МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Розрахунково-графічні завдання

з дисципліни
«Дискретна математика»

Виконала:
Студентка групи КН-114
Огорілко Вікторія
Викладач:
Мельникова Н.І.

Львів – 2019р.

Індивідуальні завдання

Варіант №20

Завдання :

1. Виконати наступні операції над графами:

1) знайти доповнення до першого графу,

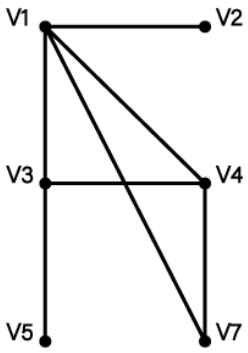
2) об’єднання графів,

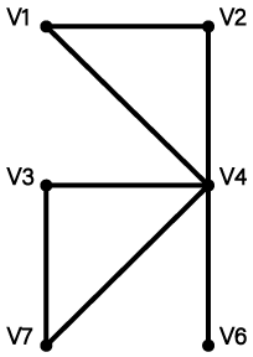
3) кільцеву суму G1 та G2 (G1⊕G2),

4) розщепити вершину у другому графі,

5) виділити підграф A, що складається з 3-х вершин в G1 і знайти стягнення A в G1 (G1\ A),

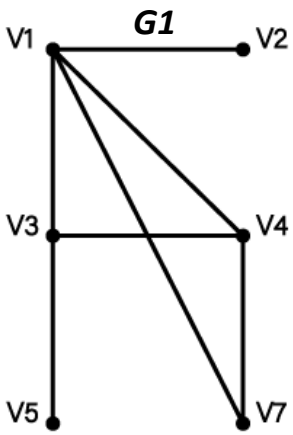
6) добуток графів $G1 \times G2$.





Розв’язування :

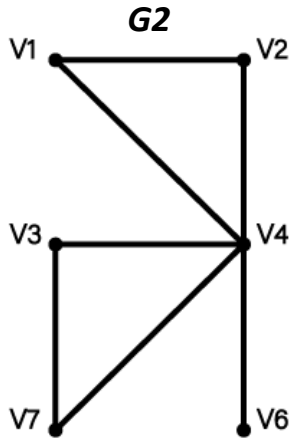
1.



$$\boldsymbol{X_1 = \{V1, V2, V3, V4, V5, V7\}}$$

$$\boldsymbol{W_1 = \{(V1, V2), (V1, V3), (V1, V4), (V1, V7),}$$

$$\boldsymbol{(V3, V4), (V3, V5), (V4, V7)\}}$$

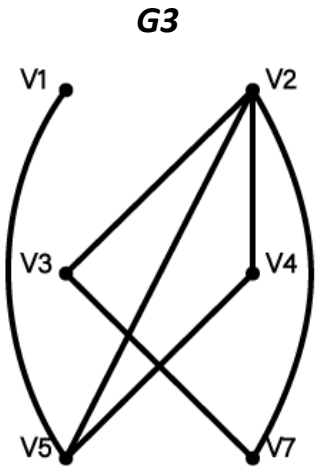


$$X_2 = \{V1, V2, V3, V4, V6, V7\}$$

$$W_2 = \{(V1, V2), (V1, V4), (V2, V4),$$

$$(V3, V4), (V3, V7), (V4, V6), (V4, V7)\}$$

а) Доповнення до G1 :

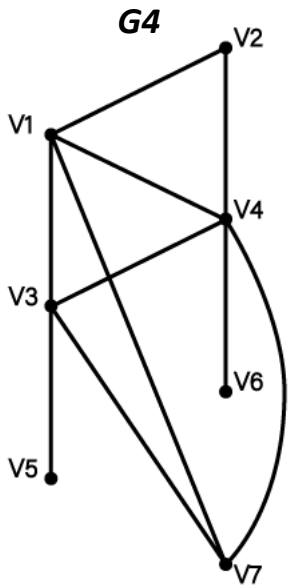


$$X_3 = \{V1, V2, V3, V4, V5, V7\}$$

$$W_3 = \{(V1, V7), (V2, V3), (V2, V4), (V2, V5),$$

$$(V2, V7), (V3, V7), (V4, V5)\}$$

б) Об’єднання G1 і G2 :



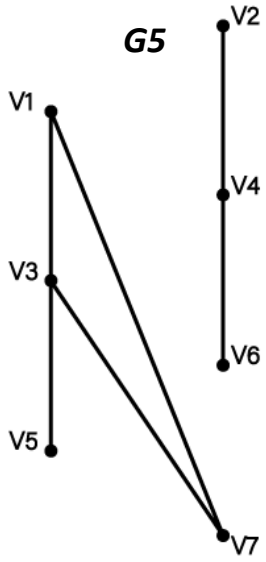
$$X_4 = \{V1, V2, V3, V4, V5, V6, V7\}$$

$$W_4 = \{(V1, V2), (V1, V3), (V1, V4), (V1, V7),$$

$$(V2, V4), (V3, V4), (V3, V5), (V3, V7),$$

$$(V4, V6), (V4, V7)\}$$

в) Кільцева сума G1 та G2 (G1⊕G2) :

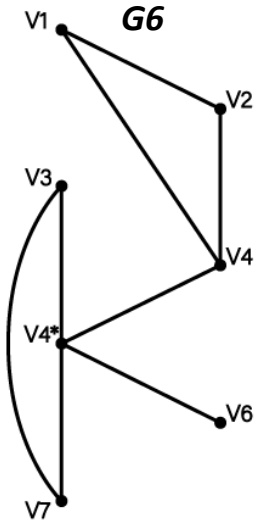


$$X_5 = \{V1, V2, V3, V4, V5, V6, V7\}$$

$$W_5 = \{(V1, V3), (V1, V7),$$

$$(V2, V4), (V3, V5), (V3, V7), (V4, V6),\}$$

г) Розчепимо вершину V4.



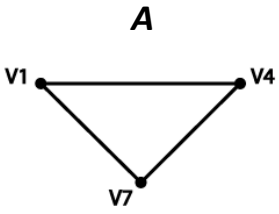
$$X_6 = \{V1, V2, V3, V4, V4*, V6, V7\}$$

$$W_6 = \{(V1, V2), (V1, V4), (V2, V4),$$

$$(V3, V4*), (V4, V4*), (V3, V7),$$

$$(V4*, V6), (V4*, V7)\}$$

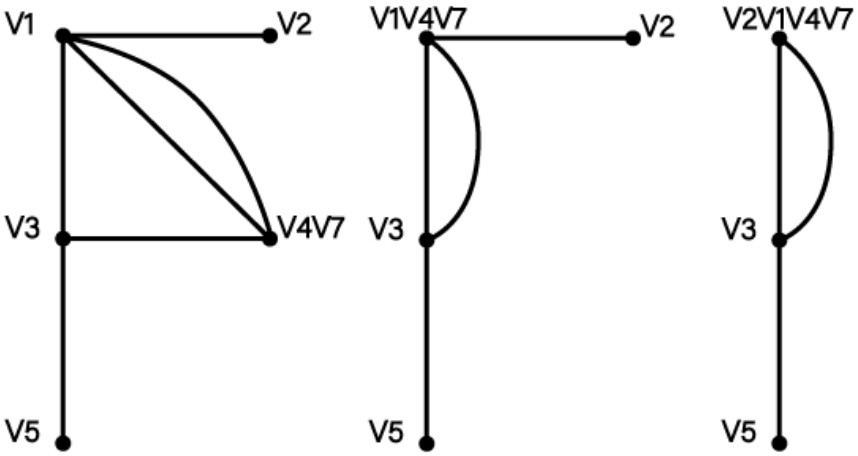
ґ) виділимо підграф A, що складається з 3-х вершин в G1 і виконаємо стягнення A в G1 (G1\ A) :



