

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

**Кафедра систем штучного інтелекту**



**Лабораторна робота №5  
на тему:  
«Знаходження найкоротшого маршруту за  
алгоритмом Дейкстра. Плоскі планарні графи»  
з дисципліни  
«Дискретна математика»**

**Виконав:**  
студент групи КН-110  
Максим Романьчук  
**Викладач:**  
Мельникова Н. І.

Львів – 2018 р.

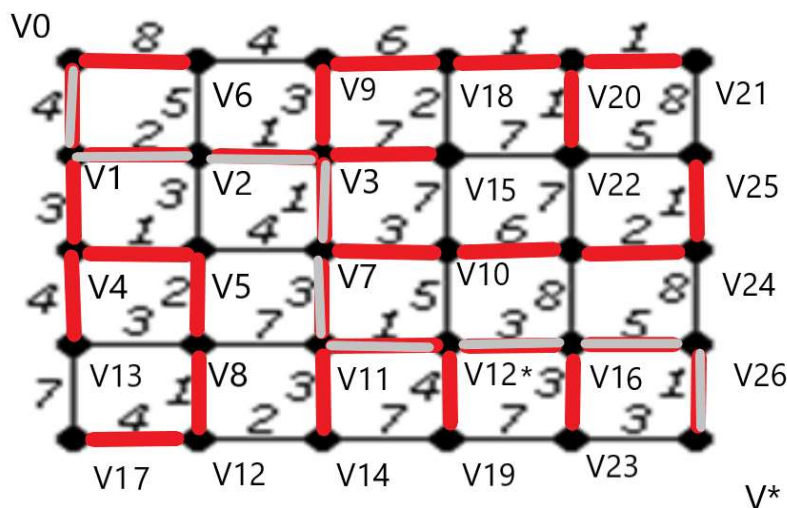
**Мета роботи:** набуття практичних вмінь та навичок з використання алгоритму Дейкстра.

### Варіант 8. Завдання 1.

**Розв'язати на графах наступні 2 задачі:**

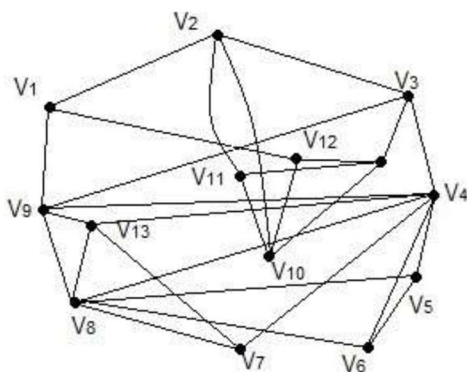
1. За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин  $V_0$  і  $V^*$ .
2. За допомогою  $\gamma$ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.

1.Відповідь:

