МІНІСТЕРСТВО ОСВІТИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСЕТЕТ
«ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра штучного інтелекту

Лабораторна робота №5

З дисципліни

«Дискретна математика»

Виконав:

Студент групи КН-115

Курило Валентин

Викладач:

Мельникова Н.І.

Львів-2019

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

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

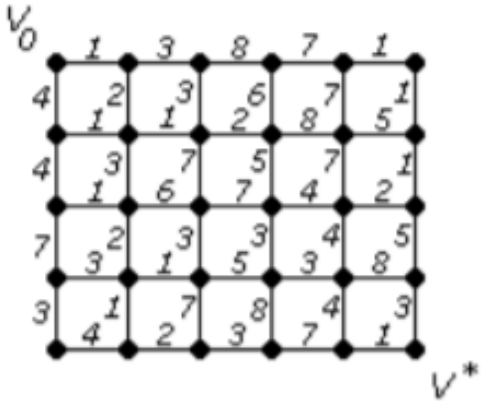
Варіант – 12.

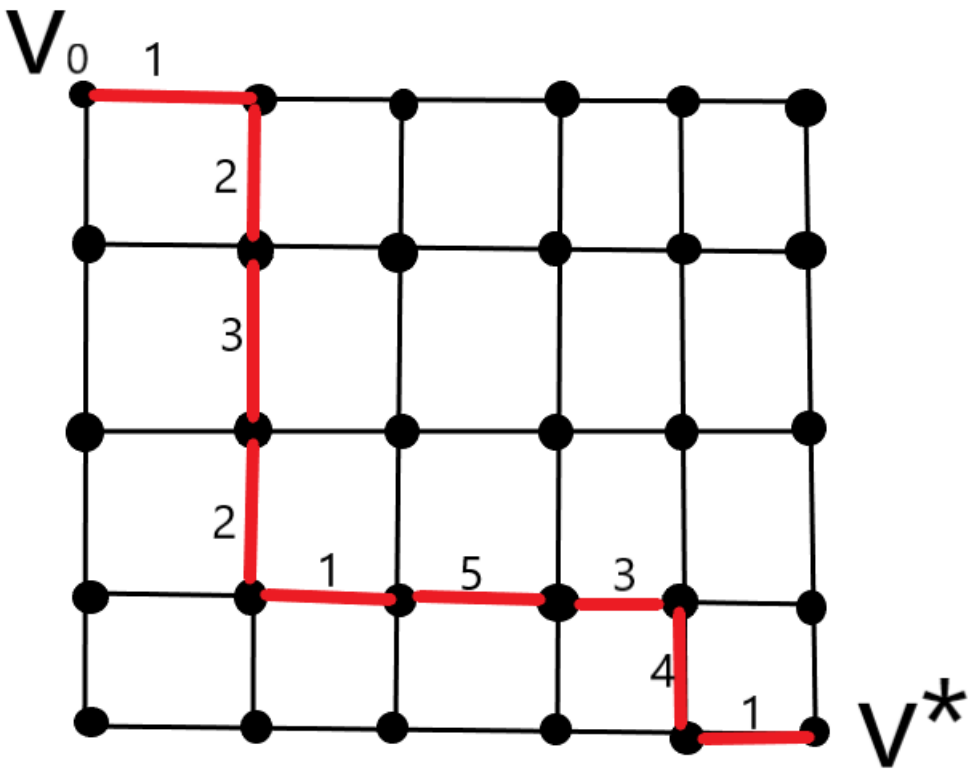
ІНДИВІДУАЛЬНІ ЗАВДАННЯ Завдання № 1

Розв'язати на графах наступні 2 задачі:

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

12

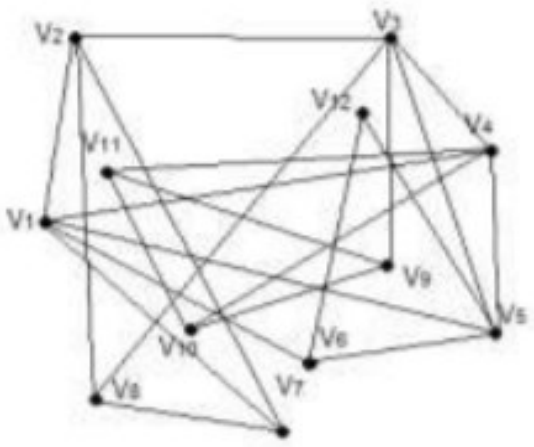




Найкоротший шлях від v_0 до v* становить 22.

