**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

**Кафедра систем штучного інтелекту**

# Лабораторна робота №5

з дисципліни

«Дискретна математика»

**Виконала:**

Студентка КН-112

Пихней Вероніка

**Викладач:**

Мельникова Н.І.

Львів – 2019

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

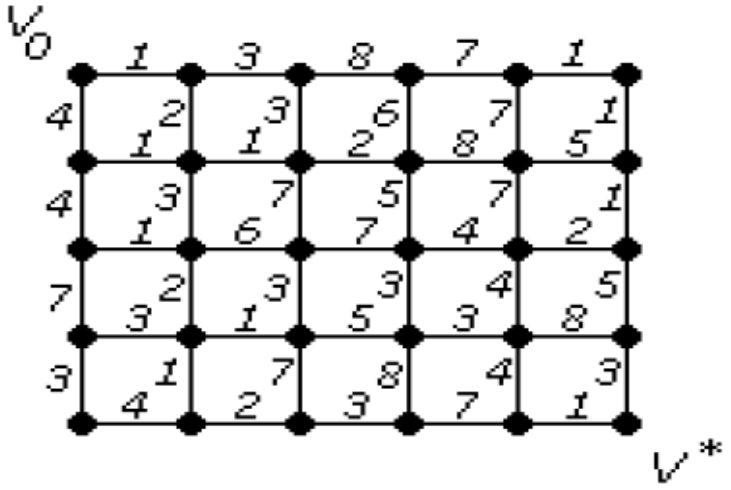
**Мета роботи:** набуття практичних вмінь та навичок з використання алгоритму Дейкстри.