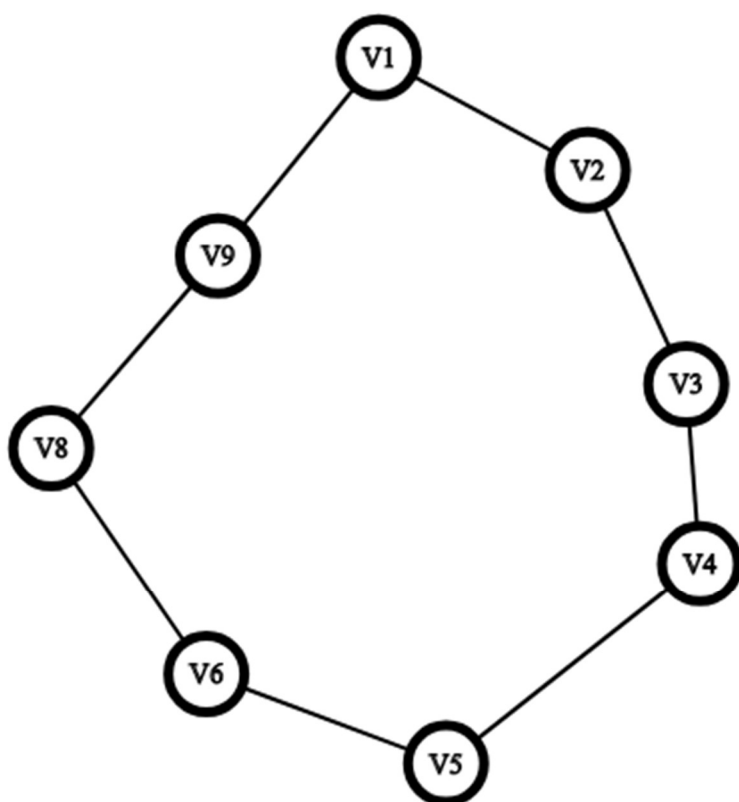


$V_1 = 4, V_2 = 6, V_3 = 7, V_4 = 7, V_5 = 8, V_6 = 8, V_7 = 8, V_8 = 10, V_9 = 10, V_{10} = 11, V_{11} = 11, V_{12} = 12, V_{12}^* = 12, V_{13} = 12, V_{14} = 14, V_{15} = 14, V_{16} = 15, V_{17} = 16, V_{18} = 16, V_{19} = 16, V_{20} = 17, V_{21} = 18, V_{22} = 18, V_{23} = 18, V_{24} = 19, V_{25} = 20, V_{26} = 20, V^* = 21.$

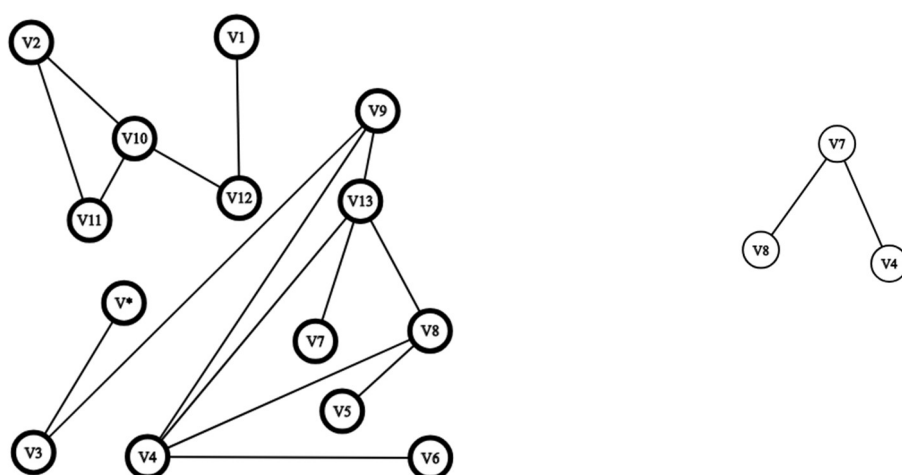
2. Відповідь:



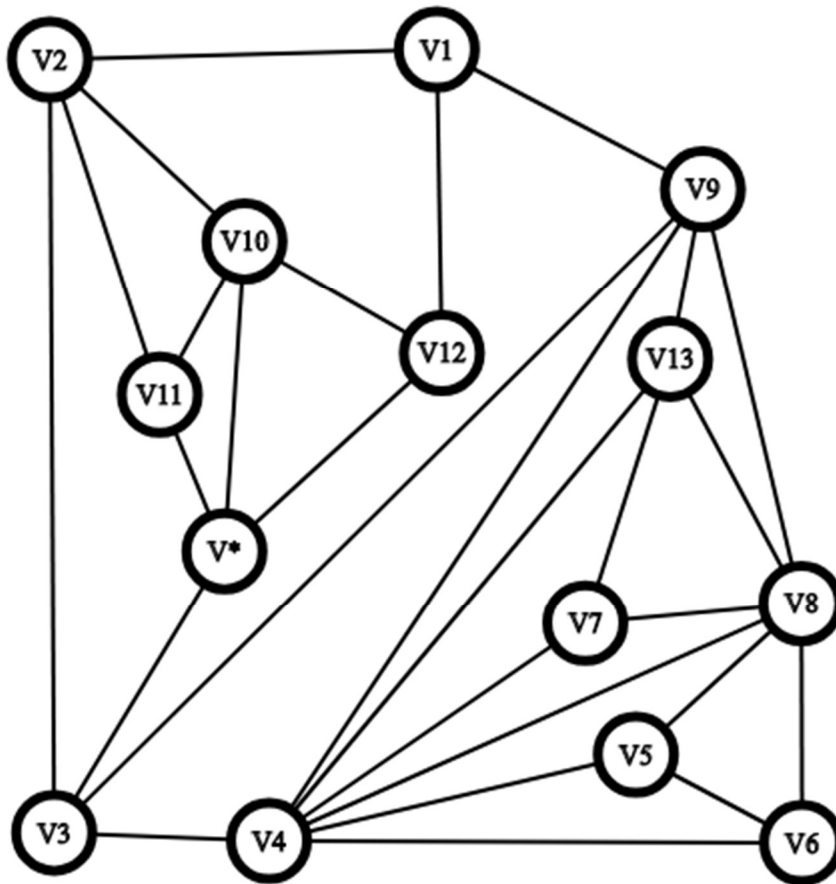
Обираю найбільший цикл графа.