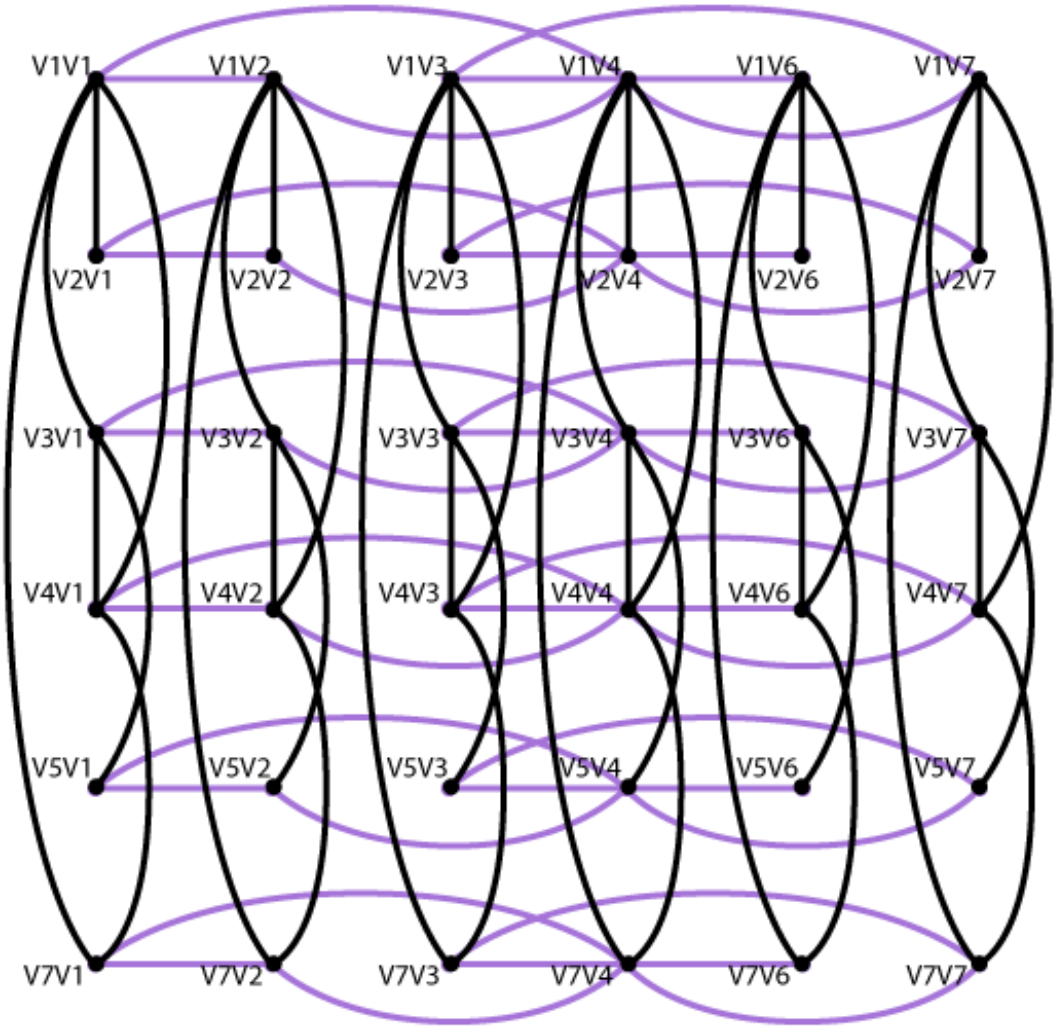
$$X_A = \{V1, V4, V7\}$$

$$W_A = \{(V1, V4), (V1, V7),$$

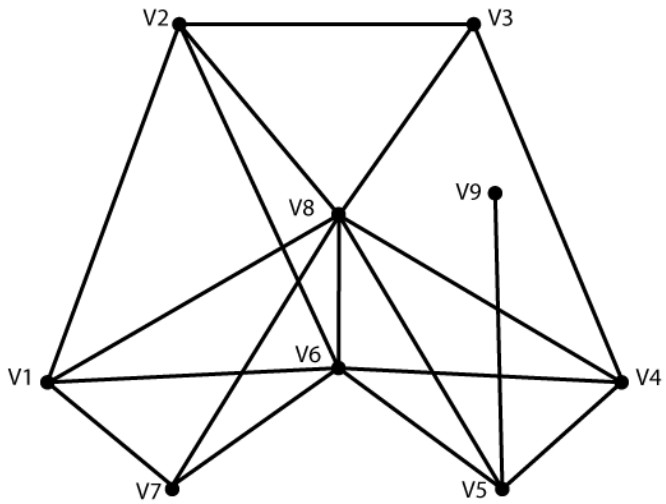
$$(V4, V7)\}$$



д) добуток графів $G1 \times G2$.



2. Скласти таблицю суміжності для орграфа.



Таблиця суміжності :

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
V_1	**0**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
V_2	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**0**
V_3	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**
V_4	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**
V_5	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**1**
V_6	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**
V_7	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**
V_8	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
V_9	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
V_1	**-**	**1**	**2**	**2**	**2**	**1**	**1**	**1**	**3**
V_2	**1**	**-**	**1**	**2**	**2**	**1**	**2**	**1**	**3**
V_3	**2**	**1**	**-**	**1**	**2**	**2**	**2**	**1**	**3**
V_4	**2**	**2**	**1**	**-**	**1**	**1**	**2**	**1**	**2**
V_5	**2**	**2**	**2**	**1**	**-**	**1**	**2**	**1**	**1**
V_6	**1**	**1**	**2**	**1**	**1**	**-**	**1**	**1**	**2**
V_7	**2**	**2**	**2**	**2**	**1**	**1**	**-**	**1**	**3**
V_8	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**-**	**2**
V_9	**3**	**3**	**3**	**2**	**1**	**2**	**3**	**2**	**-**

3. Для графа з другого завдання знайти діаметр.

Діаметр графа : 3.

4. Для графа з другого завдання виконати обхід дерева вшир (закінчується на парне число).

Вершина	**Номер**	**Черга**
V1	0	V1
V2	1	V1V2
V8	2	V1V2V8
V6	3	V1V2V8V6
V7	4	V1V2V8V6V7
-	-	V2V8V6V7
V3	5	V2V8V6V7V3
-	-	V8V6V7V3
V4	6	V8V6V7V3V6
V5	7	V8V6V7V3V6V5
-	-	V6V7V3V6V5
-	-	V7V3V6V5
-	-	V3V6V5
-	-	V6V5
-	-	V5
V9	8	V5V9
-	-	V9
-	-	Ø

Код програми для обходу графа вшир :

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

int main()
{
    int **arr, n, n0, *arr0;
  cout<<"Enter number of vertexes : ";
  cin>>n;
  cout<<"Enter number of edges : ";
  cin>>n0;
  arr = new int*[n];
  arr0 = new int[n];
  for(int i=0; i<n; i++){
    arr[i] = new int[n];
    for(int j=0;j<n;j++){
      arr[i][j]=0;
      arr0[j]=0;
    }
  }
  cout<<"Enter pair of vertexes that contains edge :\n";
  int a,b;
  for(int i=0;i<n0;i++){
    cin>>a>>b;
    arr[a-1][b-1]=1;
    arr[b-1][a-1]=1;
  }
  int num=0;
  arr0[num]=1;
```

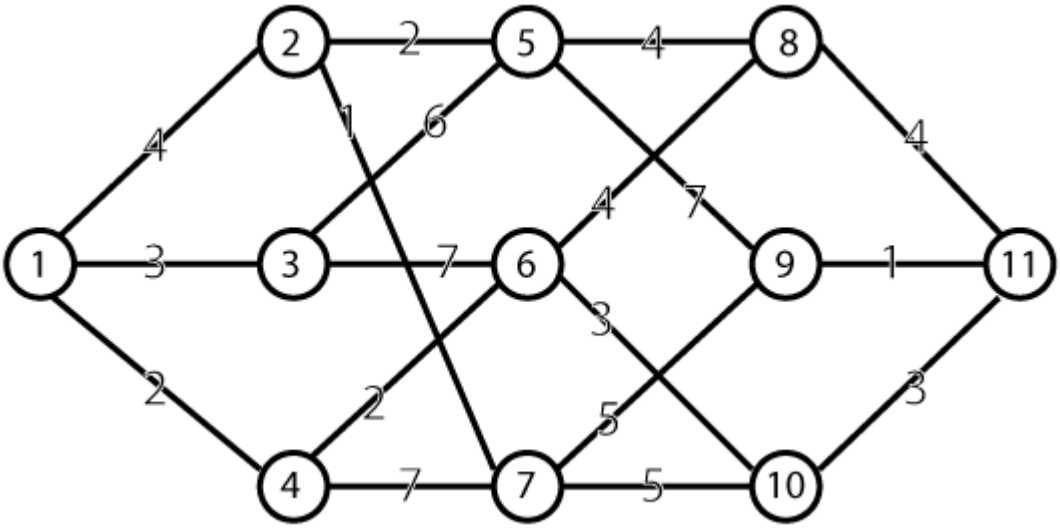
```
cout<<"V1\t\t"<<num<<"\t\t";
for(int i=0;i<n;i++){
   if(arr0[i]!=0){
      cout<<"V"<<arr0[i];
   }
}
cout<<endl;
int y=0, t=0, l=0;

for(;num<n;){
   y=0;
   for(int i=0;(i<n)&&(y==0);i++){
      if(arr0[i]!=0){y=arr0[i];}
   }
   t=0;
   for(int i=0;(i<n)&&(t==0);i++){
      if(arr[y-1][i]!=0){t=i;num++;arr0[num]=i+1;}
   }
   if(t!=0){
      cout<<"V"<<t+1<<"\t\t"<<num<<"\t\t";
      for(int i=0;i<n;i++){
   if(arr0[i]!=0){
      cout<<"V"<<arr0[i];
   }
   }
   cout<<endl;
   }
```

```
    else{
      for(int i=0, r=0;(i<n)&&(r==0);i++){
        if(arr0[i]!=0){arr0[i]=0;r++;}
      }
      cout<<"-\t\t-\t\t";
      l=0;
      for(int i=0;i<n;i++){
      if(arr0[i]!=0){
      cout<<"V"<<arr0[i];l++;
      }
      }
      if(l==0){cout<<"-";}
      cout<<endl;
    }
    for(int i=0;i<n;i++){
      if(arr0[i]!=0){
        for(int j=0;j<n;j++){
          if(arr0[j]!=0){
            arr[arr0[i]-1][arr0[j]-1]=0;
            arr[arr0[j]-1][arr0[i]-1]=0;
          }
        }
      }
    }
  }

  return 0;
}
```

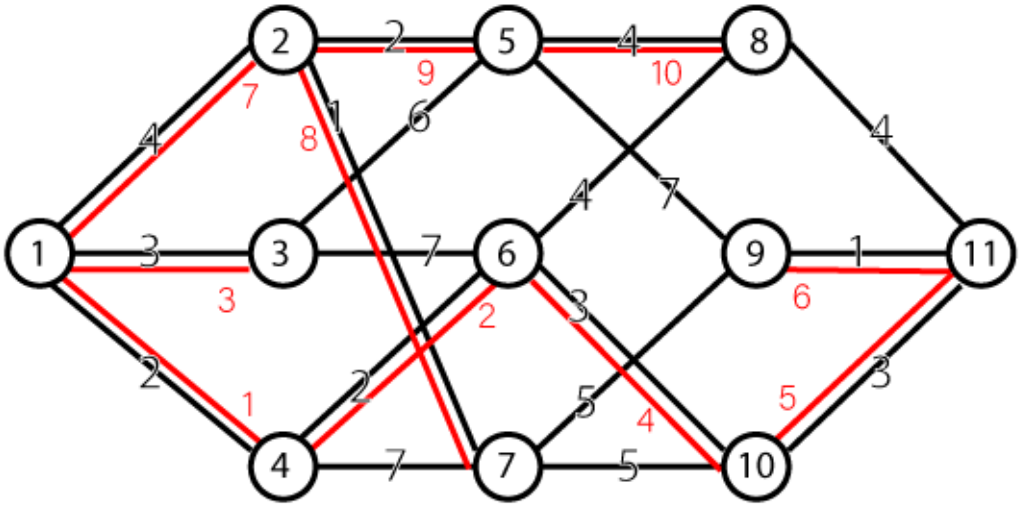
Вивід програми :