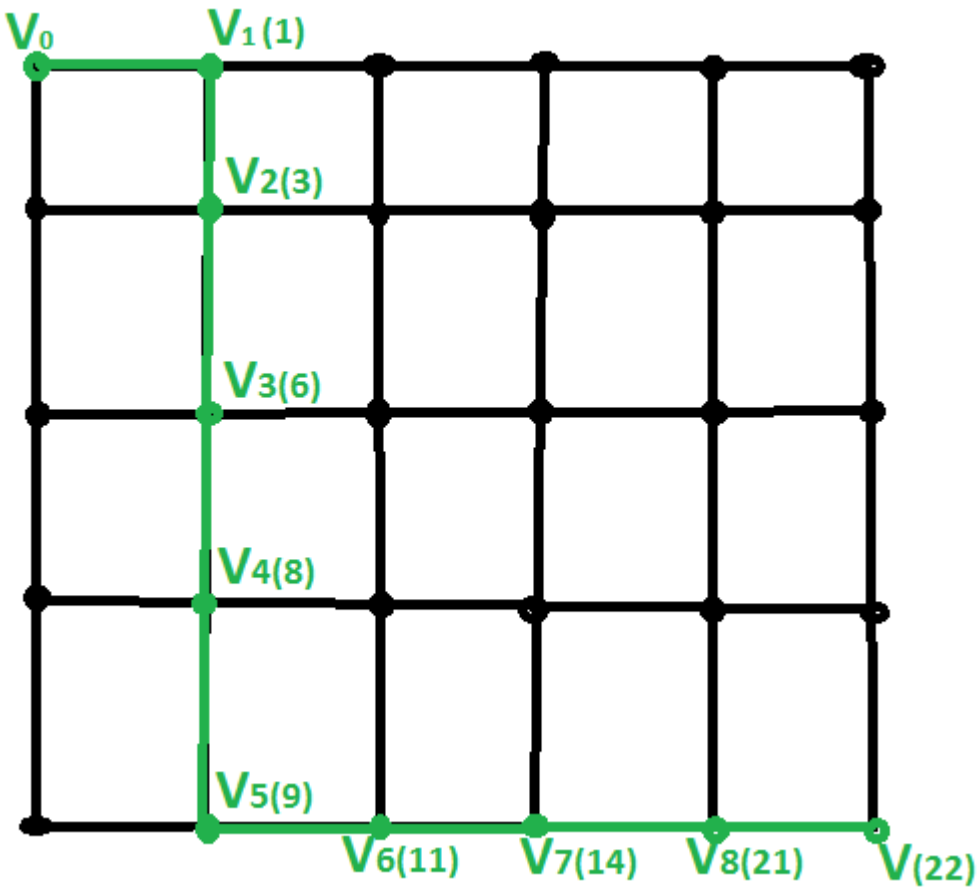
## Варіант№12

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

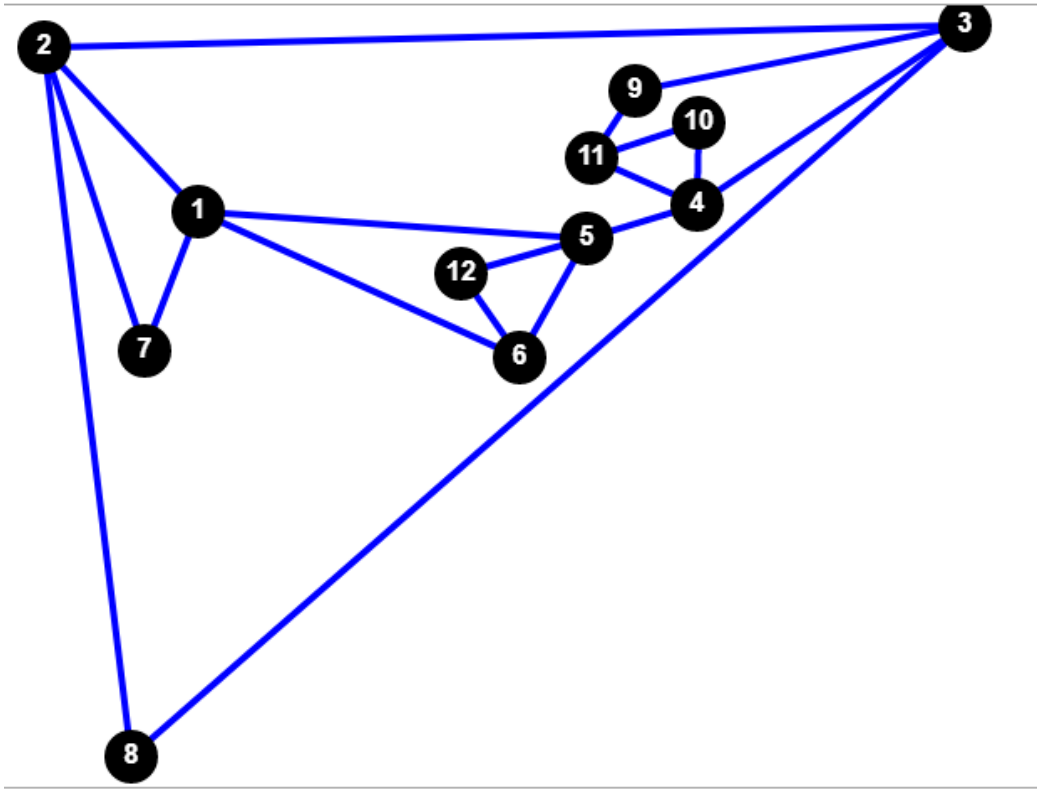
**1.** За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин $V_0$ і $V^*$.

**12**



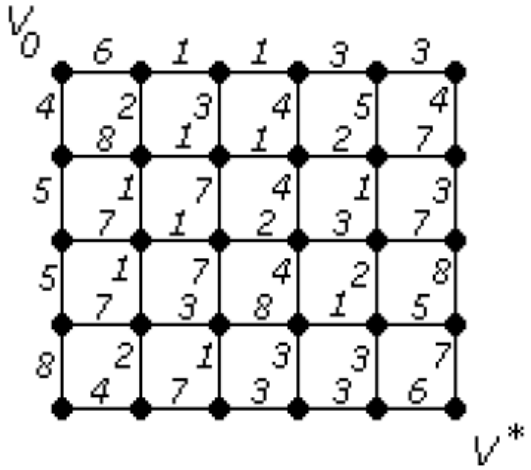



**2.** За допомогою $\gamma$ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.




Програма: Алгоритм  Дейкстри ,знаходження найкоротшого шляху між парою першин у графі.Протестувати розроблену програму на графі згідно свого варіанту.

