МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра систем штучного інтелекту

Лабораторна робота №5

З курсу “Дискретна математика ”

Виконала:
ст.гр. КН-110
Ямнюк Аліна
Викладач:
Мельникова Н.І.

Львів – 2018

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 узнаходженні найкоротших(мається на увазі найоптимальніших завагою) 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 кроцівибирають оптимальний із можливих варіантів.

Задача про найкоротший ланцюг. Алгоритм Дейкстри.

Плоскі і планарні графи

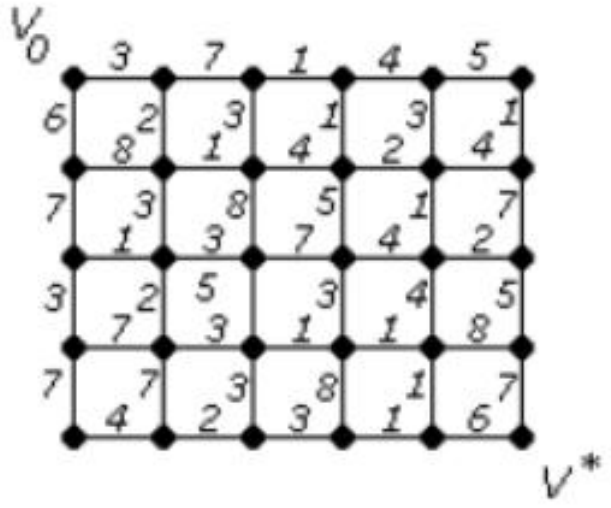
Плоским графом називається граф, вершини якого є точкамиплощини, а ребра – безперервними лініями без самоперетинань, що з'єднують відповідні вершини так, що ніякі два ребра не мають спільних точок крім інцидентної їм обом вершини. Граф називається планарним, якщо він є ізоморфним плоскому графу.

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

ІНДИВІДУАЛЬНІ ЗАВДАННЯ

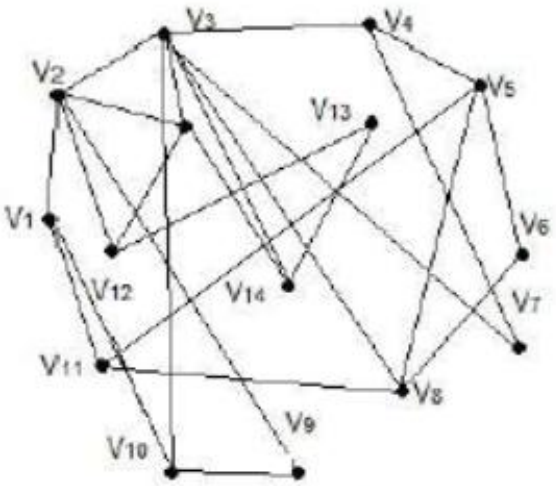
Завдання № 1. Розв'язати на графах наступні 2 задачі:

1. За допомогою алгоритму Дейкстра знайти найкоротший шляху графі поміж парою вершин V0 і V* .

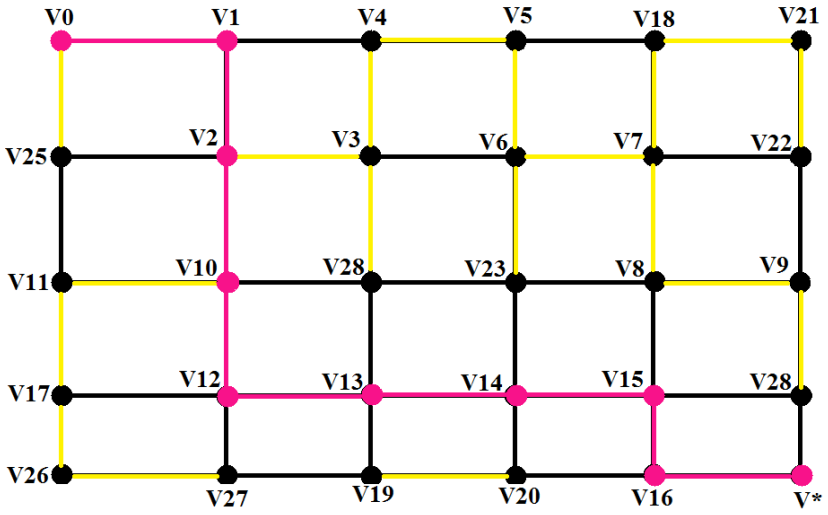


2. За допомогою γ -алгоритма зробити укладку графа у площині,або довести що вона неможлива.

