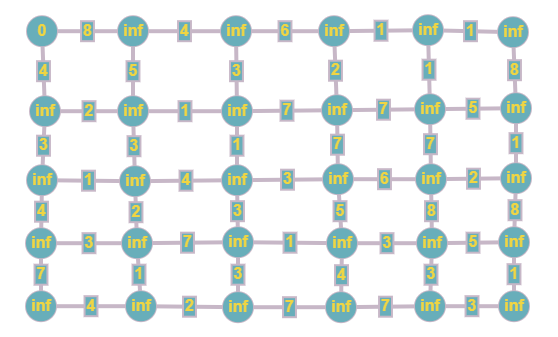
****

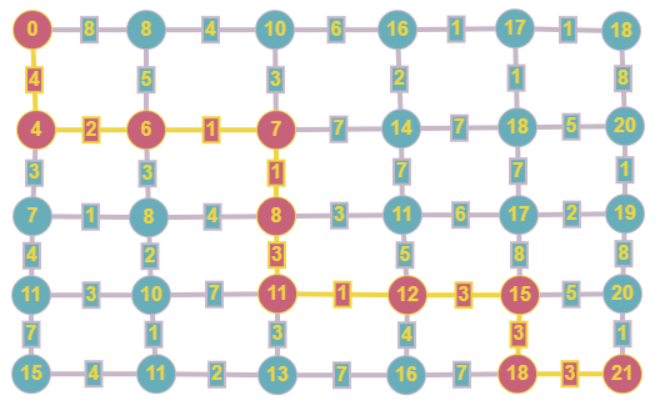
**Завдання №7:**

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



**Розв’язок завдання №7:**





#include <iostream>

#define inf 9999

using namespace std;

int min\_top(int\*\* arr, int v) {

int m = 0;

for (int i = 0; i < v; i++) {

if (arr[i][1]) {

m = i; break;

}

}

for (int i = 1; i < v; i++) {

if (arr[m][0] >= arr[i][0] && arr[i][1] == 1) {

m = i;

}

}

return m;

}

int main()

{

setlocale(LC\_ALL, "Ukrainian");

int a, b, c;

int v = 0;

cout << "Кiлькiсть вершин графа : ";

cin >> v;

int\*\* graph = new int\* [v];

for (int j = 0; j < v; j++) {

graph[j] = new int[v];

}

cout << "Введiть матрицю сумiжностi: "<<endl;

for (int a = 0; a < v; a++) {

for (int j = 0; j < v; j++) {

cin>>graph[a][j];

}

}

int p;

int\*\* tops = new int\* [v];

for (int j = 0; j < v; j++) {

tops[j] = new int[2];

}

int\* tops\_path = new int[v];

cout << "Вихiдна вершина: ";

cin >> p;

for (int i = 0; i < v; i++) {

if (i == p - 1) {

tops[i][0] = 0;

tops[i][1] = 1;

}

else {

tops[i][0] = inf;

tops[i][1] = 1;

}

}

tops\_path[p - 1] = 0;

int m;

for (int i = 0; i < v; i++) {

m = min\_top(tops, v);

for (int j = 0; j < v; j++) {

if (graph[m][j]) {

if (tops[j][0] > tops[m][0] + (graph[m][j])) {

tops[j][0] = tops[m][0] + (graph[m][j]);

tops\_path[j] = m;

}

}

}

tops[m][1] = 0;

}

////шлях

cout << "Введiть потрiбну вершину: ";

int k; cin >> k;

cout << "Мiнiмальний шлях: ";

cout << tops[k - 1][0];

cout << endl << k << " <-- ";

k--;

for (int a = 0; tops\_path[k] != p - 1; a++) {

cout << tops\_path[k] + 1 << " <-- ";

k = tops\_path[k];

}

cout << p << endl;

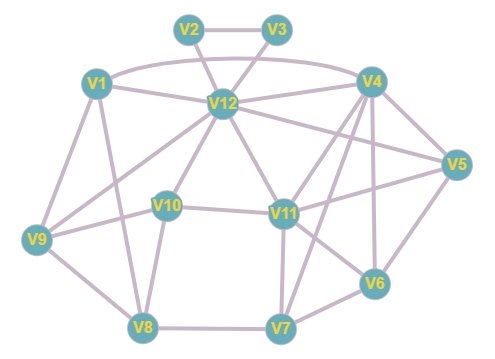
return 0;