```
Enter number of vertexes : 9
Enter number of edges : 18
Enter pair of vertexes that contains edge :
1 2
1 8
1 6
1 7
2 6
2 3
2 8
3 8
3 4
4 5
4 6
4 8
5 9
5 8
5 6
6 8
7 6
7 8
V1              0               V1
V2              1               V1V2
V6              2               V1V2V6
V7              3               V1V2V6V7
V8              4               V1V2V6V7V8
-               -               V2V6V7V8
V3              5               V2V6V7V8V3
-               -               V6V7V8V3
V4              6               V6V7V8V3V4
V5              7               V6V7V8V3V4V5
-               -               V7V8V3V4V5
-               -               V8V3V4V5
-               -               V3V4V5
-               -               V4V5
-               -               V5
V9              8               V5V9
-               -               V9
-               -               -
```

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



Алгоритм Прима :



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

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

Вага : 2+2+3+3+3+1+4+1+2+4=25

Код програми для алгоритму Прима :

```
#include <iostream>

using namespace std;

int main()

{

        int n;

        cout << "\nenter number of vertices's  : ";

        cin >> n;

        int n0;

        cout << "\nenter number of edges  : ";

        cin >> n0;

        cout << "\nnow\nenter data in order :\nA(1)   A(2)   A(1)A(2)  (weight of the edge between them)\n";

        int** arr;

        arr = new int* [n];
```

```
for (int i = 0; i < n; i++) {
	arr[i] = new int[n];
	for (int j = 0; j < n; j++) {
		arr[i][j] = 0;
	}}
int a1, a2, w;
for (int i = 0; i < n0; i++) {
	cin >> a1 >> a2 >> w;
	arr[a1-1][a2-1] = w;
	arr[a2-1][a1-1] = w;
}
int* mas;
mas = new int[n];
for (int i = 0; i < n; i++) {
	mas[i] = 0;
}
mas[0] = 1;
int sum = 0;
cout << "\n\n";
for (int r=0, t = 0; r != (n - 1);) {
	t = 0;
	for (int i = 0; i < n; i++) {
		if (mas[i] != 0) {
			for (int j = 0; j < n; j++) {
				if (arr[i][j] != 0) {
					if (t == 0) {
						a1 = i;
						a2 = j;
						w = arr[i][j];
```

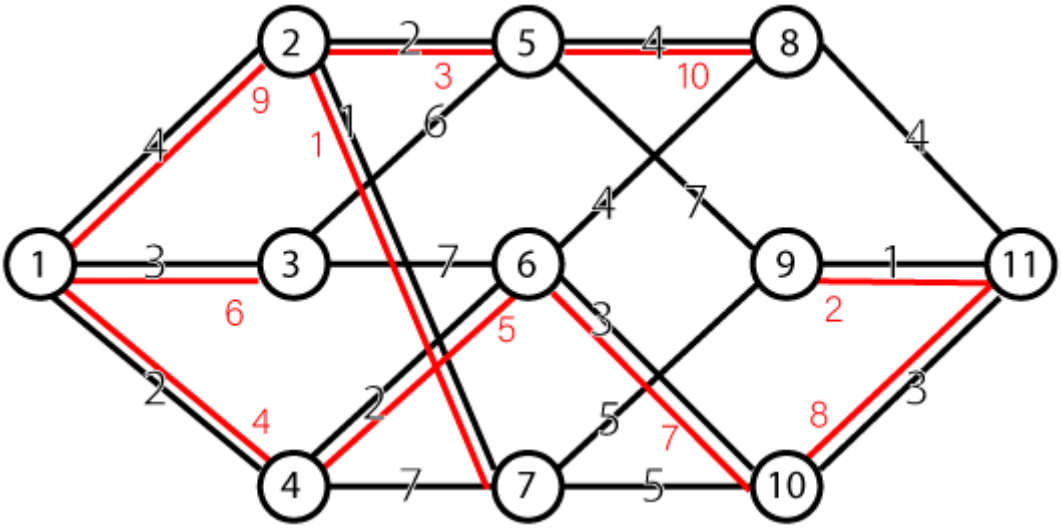
```
                                        t++;
                                }
                                else {
                                        if (arr[i][j] < w) {
                                                a1 = i;
                                                a2 = j;
                                                w = arr[i][j];
                                        }}}}}}
        sum += w;
        cout << "\nA(" << a1 + 1 << ") -> A(" << a2 + 1 << ") = " << w;
        mas[a2] = a2 + 1;
        arr[a1][a2] = 0;
        arr[a2][a1] = 0;
        r = 0;
        for (int q = 0; q < n; q++) {
                if (mas[q] != 0) { r++; }
        }}
for (int i = 0; i < n; i++) {
        if (mas[i] == 0) { a2 = i; }
}
for (int i = 0, t = 0; i < n; i++) {
        if (arr[i][a2] != 0) {
                if (t == 0) {
                        a1 = i;
                        w = arr[i][a2];
                        t++;
                }
                else {
                        if (arr[i][a2] < w) {
```

```
                                    a1 = i;

                                    w = arr[i][a2];

                                }}}}

        sum += w;

        cout << "\nA(" << a1 + 1 << ") -> A(" << a2 + 1 << ") = " << w;

        cout << "\n\nminimal weight : " << sum;

        cout << "\n\n\n";

        return 0;

}
```

Вивід програми :

```
enter number of vertices's  : 11

enter number of edges  : 18

now
enter data in order :
A(1)   A(2)   A(1)A(2)  (weight of the edge between them)
1 2 4
1 3 3
1 4 2
2 5 2
2 7 1
3 5 6
3 6 7
4 6 2
4 7 7
4 8 4
5 9 7
6 8 4
6 10 3
7 9 5
7 10 5
8 11 4
9 11 1
10 11 3



A(1) -> A(4) = 2
A(4) -> A(6) = 2
A(1) -> A(3) = 3
A(6) -> A(10) = 3
A(10) -> A(11) = 3
A(11) -> A(9) = 1
A(1) -> A(2) = 4
A(2) -> A(7) = 1
A(2) -> A(5) = 2
A(4) -> A(8) = 4

minimal weight : 25
```

Алгоритм Краскала :



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

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

Вага : 1+1+2+2+2+3+3+3+4+4=25

Код програми для алгоритму Краскала :

```
#include <iostream>
using namespace std;
struct edge
{
 int t1;
 int t2;
 int w;
};
struct eset
{
 edge data[100];
 int n;
};
eset set;
eset tree;
```

```
void sort ()
{
 edge a;
 for(int i=0;i<set.n;i++)
  {
     for(int j=0;j<set.n-1;j++)
     {
        if(set.data[j].w>set.data[j+1].w)
    {
     a = set.data[j];
     set.data[j] = set.data[j+1];
     set.data[j+1] = a;
    }}}}
int search (int is[],int d)
{
 return(is[d]); }
void aaa (int is[],int c1,int c2,int n)
{
 int i;
 for(int i=0;i<n;i++)
  {
     if(is[i]==c2)
     {
        is[i]=c1;
     }}}

void kraskal (int **arr,int n)
{
 int is[11],i,j,n1,n2;
 set.n=0;
 for(i=0;i<n;i++)
```

```
  for(j=0;j<i;j++)
  {
   if(arr[i][j]!=0)
   {
    set.data[set.n].t1=i;
    set.data[set.n].t2=j;
    set.data[set.n].w=arr[i][j];
    set.n++;}}
 sort ();
 for(i=0;i<n;i++)
  is[i]=i;
 tree.n=0;
 for(i=0;i<set.n;i++)
 {
  n1=search(is,set.data[i].t1);
  n2=search(is,set.data[i].t2);

  if(n1!=n2)
  {
   tree.data[tree.n]=set.data[i];
   tree.n=tree.n+1;
   aaa(is,n1,n2,n);
  }}}

void print ()
{
 int i,sum=0;
 for(i=0;i<tree.n;i++)
 {
  cout<<"V"<<tree.data[i].t2+1<<" -> V"<<tree.data[i].t1+1<<" = "<<tree.data[i].w<<endl;
  sum=sum+tree.data[i].w;  }
```

```
 cout<<"\nminimal weight : "<<sum<<endl;
 }


int main ()
{
 int n, m;
 cout<<"Enter the number of vertexes: ";
 cin>>n;
 cout<<"Enter the number of edges: ";
 cin>>m;
        cout << "\nnow\nenter data in order :\nA(1)   A(2)   A(1)A(2)  (weight of the edge between
them)\n";
        int** arr, **arr1;
        arr = new int* [n];
        arr1=new int*[n];
        for (int i = 0; i < n; i++) {
                arr[i] = new int[n];
                arr1[i]=new int[n];
                for (int j = 0; j < n; j++) {
                        arr[i][j] = 0;
                        arr1[i][j]=0;
                }}
        int a1, a2, w;
        for (int i = 0; i < m; i++) {
                cin >> a1 >> a2 >> w;
                arr[a1-1][a2-1] = w;
                arr[a2-1][a1-1] = w;
        }
  cout<<"\n\n";
 kraskal(arr, n);
 print();}
```

Вивід програми :

```
Enter the number of vertexes: 11
Enter the number of edges: 18

now
enter data in order :
A(1)   A(2)   A(1)A(2)  (weight of the edge between them)
1 2 4
1 3 3
1 4 2
2 5 2
2 7 1
3 5 6
3 6 7
4 6 2
4 7 7
4 8 4
5 9 7
6 8 4
6 10 3
7 9 5
7 10 5
8 11 4
9 11 1
10 11 3