16

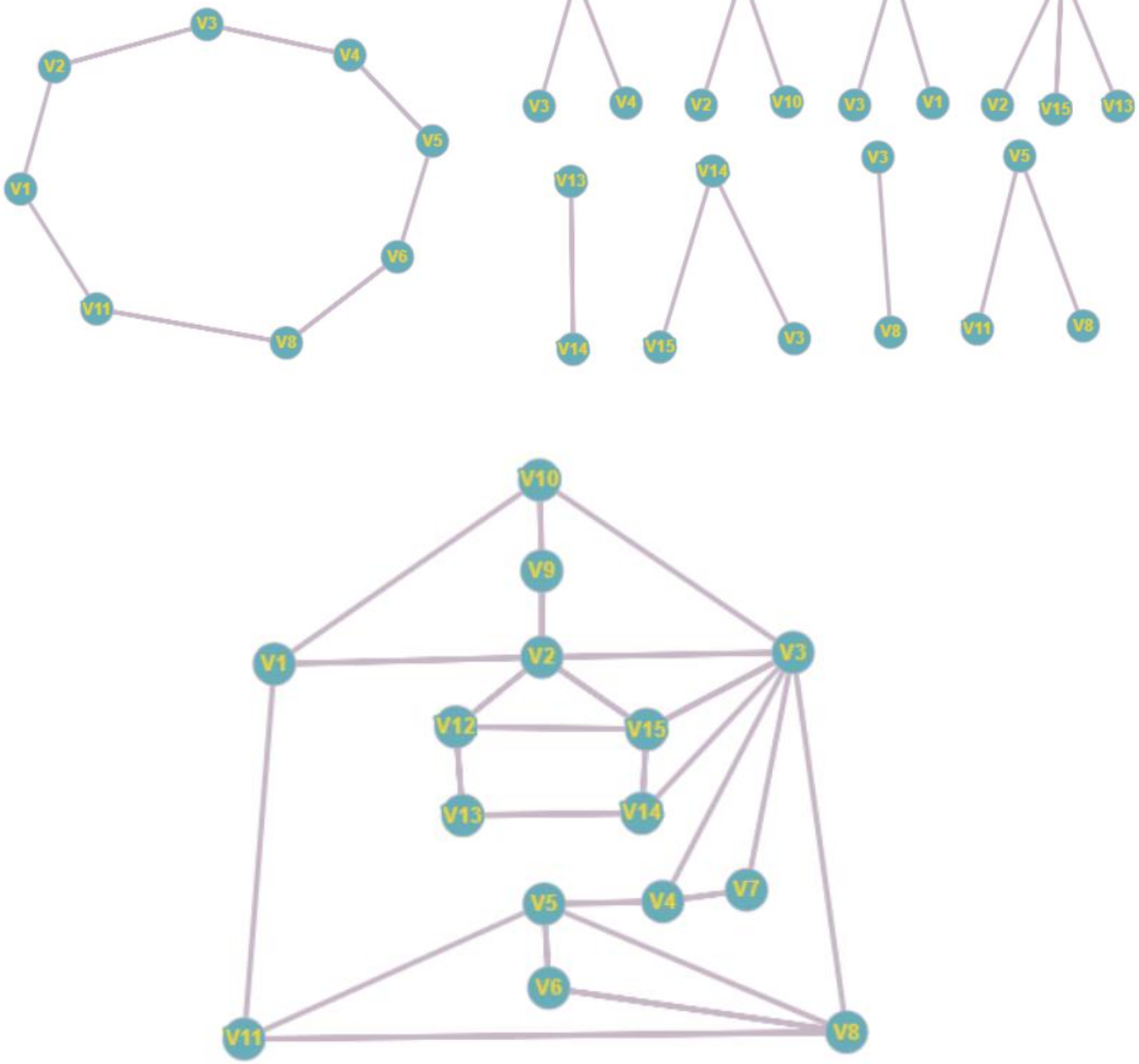


РОЗВ'ЯЗАННЯ



- найкоротший шлях

L(V0)=0	L(V8)=15	L(V16)=16	L(V24)=22
L(V1)=3	L(V9)=17	L(V17)=12	L(V25)=6
L(V2)=5	L(V10)=8	L(V18)=17	L(V26)=19
L(V3)=6	L(V11)=9	L(V19)=16	L(V27)=23
L(V4)=10	L(V12)=10	L(V20)=19	L(V28)=14
L(V5)=11	L(V13)=13	L(V21)=22	**L(V*)=22**
L(V6)=12	L(V14)=14	L(V22)=23	
L(V7)=14	L(V15)=15	L(V23)=17	



Код програми:

```
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 (If it is the end, enter number 111):\n");
18.     int st, nd, weight;
19.     while(1)
20.     {
21.         printf("1st vertex: ");
22.         scanf("%d", &st);
23.         while (((st < 0) || (st > n - 1)) && (st != 111))
24.         {
25.             printf("Retry: ");
26.             scanf ("%d", &st);
27.         }
28.         if (st == 111)
29.             break;
30.         printf("2nd vertex: ");
31.         scanf("%d", &nd);
32.         while (((nd < 0) || (nd > n - 1)) && (nd != 111))
33.         {
34.             printf("Retry: ");
35.             scanf("%d", &nd);
36.         }
37.         if (nd == 111)
38.             break;
39.         printf("Weight: ");
40.         scanf("%d", &weight);
41.         while (weight <= 0)
42.         {
43.             printf("Retry: ");
44.             scanf("%d", &weight);
45.         }
46.         if (weight == 111)
47.             break;
48.         if ((st != 111) && (nd != 111) && (weight != 111))
49.         {
50.             G[st][nd] = weight;
51.             G[nd][st] = weight;
52.         }
53.     }
54.     printf("\nEnter the starting node:");
55.     scanf("%d", &u);
56.     dejkstra(G, n, u);
57.     printf("\n");
58.     return 0;
59. }
60.
61. void dejkstra(int G[MAX][MAX], int n, int startnode)
62. {
63.