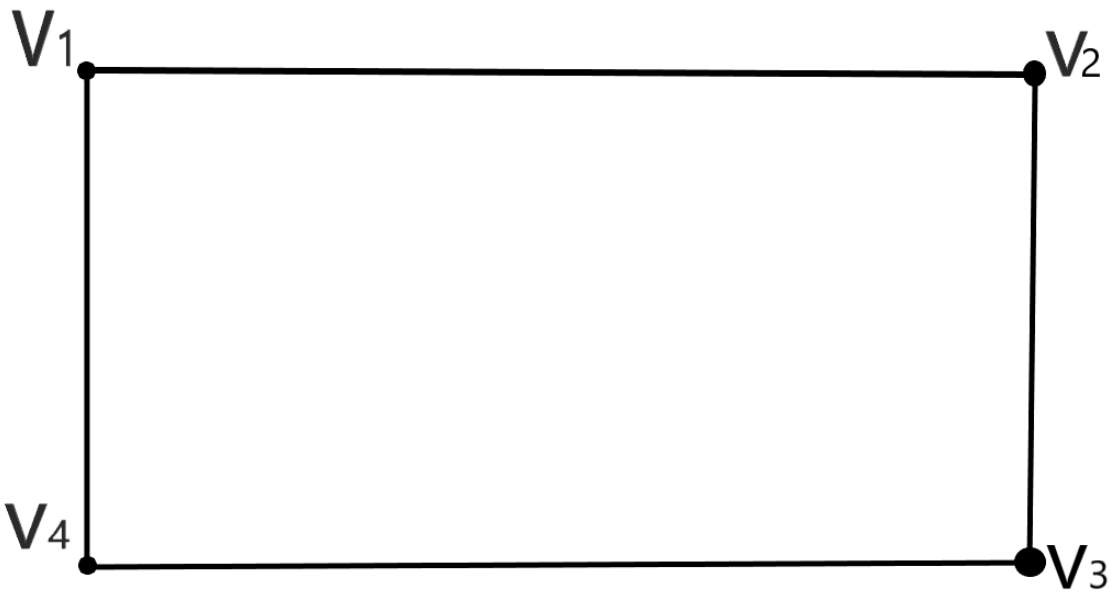
2. За допомогою γ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.