}

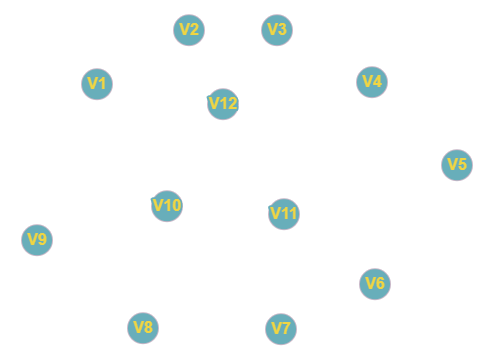
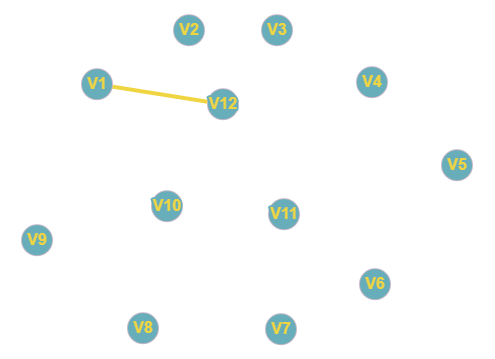
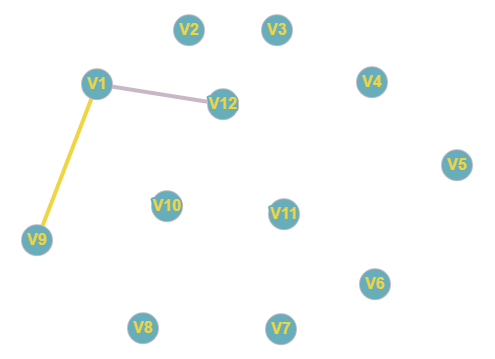
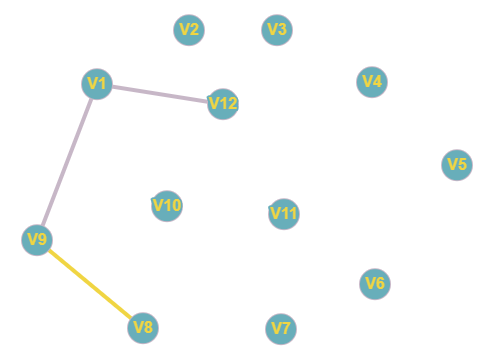
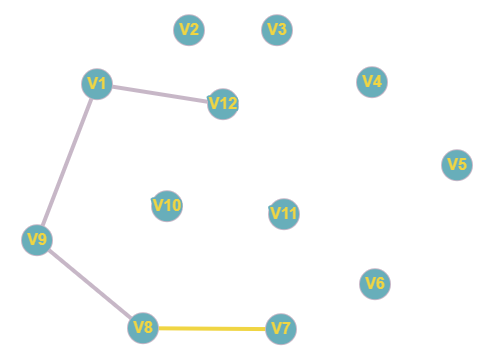
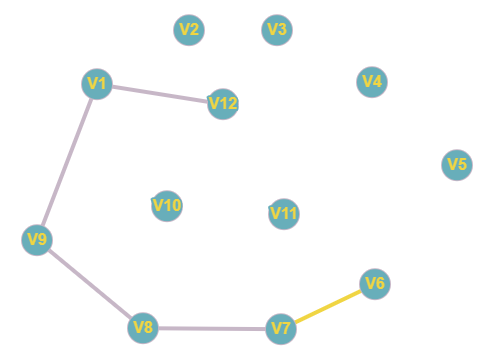
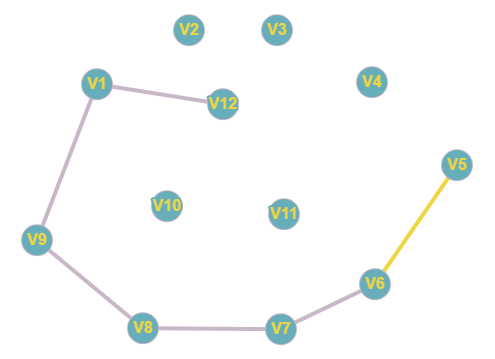