V2 -> V7 = 1
V9 -> V11 = 1
V1 -> V4 = 2
V2 -> V5 = 2
V4 -> V6 = 2
V1 -> V3 = 3
V6 -> V10 = 3
V10 -> V11 = 3
V1 -> V2 = 4
V4 -> V8 = 4

minimal weight : 25
```

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

1)Почнемо з вершини V1 :

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

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

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

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

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

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

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

Вага шляху (1,5,7,6,2,4,3,8) : 1+2+2+1+1+5+7=19

2)Почнемо з вершини V2 :

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

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

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

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

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

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

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

Вага шляху (2,4,6,3,7,5,1,8) : 1+4+1+5+2+1+5=19

3)Почнемо з вершини V3 :

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

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

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

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

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

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

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

Вага шляху (3,6,2,4,1,5,7,8) : 1+1+1+5+1+2+2=13

4)Почнемо з вершини V4 :

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

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

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

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

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

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

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

Вага шляху (4,2,6,3,7,5,1,8) : 1+1+1+5+2+1+5=16

5)Почнемо з вершини V5 :

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

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

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

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

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

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

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

Вага шляху (5,1,6,2,4,3,7,8) : 1+2+1+1+5+5+2=17

6)Почнемо з вершини V6 :

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

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

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

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

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

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

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

Вага шляху (6,2,4,1,5,7,8,3) : 1+1+5+1+2+2+7=19

7)Почнемо з вершини V7 :

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

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

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

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

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

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

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

Вага шляху (7,5,1,6,2,4,3,8) : 2+1+2+1+1+5+7=19

8)Почнемо з вершини V8 :

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

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

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

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

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

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

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

Вага шляху (8,2,4,6,3,7,5,1) : 1+1+4++5+2+1=15

Код програми для алгоритму задачі Комівояжера :

```
#include <iostream>
#include <cstring>
using namespace std;
string hid(int a) {
	string one = "", two = "V(";
	int b;
	do {
		b = a % 10;
		switch (b) {
		case 0: one = "0" + one; break;
		case 1: one = "1" + one; break;
		case 2: one = "2" + one; break;
		case 3: one = "3" + one; break;
		case 4: one = "4" + one; break;
		case 5: one = "5" + one; break;
		case 6: one = "6" + one; break;
		case 7: one = "7" + one; break;
		case 8: one = "8" + one; break;
		case 9: one = "9" + one; break;
		default: cout << "error";
		}
		a /= 10;
	} while (a != 0);
	two += one + ")";
	return two;
}
```

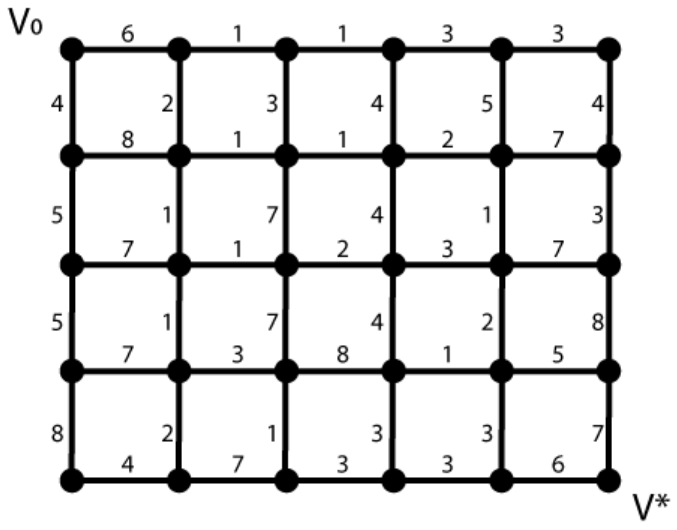
```
int main()
{
	int** arr, n, a, ** arr1, * arr2;
	string* vuvid;
	cout << "Enter number of vertexes : ";
	cin >> n;
	cout << "Enter weight of edges :\n";
	arr = new int* [n];
	arr1 = new int* [n];
	arr2 = new int[n];
	vuvid = new string[n];
	for (int i = 0; i < n; i++) {
		arr[i] = new int[n];
		arr1[i] = new int[n];
		vuvid[i] = "Starting from " + hid(i + 1) + " :\n\t" + hid(i + 1);
	}
	for (int i = 0; i < n; i++) {
		for (int j = 0; j < n; j++) {
			cin >> a;
			arr[i][j] = a;
		} }
	int t = 0, minw = 0, minn = 0, w = 0;
	for (int n0 = 0; n0 < n; n0++) {

		for (int i = 0; i < n; i++) {
			for (int j = 0; j < n; j++) {
				arr1[i][j] = arr[i][j];
			} }
		t = n0;
		w = 0;
		for (int h = 0; h < n-1; h++) {
```

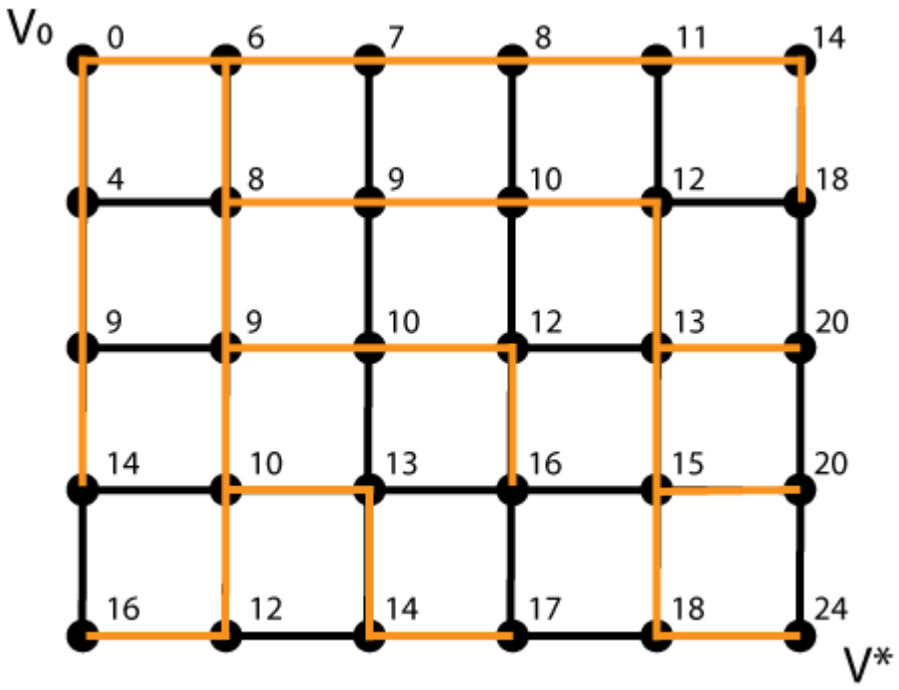
```
                minw = 0;
                minn = 0;
                for (int i = 0; i < n; i++) {
                        if (arr1[t][i] != 0) {
                                if (minw == 0) {
                                        minw = arr1[t][i];
                                        minn = i;
                                }
                                else if (arr1[t][i] < minw) {
                                        minn = i;
                                        minw = arr1[t][i];
                                }}
}
        w += minw;
                for (int i = 0; i < n; i++) {
                        arr1[t][i] = 0;
                        arr1[i][t] = 0;
                }
                t = minn;
                vuvid[n0] += hid(t + 1);
            }
            vuvid[n0] += "\n\tweight : ";
            arr2[n0] = w;
        }
        cout << "\n\n";
        for (int i = 0; i < n; i++) {
            cout << vuvid[i] << arr2[i] << endl<<endl;
        }
        return 0;
}
```

Вивід програми :

```
Enter number of vertexes : 8
Enter weight of edges :
0 4 6 5 1 2 3 5
4 0 5 1 5 1 5 1
6 5 0 5 6 1 5 7
5 1 5 0 6 4 5 5
1 5 6 6 0 3 2 2
2 1 1 4 3 0 2 2
3 5 5 5 2 2 0 2
5 1 7 5 2 2 2 0


Starting from V(1) :
        V(1)V(5)V(7)V(6)V(2)V(4)V(3)V(8)
        weight : 19

Starting from V(2) :
        V(2)V(4)V(6)V(3)V(7)V(5)V(1)V(8)
        weight : 19

Starting from V(3) :
        V(3)V(6)V(2)V(4)V(1)V(5)V(7)V(8)
        weight : 13

Starting from V(4) :
        V(4)V(2)V(6)V(3)V(7)V(5)V(1)V(8)
        weight : 16

Starting from V(5) :
        V(5)V(1)V(6)V(2)V(4)V(3)V(7)V(8)
        weight : 17

Starting from V(6) :
        V(6)V(2)V(4)V(1)V(5)V(7)V(8)V(3)
        weight : 19

Starting from V(7) :
        V(7)V(5)V(1)V(6)V(2)V(4)V(3)V(8)
        weight : 19

Starting from V(8) :
        V(8)V(2)V(4)V(6)V(3)V(7)V(5)V(1)
        weight : 15
```

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





Вага шляху : 24.

Код програми для знашодження шляху Дейкстри :

```
#include <iostream>
#include <cstring>
using namespace std;
string hid(int a) {
	string one = "", two = "V(";
	int b;
	do {
		b = a % 10;
		switch (b) {
		case 0: one = "0" + one; break;
		case 1: one = "1" + one; break;
		case 2: one = "2" + one; break;
		case 3: one = "3" + one; break;
		case 4: one = "4" + one; break;
		case 5: one = "5" + one; break;
		case 6: one = "6" + one; break;
```

