

МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА
ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту



Лабораторна робота №5
З курсу “Дискретна математика ”

Виконав:
ст.гр. КН-110
Крушельницький Юрій
Викладач:
Мельникова Н.І.

Лабораторна робота № 5.

Тема: Знаходження найкоротшого маршруту за алгоритмом Дейкстри. Плоскі планарні графи.

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

ТЕОРЕТИЧНІ ВІДОМОСТІ ТА ПРИКЛАДИ РОЗВ'ЯЗАННЯ ЗАДАЧ:

Задача знаходження найкоротшого шляху з одним джерелом полягає у знаходженні найкоротших (мається на увазі найоптимальніших за вагою) шляхів від деякої вершини (джерела) до всіх вершин графа G .

Для розв'язку цієї задачі використовується «жадібний» алгоритм, який називається алгоритмом Дейкстри.

«Жадібними» називаються алгоритми, які на кожному кроці вибирають оптимальний із можливих варіантів.

Задача про найкоротший ланцюг. Алгоритм Дейкстри.

Плоскі і планарні графи

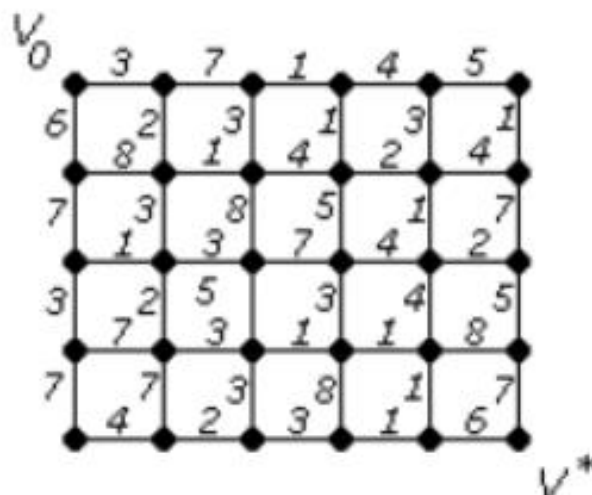
Плоским графом називається граф, вершини якого є точками площини, а ребра – безперервними лініями без самоперетинань, що з'єднують відповідні вершини так, що ніякі два ребра не мають спільних точок крім інцидентної їм обох вершини. Граф називається планарним, якщо він є ізоморфним плоскому графу.

Гранню плоского графа називається максимальна по включенню множина точок площини, кожна пара яких може бути з'єднана жордановою кривою, що не перетинає ребра графа. Границею грані будемо вважати множину вершин і ребер, що належать цій грані.

ІНДИВІДУАЛЬНІ ЗАВДАННЯ

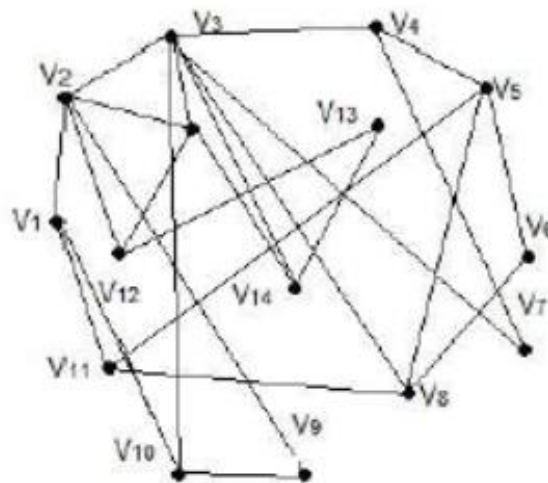
Завдання № 1. Розв'язати на графах наступні 2 задачі:

1. За допомогою алгоритму Дейкстри знайти найкоротший шлях у графі поміж парою вершин V_0 і V^* .