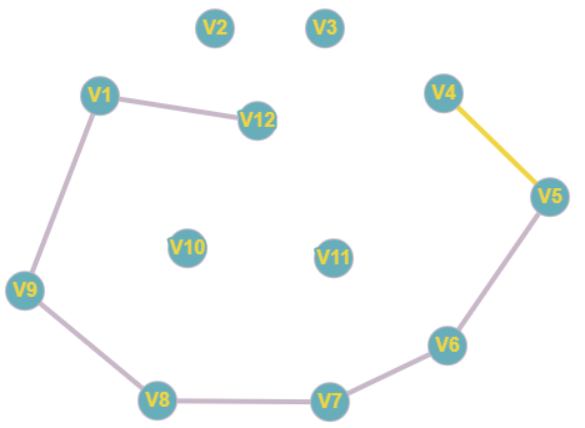
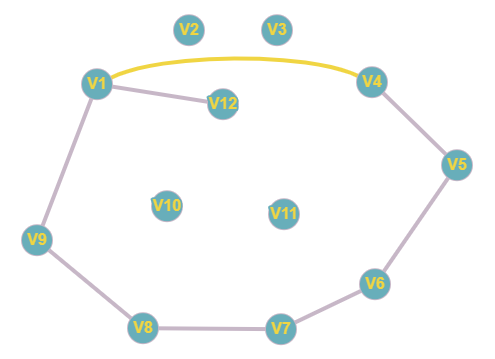
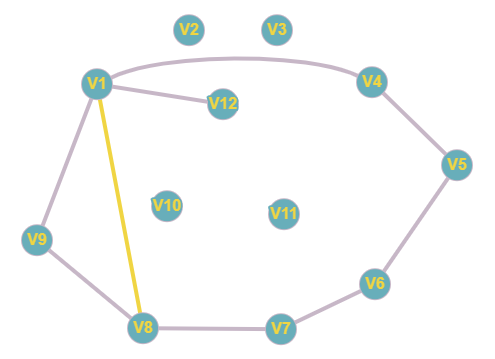
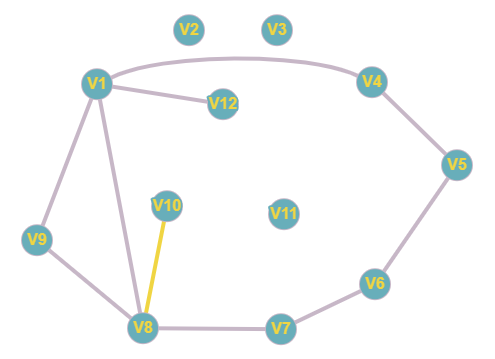
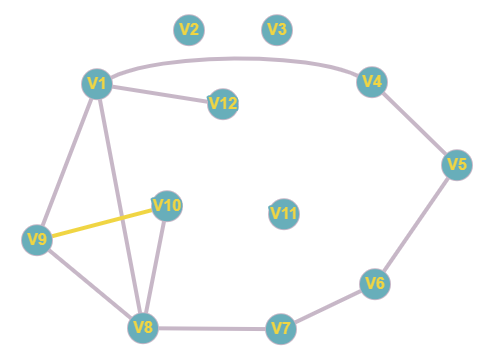
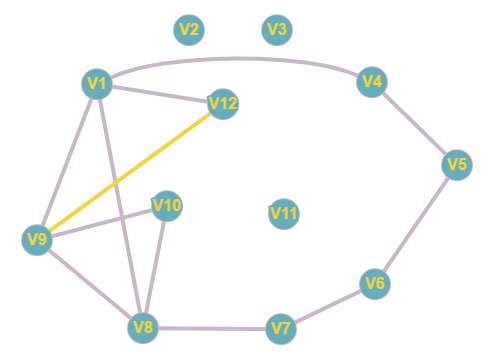
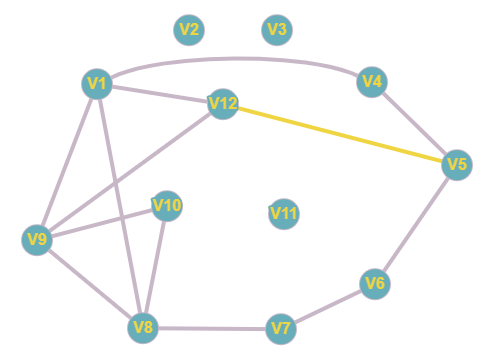