```
                case 7: one = "7" + one; break;
                case 8: one = "8" + one; break;
                case 9: one = "9" + one; break;
                default: cout << "error"; }
                a /= 10;
        } while (a != 0);
        two += one + ")";
        return two;
}
int main()
{
        string* vuvid;
        int n, n0;
        cout << "\nenter number of all vertices  : ";
        cin >> n;
        cout << "\nenter number of vertices in the first line : ";
        cin >> n0;
        cout << "\nnow\nenter data in order :\nA(1)   A(2)   A(1)A(2)  (weight of the edge between
them)\n";
        long int** arr, * arr1, * arr2;
        arr = new long int* [n];
        arr1 = new long int[n];
        arr2 = new long int[n];
        vuvid = new string[n];
        for (int i = 0; i < n; i++) {
                arr[i] = new long int[n];
                arr2[i] = 0;
                arr1[i] = 0;
                for (int j = 0; j < n; j++) {
                        arr[i][j] = 0; }}
        int y=0, o=0;
```

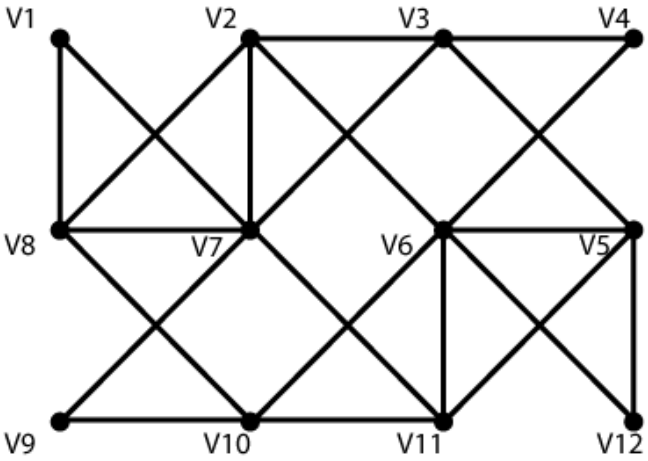
```
    int a1, a2, w;
    for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                    a1 = i + 1;
                    a2 = j + 1;
                    if (((a2 - a1 == 1) && (a1 % n0 != 0)) || (a2 - a1 == 6)) {
                            cout  << hid(a1) << "\t" << hid(a2) << "\t:\t";
                            cin >> w;
arr[a1 - 1][a2 - 1] = w;
            arr[a2 - 1][a1 - 1] = w; }}}
    arr1[0] = 1;
    vuvid[0] = hid(1);
    int k = 0, z=0,f=0;
    long int* ww;
    ww = new long int[n];
    ww[0] = 0;
    for (int i = 0; k != n-1; i++) {
            y = ww[z];
            z++;
            if(y!=n-1){
            for (int j = 0; j < n; j++) {
                            if (arr[j][y] != 0) {
                                    f++;
                                    ww[f] = j;
                                    if (arr2[j] == 0) {
                                            arr2[j] = (arr[y][j]+arr2[y]);
                                            vuvid[j] = vuvid[y] + " -> " + hid(j + 1); }
                                    else {
                                            if (arr2[j] > (arr[y][j] + arr2[y])) {
                                                    arr2[j] = (arr[y][j] + arr2[y]);
                                                    vuvid[j] = vuvid[y] + " -> " + hid(j + 1); }}
```

```
                    arr[y][j] = 0;
                    arr[j][y] = 0; }}
            arr1[y] = 2;
            k = 0;
            for (int j = 0; j < n; j++) {
                if (arr1[j] == 2) { k++; }}}}
    cout <<endl<< vuvid[n - 1] << " = " << arr2[n - 1]<<"\n\n\n\n";
    return 0; }
```

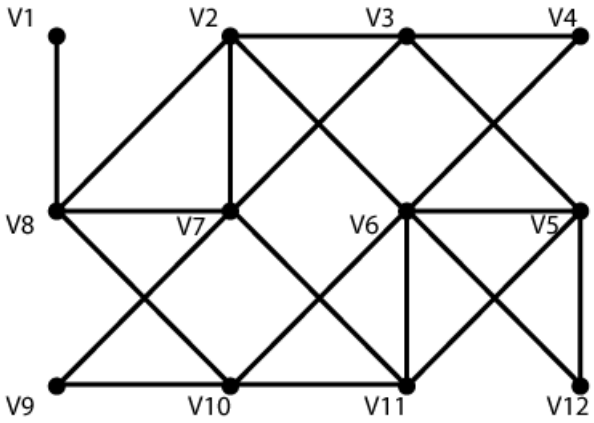
Вивід програми :

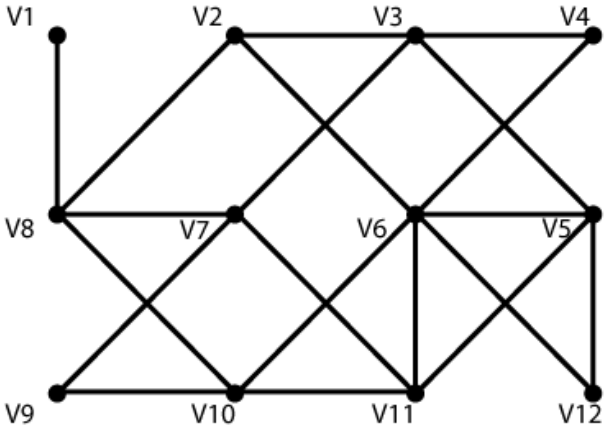
```
enter number of all vertices  : 30

enter number of vertices in the first line : 6

now
enter data in order :
A(1)   A(2)   A(1)A(2)  (weight of the edge between them)
V(1)    V(2)    :       6
V(1)    V(7)    :       4
V(2)    V(3)    :       1
V(2)    V(8)    :       2
V(3)    V(4)    :       1
V(3)    V(9)    :       3
V(4)    V(5)    :       3
V(4)    V(10)   :       4
V(5)    V(6)    :       3
V(5)    V(11)   :       5
V(6)    V(12)   :       4
V(7)    V(8)    :       8
V(7)    V(13)   :       5
V(8)    V(9)    :       1
V(8)    V(14)   :       1
V(9)    V(10)   :       1
V(9)    V(15)   :       7
V(10)   V(11)   :       2
V(10)   V(16)   :       4
V(11)   V(12)   :       7
V(11)   V(17)   :       1
V(12)   V(18)   :       3
V(13)   V(14)   :       7
V(13)   V(19)   :       5
V(14)   V(15)   :       1
V(14)   V(20)   :       1
V(15)   V(16)   :       2
V(15)   V(21)   :       7
V(16)   V(17)   :       3
V(16)   V(22)   :       4
V(17)   V(18)   :       7
V(17)   V(23)   :       2
V(18)   V(24)   :       8
V(19)   V(20)   :       7
V(19)   V(25)   :       8
V(20)   V(21)   :       3
V(20)   V(26)   :       2
V(21)   V(22)   :       8
V(21)   V(27)   :       1
V(22)   V(23)   :       1
V(22)   V(28)   :       3
V(23)   V(24)   :       5
V(23)   V(29)   :       3
V(24)   V(30)   :       7
V(25)   V(26)   :       4
V(26)   V(27)   :       7
V(27)   V(28)   :       3
V(28)   V(29)   :       3
V(29)   V(30)   :       6