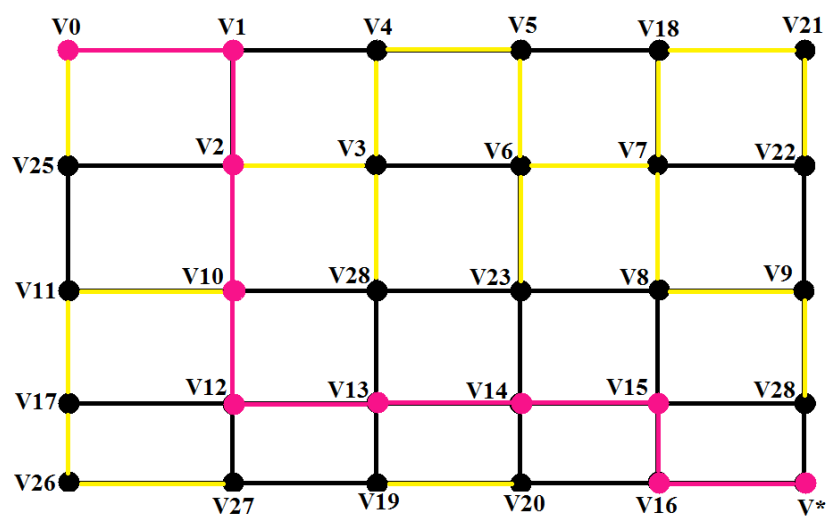


2. За допомогою γ - алгоритма зробити укладку графа у площині, або довести що вона неможлива.