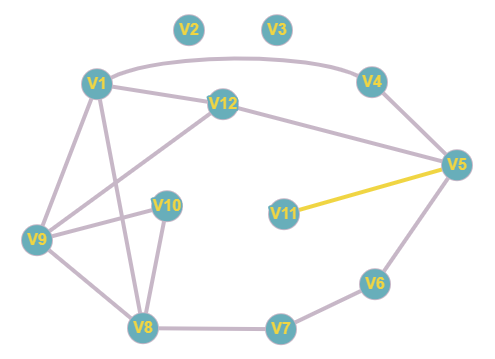
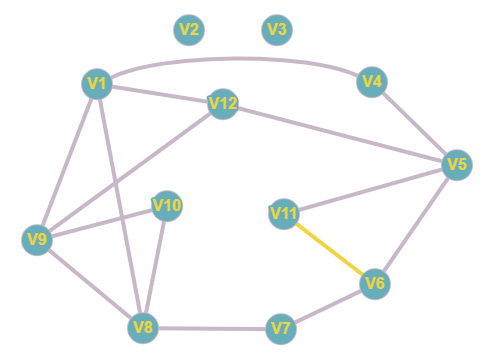
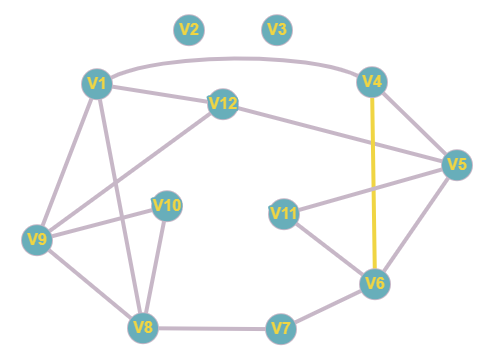
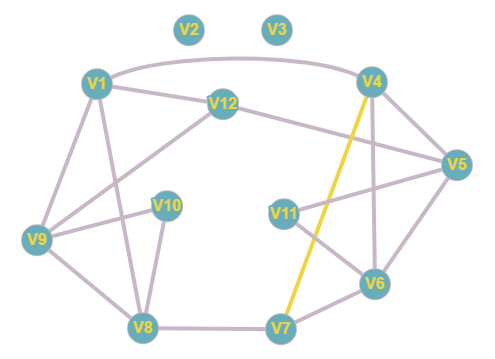
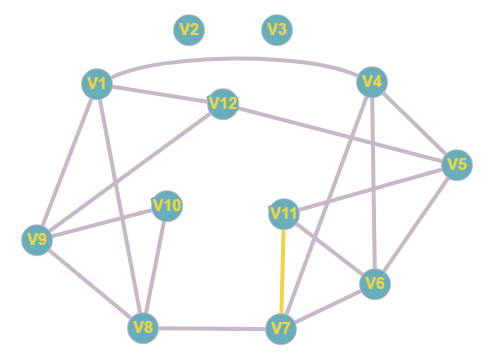
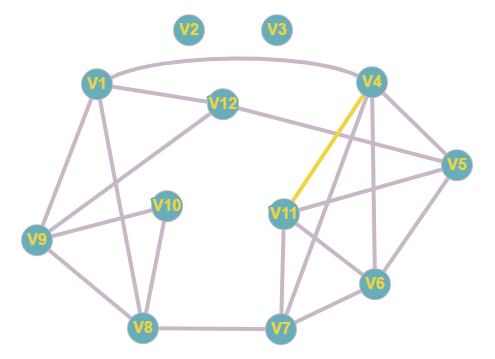