V(1) -> V(2) -> V(8) -> V(9) -> V(10) -> V(11) -> V(17) -> V(23) -> V(29) -> V(30) = 24
```

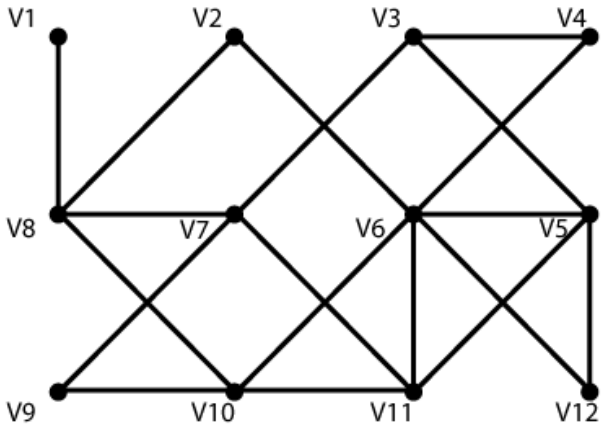
8. Знайти ейлеровий цикл в ейлеровому графі двома методами: а) Флері; б) елементарних циклів.

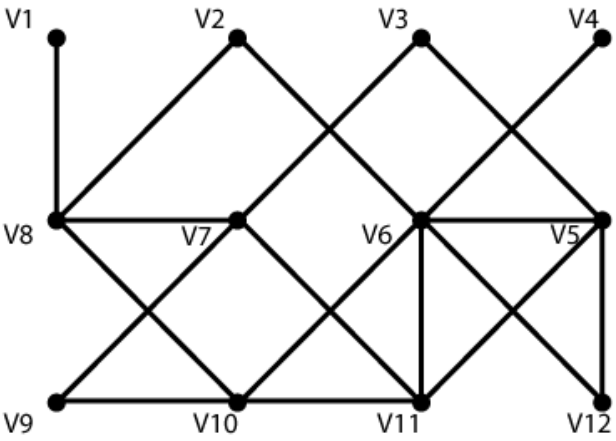


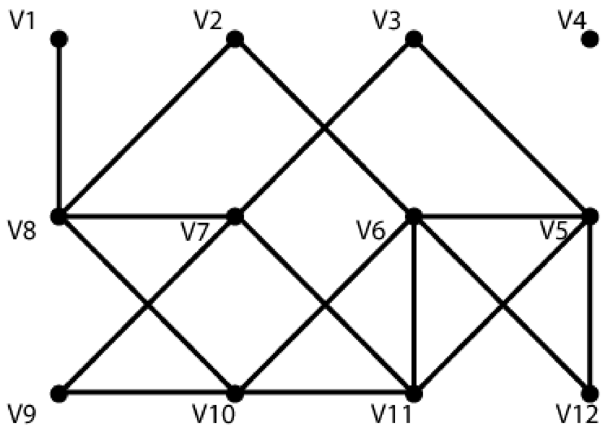
1)Флері:

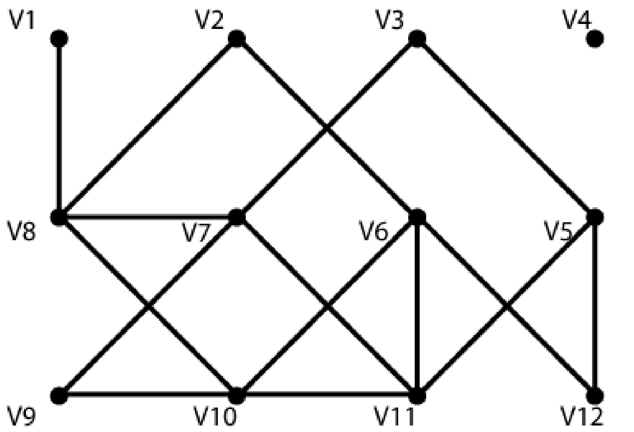


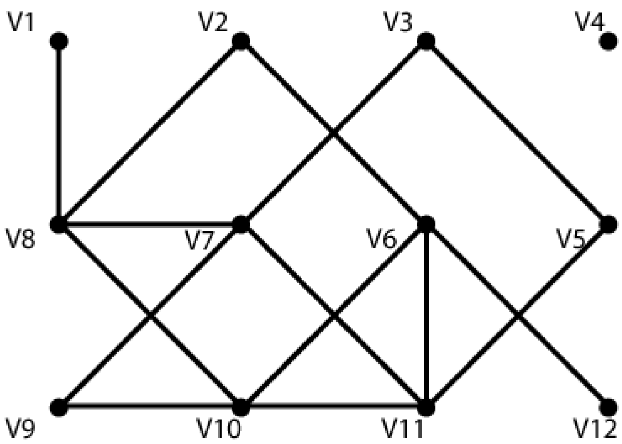


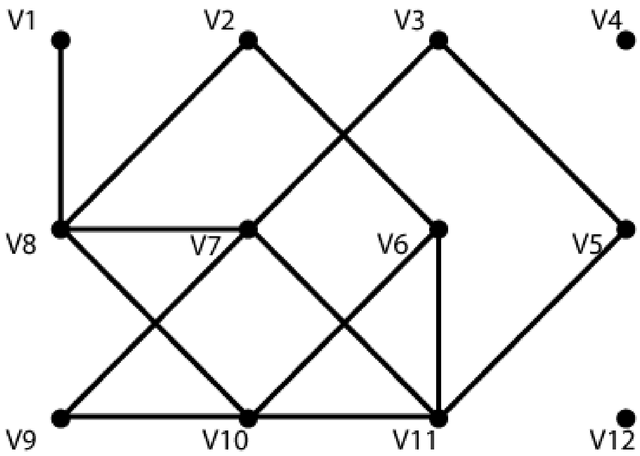


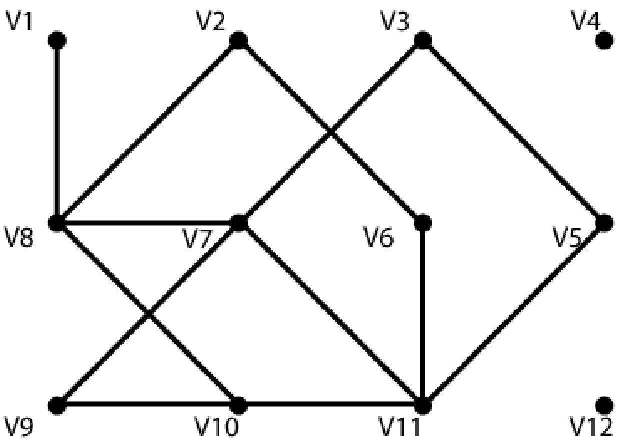


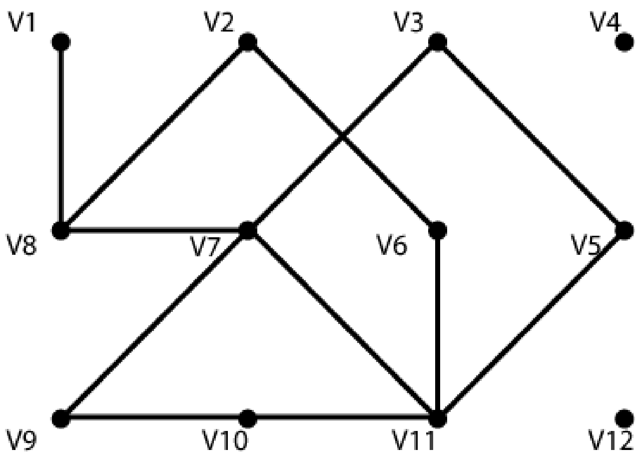

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

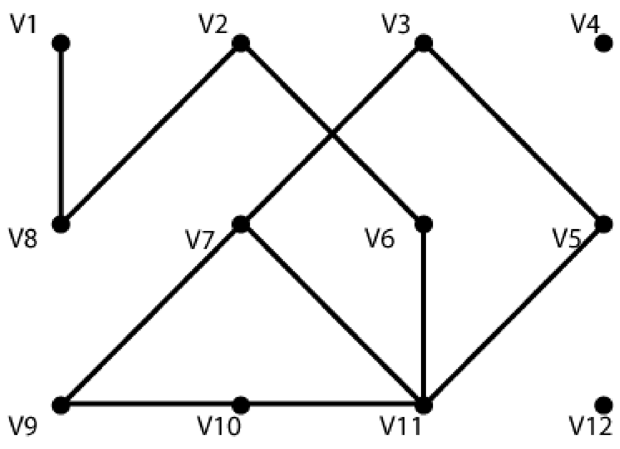

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

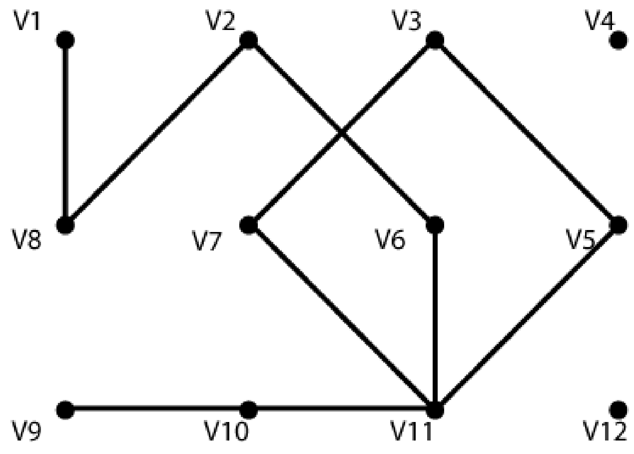

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

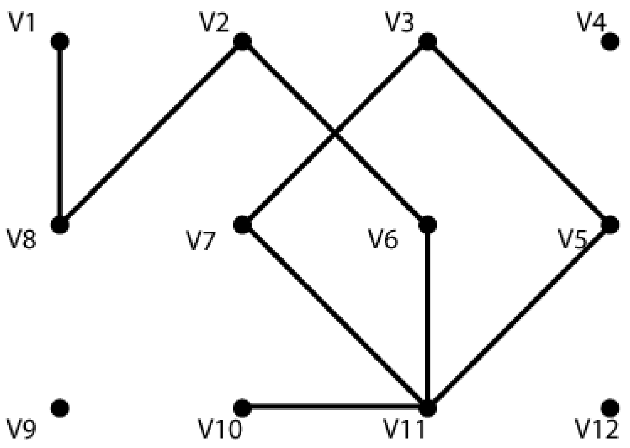

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

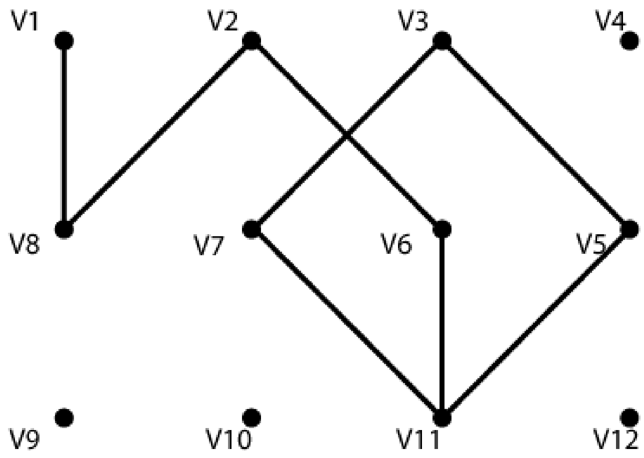

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

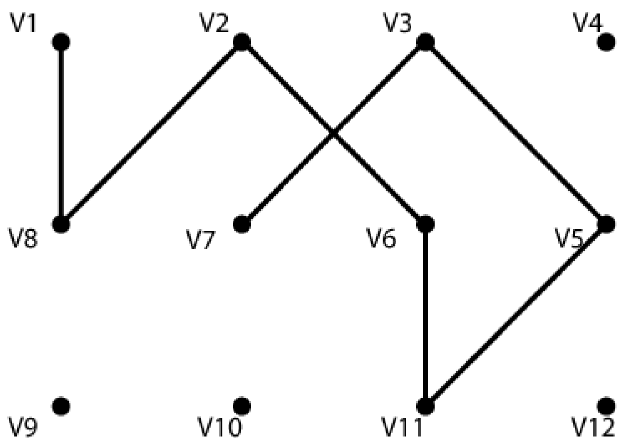

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