**12**



```cpp
#include <iostream>
#include <fstream>
#include <cstring>
#include <stdio.h>


using namespace std;

const int Number_of_point=30;
const int Infinity=1000;

struct Vertices
    {
        int Weight=Infinity;
        int From=0;
        bool Fixed=false;

    };

int main()
{

      string Rebro;
```

```cpp
   char Rebro_char[100];
   char *pch;
   int Matrix[Number_of_point][Number_of_point];

   Vertices Vertex[Number_of_point];
   Vertex[0].Weight=0;
   Vertex[0].Fixed=true;
   int current_vertex=0;


   //занулюємо матрицю
   for(int i=0;i<Number_of_point;i++){for(int j=0;j<Number_of_point;j++) Matrix[i][j]=0;}
   int i=0, j=0, row=0;
   ifstream File("MATRIX.txt");
   while(getline(File,Rebro))
   {
      strcpy(Rebro_char, Rebro.c_str());
      char *pch = strtok(Rebro_char, " ");
      while (pch!= NULL)
      {
          if(row%2==0)
          {
              Matrix[i][i+1]=(int)*pch-48;
              Matrix[i+1][i]=(int)*pch-48;
              i++;
          }
          else
          {
              Matrix[j][j+6]=(int)*pch-48;
              Matrix[j+6][j]=(int)*pch-48;
```

```
            j++;
        }

        pch = strtok(NULL, " ");
    }
    row++;
    if(row%2==0)i++;
  }

//друкуємо матрицю
for(int i=0;i<Number_of_point;i++){for(int j=0;j<Number_of_point;j++) cout<<Matrix[i][j]<<" ";cout<<endl;}


----------------------------------------------------------------------------------------------------------------------------------------------------------*/


//Алгоритм Дейкстрі
while(Vertex[Number_of_point-1].Fixed!=true)
{
    //знаходимо сумiжні вершини для current_vertex
    int connected_vertices[Number_of_point-1];
    int connected_vertices_index=0;
    for(int i=0; i<Number_of_point; i++)if(Matrix[current_vertex][i]!=0){connected_vertices[connected_vertices_index]=i;connected_vertic

    //для всіх сумiжних вершин знаходимо шлях(вагу) до них
    for(int i=0;i<connected_vertices_index;i++)
    {
        if((Vertex[current_vertex].Weight+Matrix[current_vertex][connected_vertices[i]])<Vertex[connected_vertices[i]].Weight)
        {
            Vertex[connected_vertices[i]].Weight=Vertex[current_vertex].Weight+Matrix[current_vertex][connected_vertices[i]];
            Vertex[connected_vertices[i]].From=current_vertex;
        }
    }
```

```
    //Знаходимо вершину з мінімальною вагою і зафіксовуємо її
    int minimum=Infinity;
    int aaa=0;
    for(int i=0;i<Number_of_point;i++)
        if((Vertex[i].Weight<minimum)&&(Vertex[i].Fixed!=true))
            {
                minimum=Vertex[i].Weight;
                aaa=i;
            }
    Vertex[aaa].Fixed=true;

    //продовжуємо йти вiд вершини з мiнiмальною вагою
    current_vertex = aaa;

}


//Друк маршруту
int track[Number_of_point];
int track_index=0;
current_vertex=Number_of_point-1;
track[track_index]=current_vertex;
track_index++;
while(current_vertex>0)
{
   track[track_index]=Vertex[current_vertex].From;
   track_index++;
   current_vertex=Vertex[current_vertex].From;
}

cout<<endl<<"Min Track ("<<Vertex[Number_of_point-1].Weight<<") : ";
for(int i=track_index-1;i>=0;i--)
{
```

```
    cout<<track[i];
    if(i!=0)cout<<"-";else cout<<endl;
}

return 0;
```


Висновок:

Набула практичних вмінь та навичок з використання алгоритму Дейкстри.