МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА

Кафедра систем штучного інтелекту

Лабораторна робота №5

з дисципліни «Дискретна математика»

Виконала:

студентка групи КН-115

Попів Христина

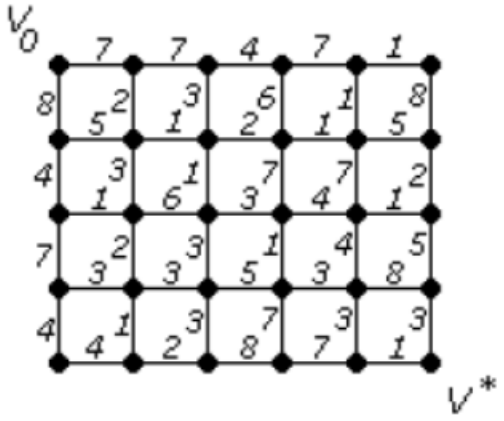
Викладач:

Мельникова Н.І.

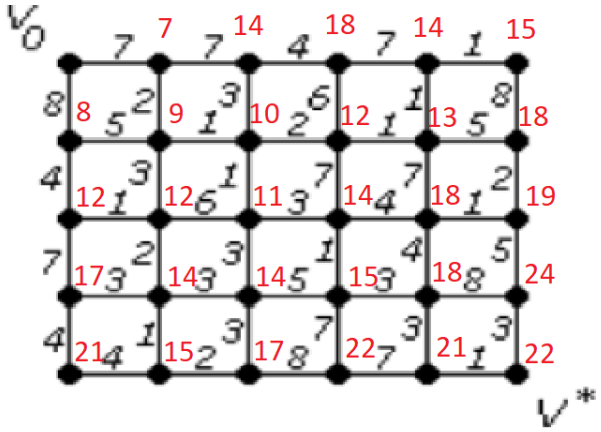
Львів – 2019 р.

Варіант 10

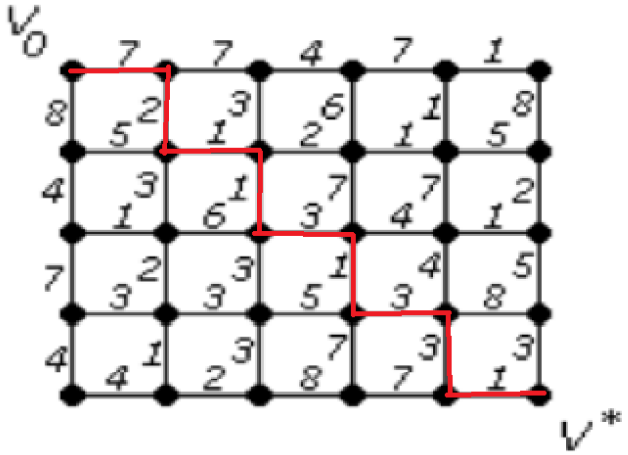
1.За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин V0 і V* .



Розв'язання



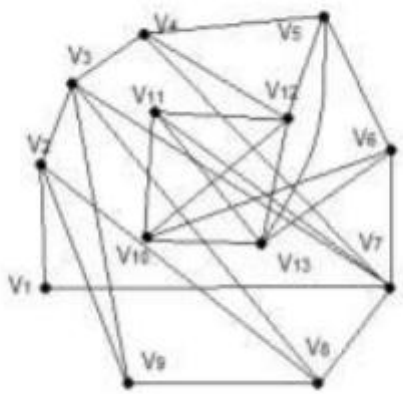
Найкоротший шлях:



Отже, найкоротша відстань від вершини 1 до вершини 30 рівна **22**.

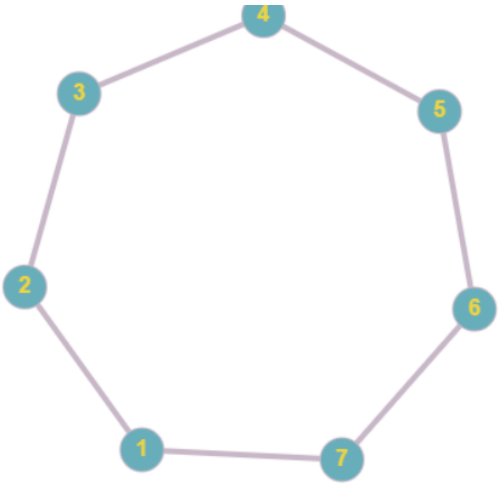
2.За допомогою γ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.