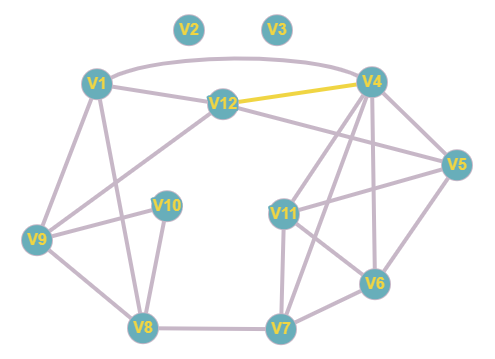
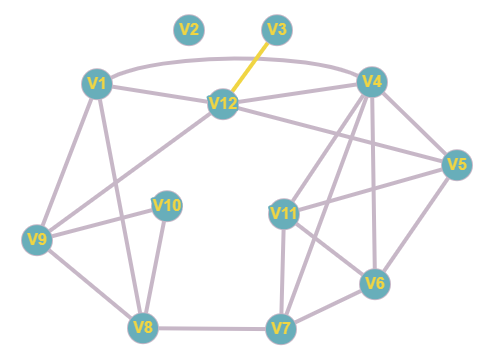
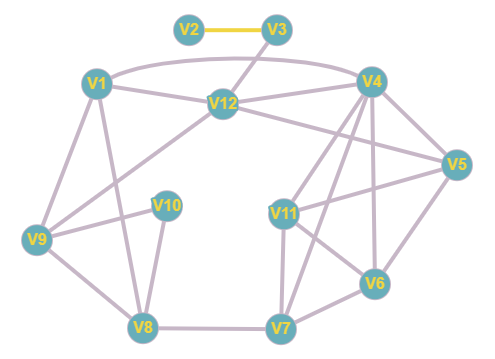
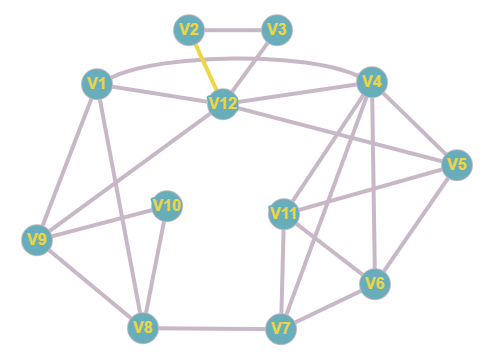
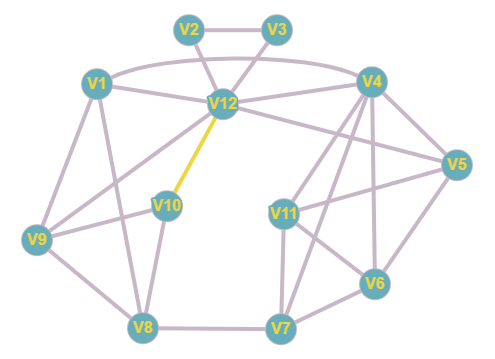
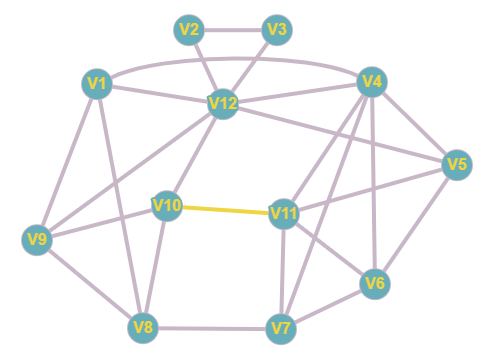
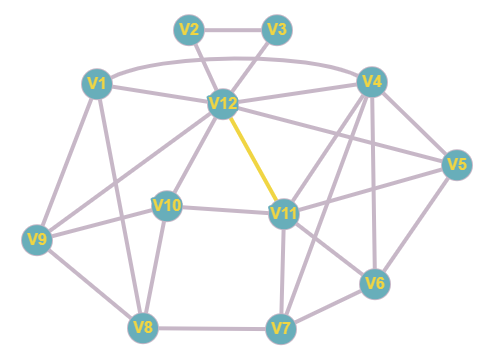
****