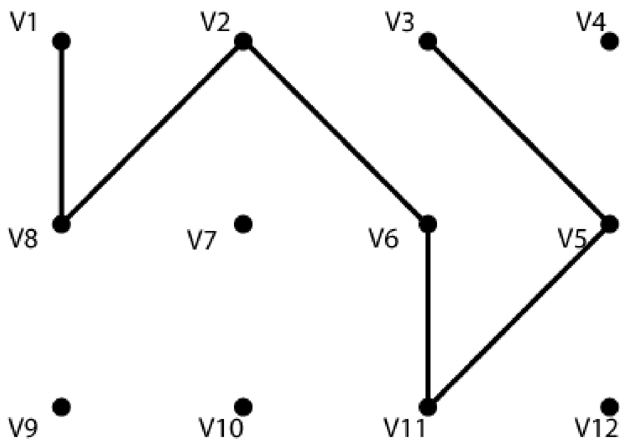

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

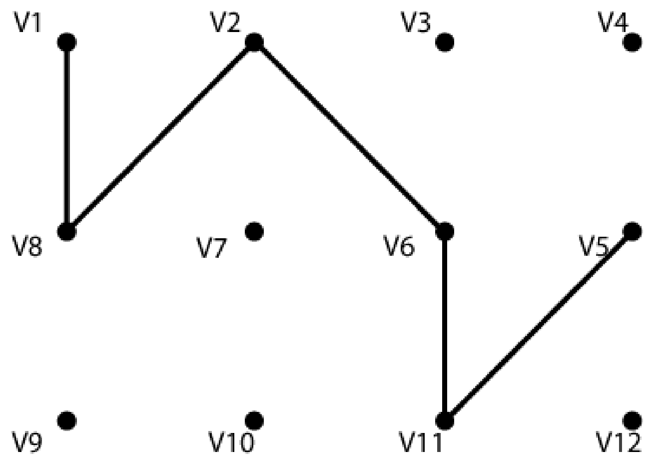

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

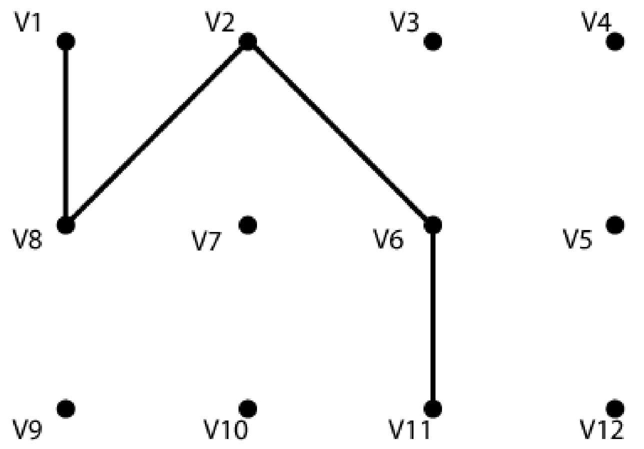

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

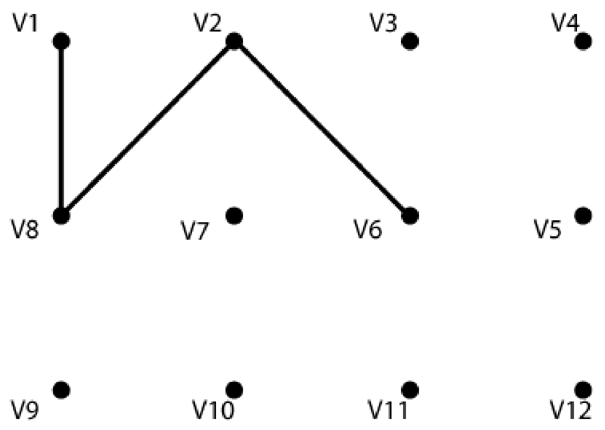

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

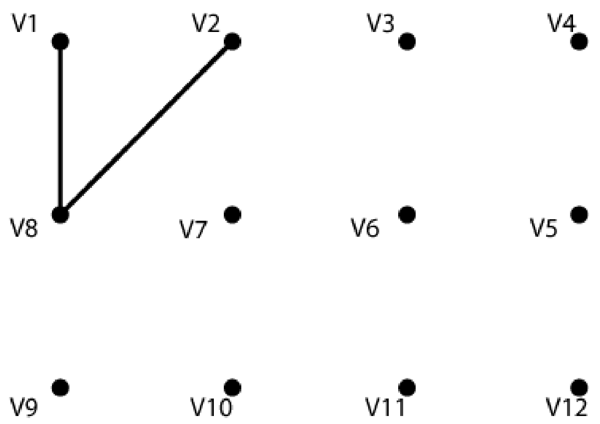

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

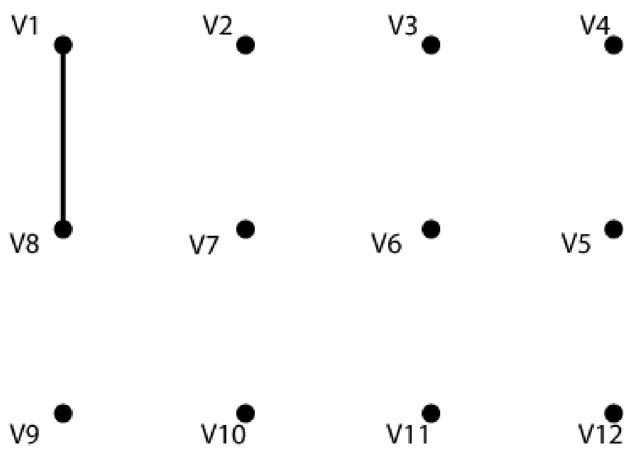

V1
V2
V3
V4
V8
V7
V6
V5
V9
V10
V11
V12

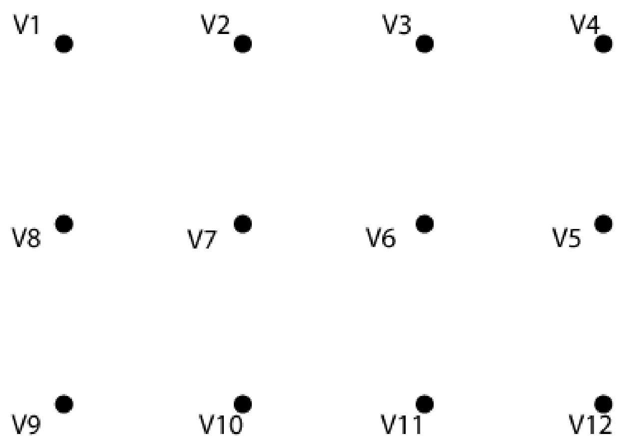

V1 V2 V3 V4
V8 V7 V6 V5
V9 V10 V11 V12

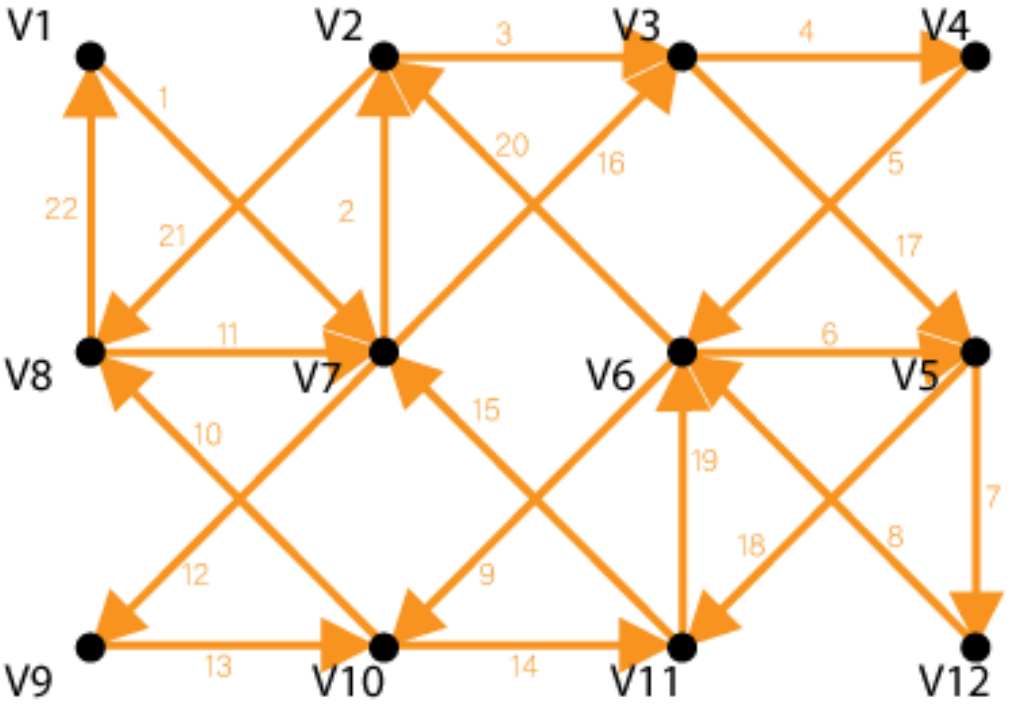

V1 V2 V3 V4
V8 V7 V6 V5
V9 V10 V11 V12


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


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


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


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



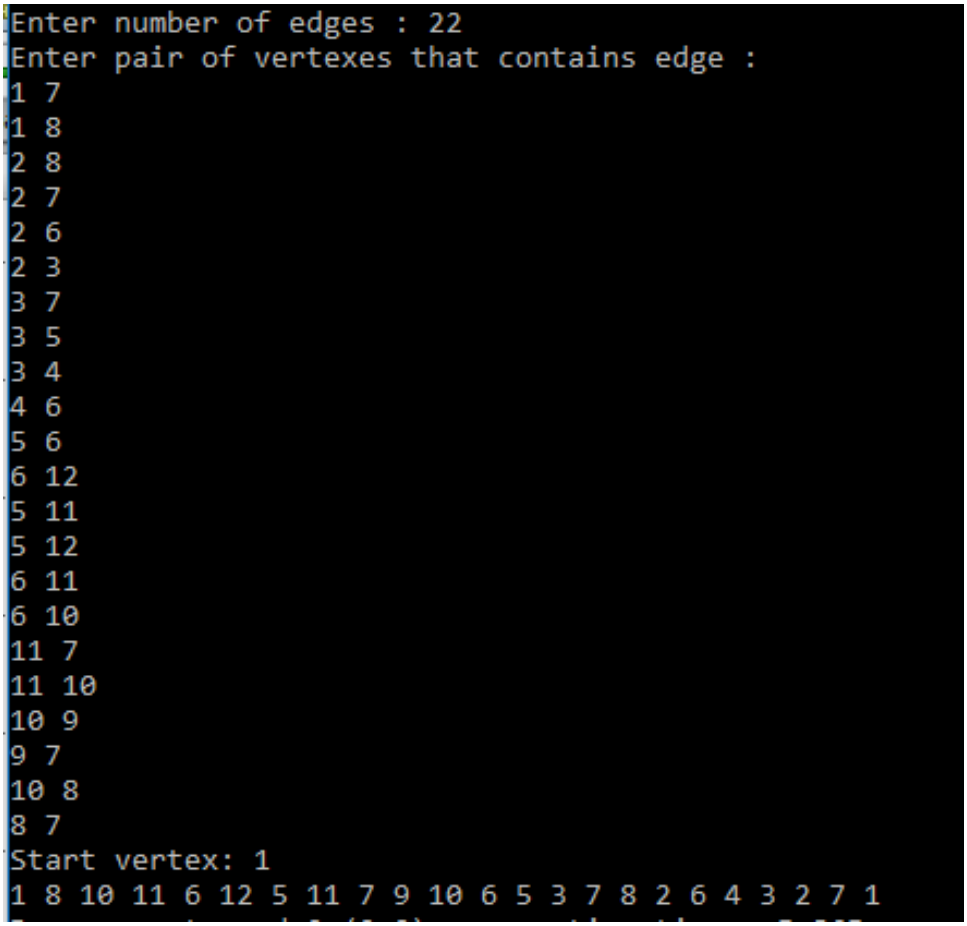
Код програми для знаходження ейлерового циклу :