І залишки

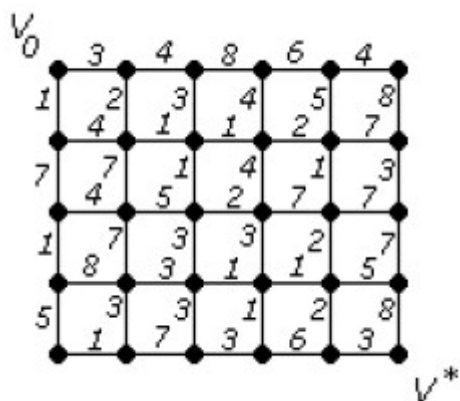


Ось результат:



### Варіант 8. Завдання №2.

Написати програму, яка реалізує алгоритм Дейкстри знаходження найкоротшого шляху між парою вершин у графі. Протестувати розроблену програму на графі згідно свого варіанту.



## Код програми:

```
1 #include <stdio.h>
2
3 void print_the_shortest_way(int array [30][4],int node) {
4     if(array[node][3]==-1) {
5         printf("V%d=%d ",array[node][0],array[node][2]);
6     }
7     else {
8         print_the_shortest_way(array,array[node][3]);
9         printf("V%d=%d ",array[node][0],array[node][2]);
10    }
11 }
12
13 int main(void) {
14     int i,k,lenth=0;
15     int result [30] [2];
16     int nodes [30] [4] = { //{number,times_used,path_value,last_node}
17         {1,1,0,-1},{2,0,0,0},{3,0,0,0},{4,0,0,0},{5,0,0,0},{6,0,0,0},
18         {7,0,0,0},{8,0,0,0},{9,0,0,0},{10,0,0,0},{11,0,0,0},{12,0,0,0},
19         {13,0,0,0},{14,0,0,0},{15,0,0,0},{16,0,0,0},{17,0,0,0},{18,0,0,0},
20         {19,0,0,0},{20,0,0,0},{21,0,0,0},{22,0,0,0},{23,0,0,0},{24,0,0,0},
21         {25,0,0,0},{26,0,0,0},{27,0,0,0},{28,0,0,0},{29,0,0,0},{30,0,0,0}
22     };
23     int edges [49] [3] = { //{from,to,weight}
24         {1,2,3},{2,3,4},{3,4,8},{4,5,6},{5,6,4},
25         {1,7,1},{2,8,2},{3,9,3},{4,10,4},{5,11,5},{6,12,8},
26         {7,8,4},{8,9,1},{9,10,1},{10,11,2},{11,12,7},
27         {7,13,7},{8,14,7},{9,15,1},{10,16,4},{11,17,1},{12,18,3},
28         {13,14,4},{14,15,5},{15,16,2},{16,17,7},{17,18,7},
29         {13,19,1},{14,20,7},{15,21,3},{16,22,3},{17,23,2},{18,24,7},
30         {19,20,8},{20,21,3},{21,22,1},{22,23,1},{23,24,5},
31         {19,25,5},{20,26,3},{21,27,3},{22,28,1},{23,29,2},{24,30,8},
32         {25,26,1},{26,27,7},{27,28,3},{28,29,6},{29,30,3}
33     };
34     //sorting edges
35     int temp,pointer;
36     do{
37         pointer=0;
38         for (i = 0; i < 48; i++){
39             if (edges[i][2] > edges[i + 1][2]) {
40                 for (k = 0, temp = 0; k < 3; k++) {
41                     temp = edges [i + 1][k];
42                     edges [i + 1][k] = edges [i][k];
43                     edges [i][k] = temp;
44                 }
45                 pointer++;
46             }
47         }
48     } while(pointer > 0);
49 }
```

```

41     temp = edges [i + 1][k];
42     edges [i + 1][k] = edges [i][k];
43     edges [i][k] = temp;
44 }
45     pointer++;
46 }
47 }
48 }while(pointer!=0);
49
50 while(nodes[29][1]==0) {
51     for(k=1;k<9;k++) {
52         for (i = 0; i < 49; i++) {