**Завдання №8:**

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



**Розв’язок завдання №8.а:**



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

#include<iostream>

#include<vector>

using namespace std;

void Search(int v, vector < vector<int> >\* G, int N)

{

int i;

for (i = 0; i < N; i++) {

if ((\*G)[v][i])

{

(\*G)[v][i] = (\*G)[i][v] = 0;

Search(i, G, N);

}

}

cout << v + 1 << " => ";

}

int main()

{

int N = 0;

cin >> N;

vector < vector<int> > G(N, vector<int>(N));

for (int i = 0; i < N; ++i) {

for (int j = 0; j < N; ++j) {

cin >> G[i][j];

}

}

int count, p, q, sum;

count = 1;

for (p = 0; p < N; p++){

sum = 0;

for (q = 0; q < N; q++){

sum += G[p][q];

}

if (sum % 2) count = 0;

}

cout << endl;

if (count)

Search(0, &G, N);

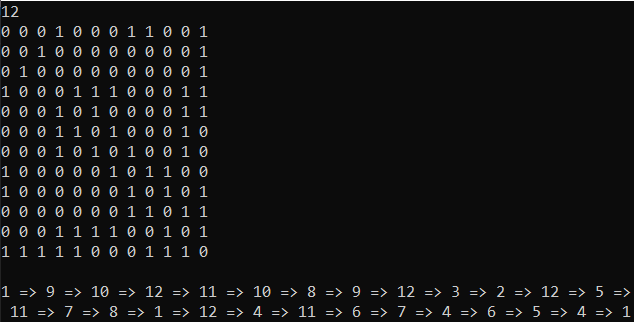
else

cout << "vsio fignia davai po novoi\n";

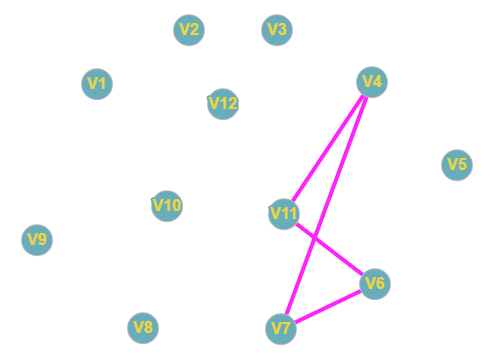
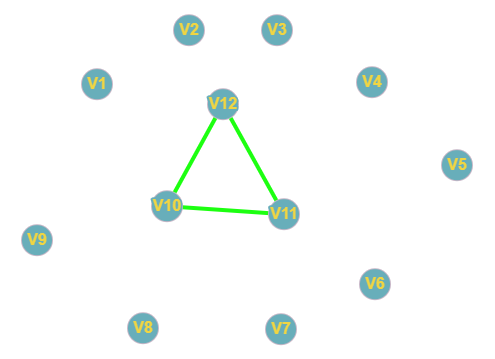
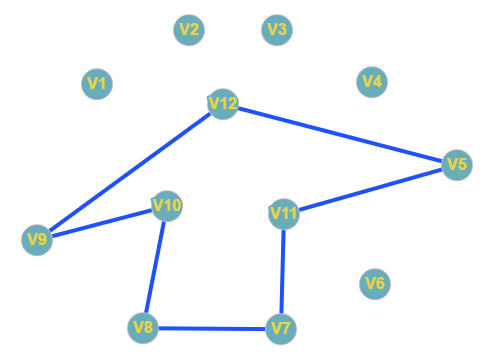
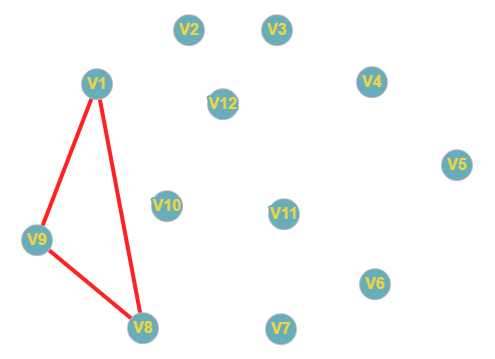
cout << endl;