```
#include <iostream>
#include <cstdio>
#define N 12
#define STACK_SIZE 100

using namespace std;

int G[N][N]={0};
int k;
int Stack[STACK_SIZE];
void Search(int v)
{
   int i;
   for(i = 0; i < N; i++)
      if(G[v][i])
      {
         G[v][i] = G[i][v] = 0;
```

```
            Search(i);
        }
    Stack[++k] = v+1;
}
int main()
{
    int n;
    cout<<"Enter number of edges : ";
    cin>>n;
    int a,b;
    cout<<"Enter pair of vertexes that contains edge :\n";
    for(int i=0;i<n;i++){
        cin>>a>>b;
        G[a-1][b-1]=1;
        G[b-1][a-1]=1;
    }

    int T, p, q, s;
    int j, vv;
    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 vertex: ";
```

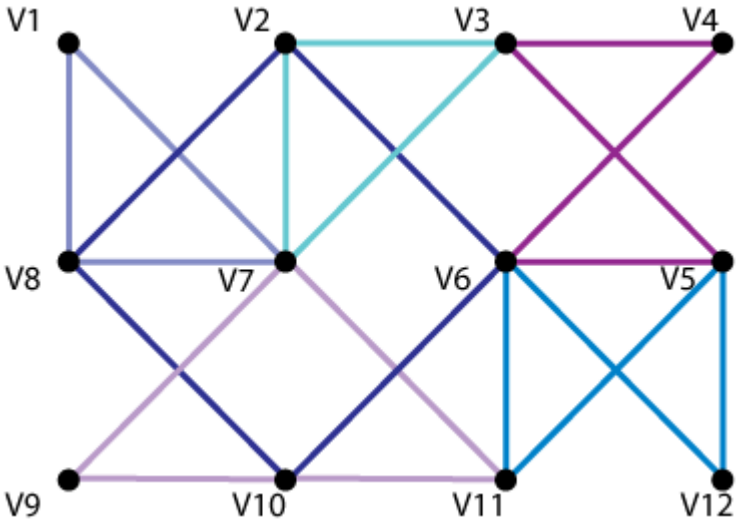
```
    cin >> vv;
    vv-=1;
    if(T)
    {
       Search (vv);
       for(j = 0; j <= k; j++)
          cout << Stack[j] << " ";
    }
    else
       cout << "it is not Eulerian graph\n";
    return 0;
}
```

Вивід програми :

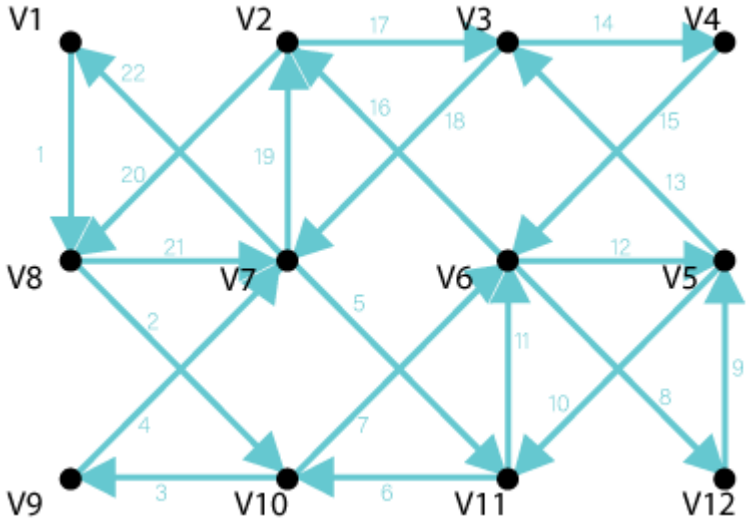


2)Елементарні цикли :

Виділимо прості цикли:



Отримаємо такий ейлеровий цикл :



Код програми для Елементарних циклів :

```
#include <iostream>

#include <iostream>

#include <vector>

using namespace std;

vector <int> graph;
```

```
int inf = 99;


bool check(vector<int> V, int vertix) {
	for (auto i = V.begin(); i != V.end(); i++) {
		if (*i == vertix) return false;
	}
	return true;
}
void Find(vector<int>* V, int** B, int n, int position, int f_vertix) {
	for (int i = position, k = 0; k < 1; i++, k++) {
		for (int j = 0; j < n; j++)
			if (B[i][j] == 1 && check((*V), j)) {
				if (j == f_vertix && (*V).size() > 2) {
					if (inf > V->size()) {
						graph.clear();
						graph.push_back(f_vertix + 1);
						for (auto it = (*V).begin(); it != (*V).end();
							it++)
							graph.push_back(*it + 1);
						graph.push_back(f_vertix + 1);
						inf = V->size();
						break;
					}
				}
				else {
					(*V).push_back(j);
					Find(V, B, n, j, f_vertix);
				}
			}
	}
	if (V->size() != 0)
```

```
                V->pop_back();
}
int main() {
        int n, n0;
        cout << "Enter number of vertices: ";
        cin >> n;
        int** A = new int* [n];
        for (int i = 0; i < n; i++) {
                A[i] = new int[n];
                for (int j = 0; j < n; j++) {
                        A[i][j] = 0;
                }
        }
        cout << "Enter number of edges : ";
        cin >> n0;
        int a, b;
        cout << "Enter pair of vertexes that contains edge :\n";
        for (int i = 0; i < n0; i++) {
                cin >> a >> b;
                A[a - 1][b - 1] = 1;
                A[b - 1][a - 1] = 1;
        }
        /*
        for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                        cin >> A[i][j];
                }
        }*/
        vector<int> C;
        vector<int> B;
        cout << endl;
```

```
	int counted, p, j, sum;
	counted = 1;
	for (p = 0; p < n; p++)
	{
		sum = 0;
		for (j = 0; j < n; j++)
		{
			sum += A[p][j];
		}
		if (sum % 2) counted = 0;
	}
	cout << endl;
	cout << "So, the cycles are: " << endl;
	if (counted) {
		for (int j = 0; j < n; j++) {
			inf = 99;
			Find(&C, A, n, j, j);
			for (int i = 1; i <= graph.size(); i++) {
				cout <<"V"<< graph[i - 1] << " ";
			}
			cout << endl;
			graph.clear();
		}
	}
	else
		cout << "It is not correct \n";
	cout << endl;
	return 0;
}
```

Вивід програми :

```
Enter number of vertices: 12
Enter number of edges : 22
Enter pair of vertexes that contains edge :
1 7
1 8
2 8
2 7
2 6
2 3
3 7
3 5
3 4
4 6
5 6
6 12
5 11
5 12
6 11
6 10
11 7
11 10
10 9
9 7
10 8
8 7


So, the cycles are:
V1 V7 V2 V8 V1
V2 V3 V4 V6 V2
V3 V2 V6 V4 V3
V4 V3 V2 V6 V4
V5 V3 V2 V6 V5
V6 V2 V3 V4 V6
V7 V1 V8 V2 V7
V8 V1 V7 V2 V8
V9 V7 V8 V10 V9
V10 V6 V2 V8 V10
V11 V5 V3 V7 V11
V12 V5 V11 V6 V12
```

9. Спростити формулу (привести її до скороченої ДНФ). $(\boldsymbol{x} \vee \bar{\boldsymbol{z}})(\bar{\boldsymbol{y}} \vee \boldsymbol{z})$

Розв'язування:

$$(\boldsymbol{x} \vee \bar{\boldsymbol{z}})(\bar{\boldsymbol{y}} \vee \boldsymbol{z}) = \boldsymbol{x}\bar{\boldsymbol{y}} \vee \boldsymbol{xz} \vee \bar{\boldsymbol{z}}\bar{\boldsymbol{y}} \vee \bar{\boldsymbol{z}}\boldsymbol{z} = \boldsymbol{x}\bar{\boldsymbol{y}} \vee \boldsymbol{xz} \vee \bar{\boldsymbol{z}}\bar{\boldsymbol{y}}$$