16

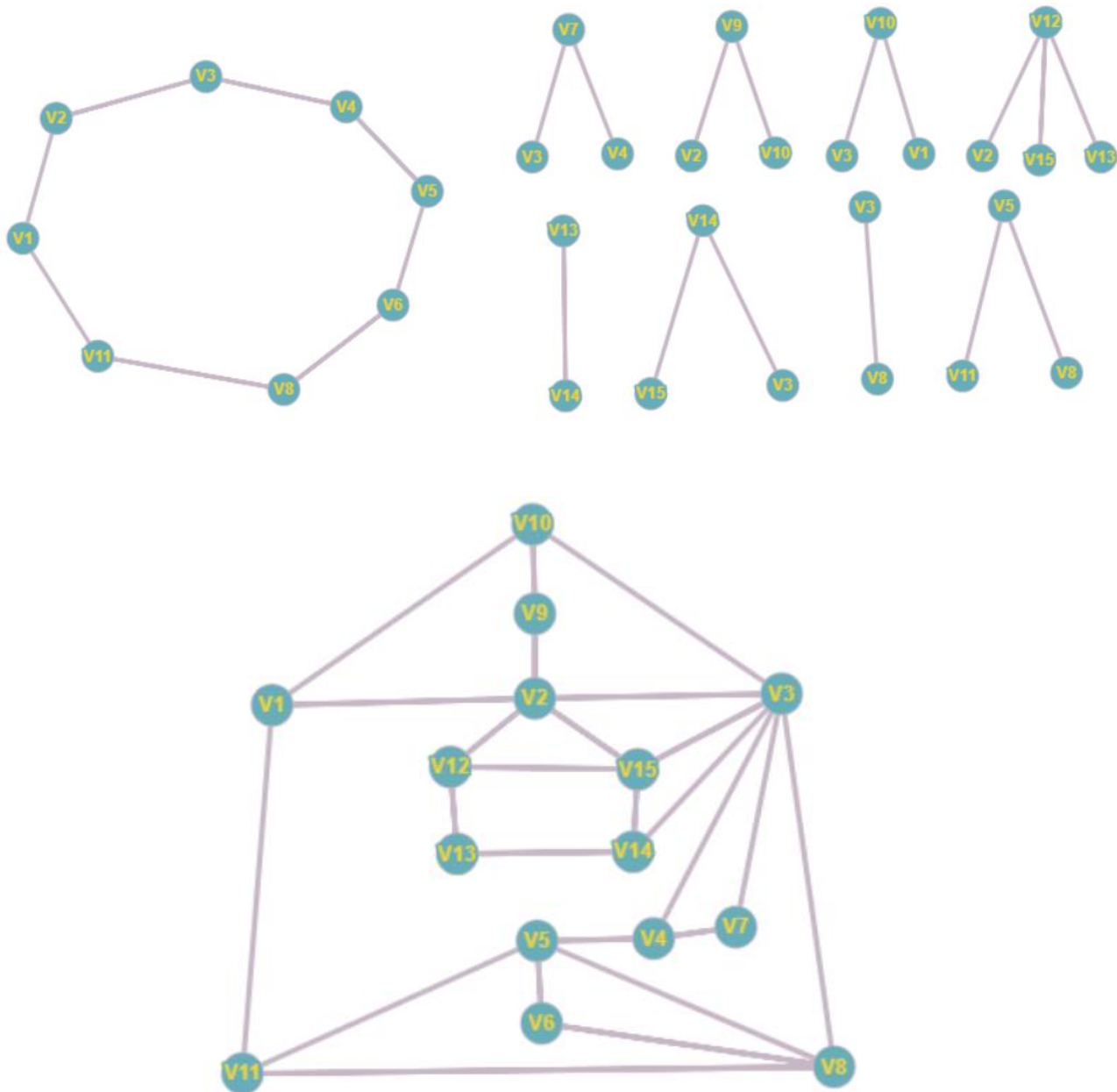


РОЗВ'ЯЗАННЯ



■ - найкоротший шлях

$L(V_0)=0$	$L(V_8)=15$	$L(V_{16})=16$	$L(V_{24})=22$
$L(V_1)=3$	$L(V_9)=17$	$L(V_{17})=12$	$L(V_{25})=6$
$L(V_2)=5$	$L(V_{10})=8$	$L(V_{18})=17$	$L(V_{26})=19$
$L(V_3)=6$	$L(V_{11})=9$	$L(V_{19})=16$	$L(V_{27})=23$
$L(V_4)=10$	$L(V_{12})=10$	$L(V_{20})=19$	$L(V_{28})=14$
$L(V_5)=11$	$L(V_{13})=13$	$L(V_{21})=22$	$L(V^*)=22$
$L(V_6)=12$	$L(V_{14})=14$	$L(V_{22})=23$	
$L(V_7)=14$	$L(V_{15})=15$	$L(V_{23})=17$	



Завдання № 2. Написати програму, яка реалізує алгоритм Дейкстри знаходження найкоротшого шляху між парою вершин у графі.
Протестувати розроблену програму на графі згідно свого варіанту.

```

1 #include<stdio.h>
2
3 #define INFINITY 9999
4 #define MAX 30
5
6 void dejkstra(int G[MAX][MAX], int n, int startnode);
7
8 int main()
9 {
10     int G[MAX][MAX], i, j, n, u;
11     printf("Enter number of vertices:");
12     scanf("%d", &n);
13     for (i = 0; i < n; i++)
14         for (j = 0; j < n; j++)
15             G[i][j] = 0;
16
17     printf("Enter graph:\n");
18     int nd, st, weight;
19
20     for( int z=0;z<49;z++)
21     {
22         printf("1st vertex: ");
23         scanf("%d", &st);
24         while (((st < 0) || (st > n - 1)) && (st != 111))
25         {
26             printf("Retry: ");
27             scanf ("%d", &st);
28         }
29         printf("2nd vertex: ");
30         scanf("%d", &nd);
31         while (((nd < 0) || (nd > n - 1)) && (nd != 111))
32         {
33             printf("Retry: ");
34             scanf("%d", &nd);
35         }
36         printf("Weight: ");
37         scanf("%d", &weight);
38         while (weight <= 0)
39         {
40             printf("Retry: ");
41             scanf("%d", &weight);
42         }
43         G[st][nd] = weight;
44         G[nd][st] = weight;
45     }
46
47     printf("\nStarting node is V0");
48     u=0;
49     dejkstra(G, n, u);
50     printf("\n");
51     return 0;
52 }
53
54 void dejkstra(int G[MAX][MAX], int n, int startnode)
55 {
56
57     int cost[MAX][MAX], distance[MAX], pred[MAX];
58     int visited[MAX], count, mindistance, nextnode, i, j;
59
60     for (i = 0; i < n; i++)
61         for (j = 0; j < n; j++)
62             if (G[i][j] == 0)
63                 cost[i][j] = INFINITY;
64             else
65                 cost[i][j] = G[i][j];
66
67     for(i = 0; i < n; i++)
68     {
69         distance[i] = cost[startnode][i];
70         pred[i] = startnode;
71         visited[i] = 0;
72     }
73
74     distance[startnode] = 0;
75     visited[startnode] = 1;
76     count = 1;
77
78     while (count < n - 1)
79     {
80         mindistance = INFINITY;
81
82         for (i = 0; i < n; i++)
83             if (distance[i] < mindistance && !visited[i])
84             {
85                 mindistance = distance[i];
86                 nextnode = i;
87             }
88
89         visited[nextnode] = 1;
90         for (i = 0; i < n; i++)

```

```

91         if (!visited[i])
92             if (mindistance + cost[nextnode][i] < distance[i])
93             {
94                 distance[i] = mindistance + cost[nextnode][i];
95                 pred[i] = nextnode;
96             }
97         count++;
98     }
99
100     for (i = n-1; i < n; i++)
101         if (i != startnode)
102         {
103             printf("\nDistance from V0 to V* = %d", distance[i]);
104             printf("\nThe shortest way = %d", i);
105
106             j = i;
107             do
108             {
109                 j = pred[j];
110                 printf("<-%d", j);
111             } while (j != startnode);
112         }
113 }

```

Enter number of vertices:30

Enter graph:

1st vertex: 23

2nd vertex: 29

Weight: 7

Starting node is V0

Distance from V0 to V* = 22

The shortest way = 29<-28<-22<-21<-20<-19<-13<-7<-1<-0