64.     int cost[MAX][MAX], distance[MAX], pred[MAX];
65.     int visited[MAX], count, mindistance, nextnode, i, j;
66.
67.     for (i = 0; i < n; i++)
68.         for (j = 0; j < n; j++)
69.             if (G[i][j] == 0)
70.                 cost[i][j] = INFINITY;
```

```
        else
            cost[i][j] = G[i][j];

    for(i = 0; i < n; i++)
    {
        distance[i] = cost[startnode][i];
        pred[i] = startnode;
        visited[i] = 0;
    }

    distance[startnode] = 0;
    visited[startnode] = 1;
    count = 1;

    while (count < n - 1)
    {
        mindistance = INFINITY;

        for (i = 0; i < n; i++)
            if (distance[i] < mindistance && !visited[i])
            {
                mindistance = distance[i];
                nextnode = i;
            }

            visited[nextnode] = 1;
            for (i = 0; i < n; i++)
                if (!visited[i])
                    if (mindistance + cost[nextnode][i] < distance[i])
                            {
                                distance[i] = mindistance + cost[nextnode][i];
                                pred[i] = nextnode;
                            }
                count++;
            }

            for (i = 29; i < n; i++)
                if(i != startnode)
                {
                    printf("\nDistance from 0 to %d = %d", i, distance[i]);
                    printf("\nThe shortest way = %d", i);

                    j = i;
                    do
                    {
                        j = pred[j];
                        printf("<-%d",j);
                    } while (j != startnode);
                }
        }
```

Результат роботи програми:

```
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
```