53             if (nodes [edges[i][0] - 1][1] != nodes [edges[i][1] - 1][1] && edges[i][2]==k) {
54                 if (nodes [edges[i][0] - 1][1] == 0 && nodes [edges[i][1] - 1][1] != 0) {
55                     nodes [edges[i][0] - 1][1] = nodes [edges[i][1] - 1][1];
56                     nodes [edges[i][1] - 1][1] = nodes [edges[i][1] - 1][1] + 1;
57                     nodes [edges[i][0] - 1][2] = nodes [edges[i][1] - 1][2] + edges[i][2];
58                     result[lenth][0] = nodes[edges[i][0] - 1][0] - 1;
59                     result[lenth][1] = nodes[edges[i][0] - 1][2];
60                     nodes [edges[i][0] - 1][3] = nodes [edges[i][1] - 1][0]-1;
61                     lenth++;
62                 }
63                 else if (nodes [edges[i][0] - 1][1] != 0 && nodes [edges[i][1] - 1][1] == 0) {
64                     nodes [edges[i][1] - 1][1] = nodes [edges[i][0] - 1][1];
65                     nodes [edges[i][0] - 1][1] = nodes [edges[i][0] - 1][1] + 1;
66                     nodes [edges[i][1] - 1][2] = nodes [edges[i][0] - 1][2] + edges[i][2];
67                     result[lenth][0] = nodes[edges[i][1] - 1][0] - 1;
68                     result[lenth][1] = nodes[edges[i][1] - 1][2];
69                     nodes [edges[i][1] - 1][3] = nodes [edges[i][0] - 1][0]-1;
70                     lenth++;
71                 }
72             }
73         }
74     }
75 }
76 //sorting valuable nodes
77 do{
78     pointer=0;
79     for (i = 0; i < lenth; i++){
80         if (result[i][1] > result[i + 1][1]) {
81             for (k = 0, temp = 0; k < 2; k++) {
82                 temp = result [i + 1][k];
83                 result[i + 1][k] = result[i][k];
84                 result[i][k] = temp;
85             }
86             pointer++;
87         }
88     }
89 }while(pointer!=0);
90 printf("Used edges: \n");
91 for(i=1;i<=lenth;i++) {
92     printf("V%d = %d\n",result[i][0],result[i][1]);
93 }
94 printf("the shortest way is: \n");
95 print_the_shortest_way(nodes,29);
96
97 return 0;

```

Результат роботи:

Used edges:

V6 = 1

V1 = 3

V7 = 5

V8 = 6

V2 = 7

V9 = 7

V14 = 7

V12 = 8

V10 = 9

V15 = 9

V20 = 10

V13 = 12

V21 = 12

V26 = 13

V4 = 14

V3 = 15

V27 = 16

V11 = 16

V16 = 16

V19 = 19

V25 = 22

V28 = 22

V17 = 23

V5 = 24

V18 = 27

V23 = 30

V24 = 32

V29 = 38

the shortest way is:

V1=0 V7=1 V8=5 V9=6 V15=7 V16=9 V17=16 V18=23 V24=30 V30=38

**Висновок:** Виконуючи цю роботу, я набув практичних вмінь та навичок з використання алгоритму Дейкстра.