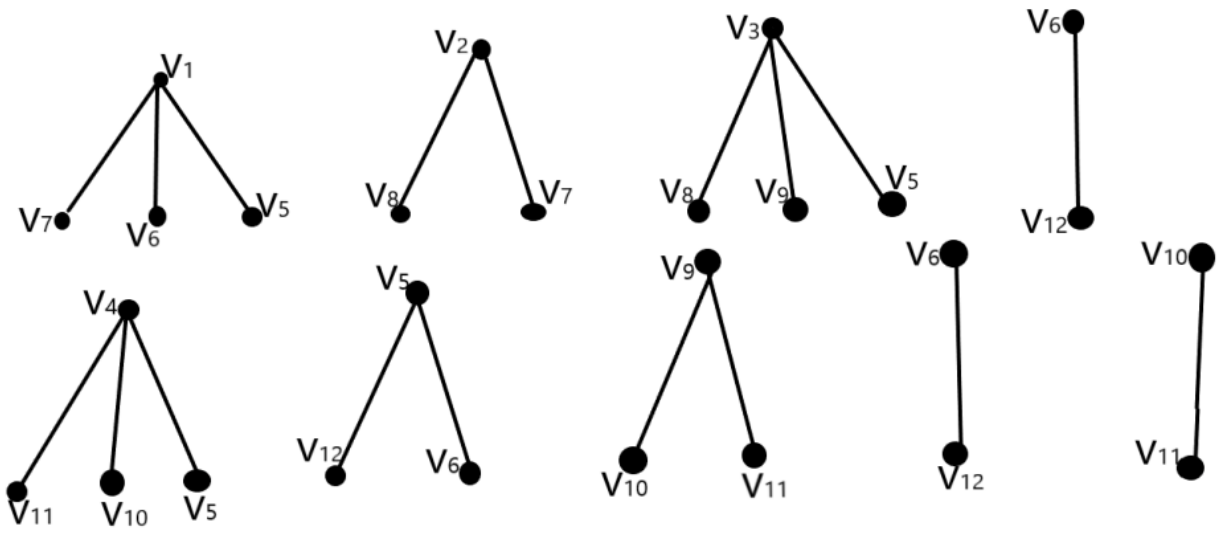
12



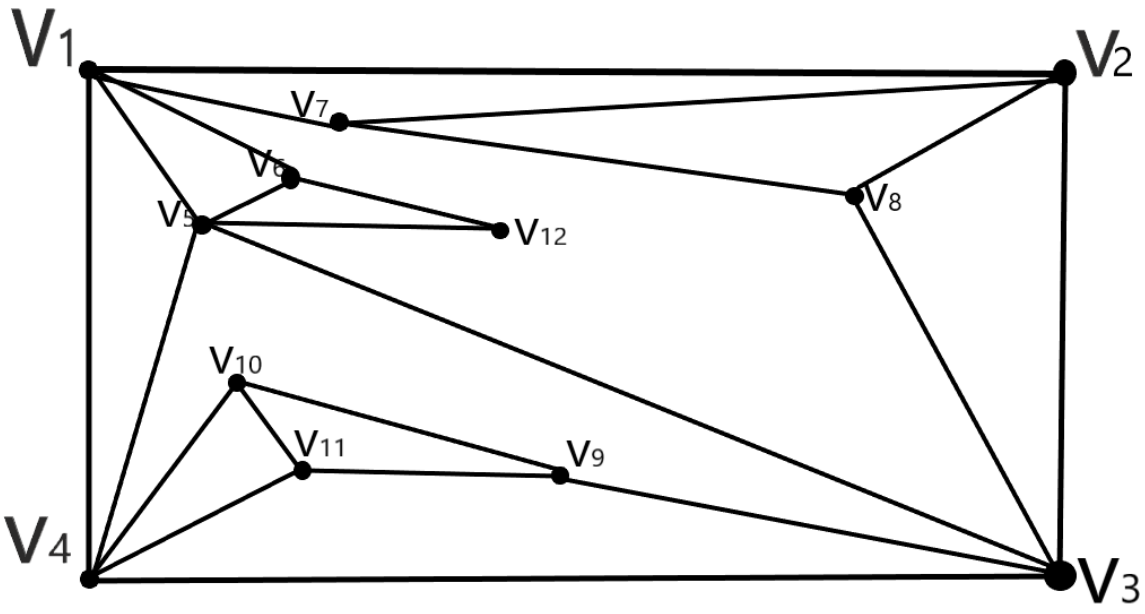
Розв’язання:

Для розв’язання беру цикл v_1,v_2,v_3,v_4


V1
V2
V4
V3


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

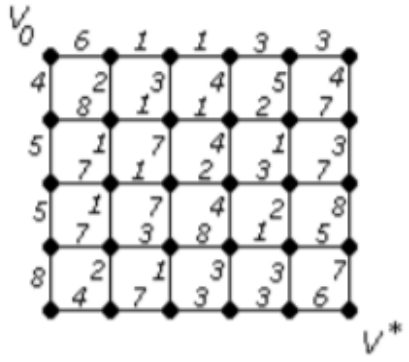
Результатом γ -алгоритма буде:



Завдання №2.

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

12



Програма:

```
#include <iostream>
#include <limits.h>
#include <stdio.h>
#define V 30
using namespace std;
int minDistance(int dist[], bool sptSet[])
{
    int min = INT_MAX, min_index;

    for (int v = 0; v < V; v++)
        if (sptSet[v] == false && dist[v] <= min)
            min = dist[v], min_index = v;

    return min_index;
}

int printSolution(int dist[])
{
    printf("Vertex \t\t Distance from Source\n");
    for (int i = 0; i < V; i++)
        printf("%d \t\t %d\n", i, dist[i]);
}

void dijkstra(int graph[V][V], int src)
{
    int dist[V];
    bool sptSet[V];
    for (int i = 0; i < V; i++)
        dist[i] = INT_MAX, sptSet[i] = false;

    dist[src] = 0;

    for (int count = 0; count < V - 1; count++) {

        int u = minDistance(dist, sptSet);

        sptSet[u] = true;

        for (int v = 0; v < V; v++)

            if (!sptSet[v] && graph[u][v] && dist[u] != INT_MAX
                && dist[u] + graph[u][v] < dist[v])
```

```
    printSolution(dist);
}


int main()
{
    int graph[V][V] = { { 0, 6, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 6, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 1, 0, 1, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 1, 0, 3, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 4, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 2, 0, 0, 0, 0, 8, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 3, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 4, 0, 0, 0, 0, 1, 0, 2, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 5, 0, 0, 0, 0, 2, 0, 7, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 7, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 1, 0, 2, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 2, 0, 3, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 3, 0, 7, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 7, 0, 3, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 3, 0, 8, 0, 0, 0, 0, 1, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 8, 0, 1, 0, 0, 0, 0, 3, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 1, 0, 5, 0, 0, 0, 0, 3, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 3, 7, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 4, 0, 7, 0, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 7, 0, 3, 0, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 3, 0, 3, 0, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 3, 0, 6, 0},
      { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 6, 0, 0},
       };

    dijkstra(graph, 0);

    return 0;
}
```

Результат:

```
Vertex          Distance from Source
0               0
1               6
2               7
3               8
4               11
5               14
6               4
7               8
8               9
9               10
10              12
11              18
12              9
13              9
14              10
15              12
16              13
17              20
18              14
19              10
20              13
21              16
22              15
23              20
24              16
25              12
26              14
27              17
28              18
29              24
```