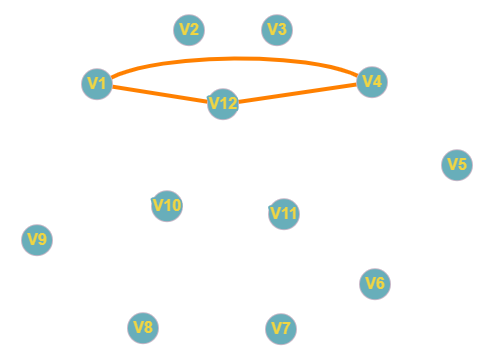
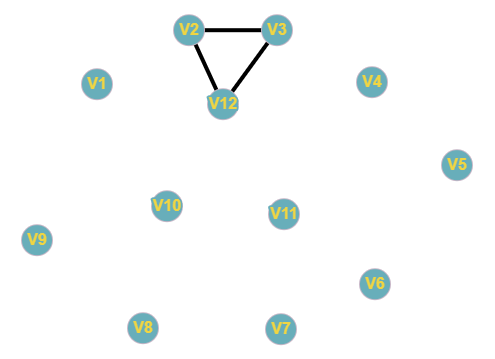
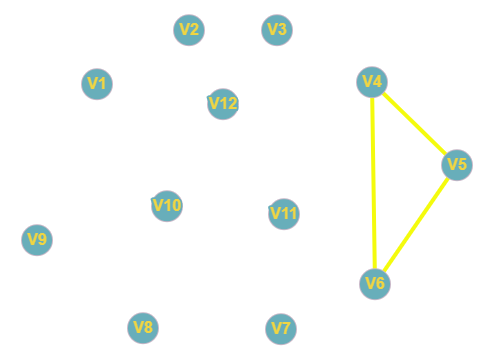
return 0;

}

****

**Розв’язок завдання №8.б**





Червоний V1=>V8=>V9=>V1

Синій V9=>V12=>V5=>V11=>V7=>V8=>V10=>V9

Зелений V12=>V11=>V10=>V12

Рожевий V11=>V4=>V7=>V6=>V11

Жовтий V4=>V5=>V6=>V4

Чорний V2=>V3=>V12=>V2

Оранжевий V1=>V12=>V4=>V1

#include <iostream>

#include <vector>

using namespace std;

vector<int> Vcon;

int Inf = 999;

bool check(vector<int> V, int pork) {

for (auto i = V.begin(); i != V.end(); i++) {

if (\*i == pork)return false;

}

return true;

}

void Find(vector<int>\* V, int\*\* arr, int n, int pos, int start\_pork) {

for (int i = pos, k = 0; k < 1; i++, k++) {

for (int j = 0; j < n; j++)

if (arr[i][j] == 1 && check((\*V), j)) {

if (j == start\_pork && (\*V).size() > 2) {

if (Inf > V->size()) {

Vcon.clear();

Vcon.push\_back(start\_pork + 1);

for (auto it = (\*V).begin(); it != (\*V).end();

it++)

Vcon.push\_back(\*it + 1);

Vcon.push\_back(start\_pork + 1);

Inf = V->size();

break;

}

}

else {

(\*V).push\_back(j);

Find(V, arr, n, j, start\_pork);

}

}

}

if (V->size() != 0)

V->pop\_back();

}

int main() {

int n;

cout << "Enter number of porks: ";

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> WAS;

cout << endl;

int count, p, q, sum;

count = 1;

for (p = 0; p < n; p++)

{

sum = 0;

for (q = 0; q < n; q++)

{

sum += arr[p][q];

}

if (sum % 2) count = 0;

}

cout << endl;

if (count) {

for (int j = 0; j < n; j++) {

Inf = 999;

Find(&V, arr, n, j, j);

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

cout << Vcon[i - 1] << " ";

}

cout << endl;

Vcon.clear();

}

}

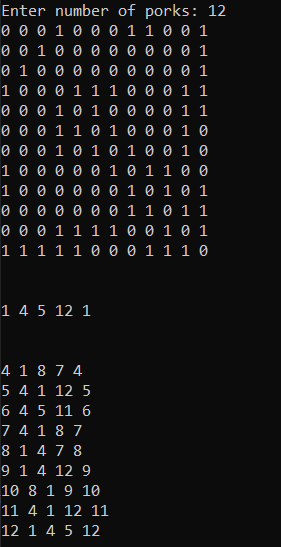
else

cout << "tikai z sela\n";

cout << endl;

return 0;

}



**Завдання №9**

Спростити формули (привести їх до скороченої ДНФ).