10

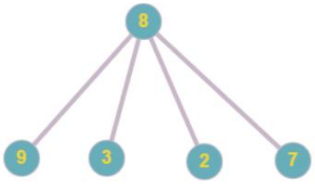


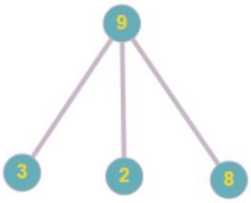
Розв’язання

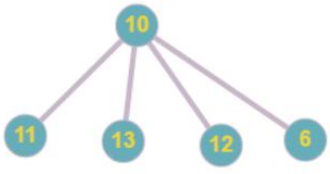
Вибираємо з даного графа простий цикл:

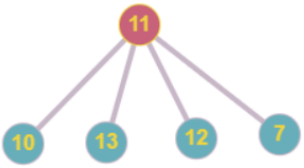


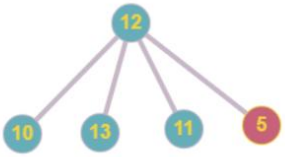
Розбиваємо граф на сегменти:

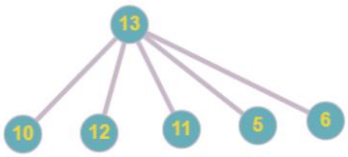


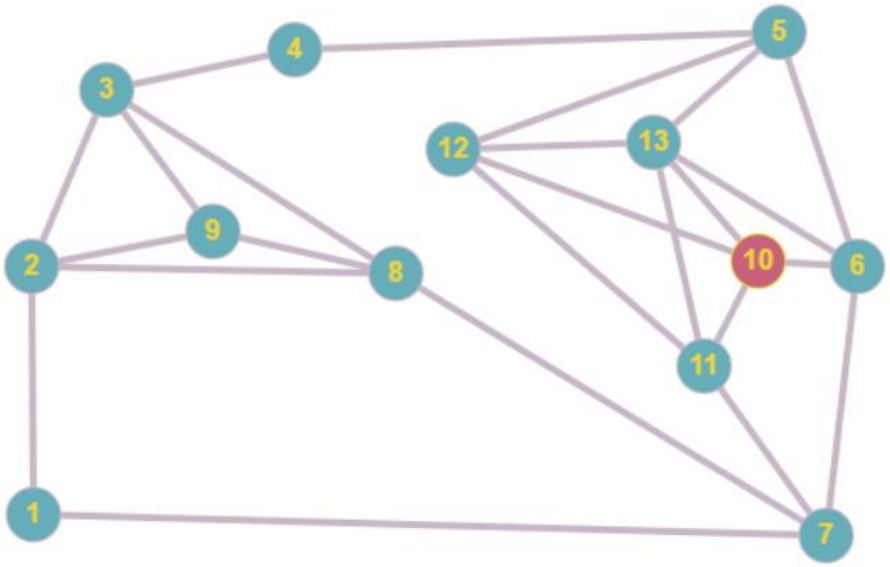






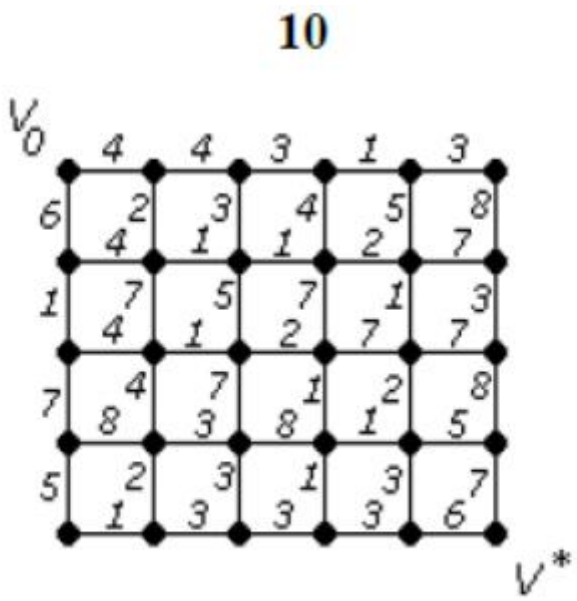






Отже , укладку графа у площині є неможливою , оскільки існує перетен ребер.

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



Текст програми:

```
#include<iostream>
                    #define INFINITY 999999
                    #define MAX 30
                    using namespace std;
                    void Algo_Deikstra(int Graph[MAX][MAX], int number_of_dots, int start) {
                            int distance[MAX];// 1 рядок з таблички
                            int pred[MAX];//для позначання шляху
                            int visited[MAX];
                            int counter, mindistance, next_dot;
```

```
	for (int i = 0; i < number_of_dots; i++) {//задали нескінечності всім нульовим значенням
		for (int j = 0; j < number_of_dots; j++) {
			if (Graph[i][j] == 0) {
				Graph[i][j] = INFINITY;
			}
		}
	}
	for (int i = 0; i < number_of_dots; i++) {
		distance[i] = Graph[start][i];//записали перший ряд в табличку
		pred[i] = start;//заповнили масив пройденною точкою
		visited[i] = 0;//
	}
	distance[start] = 0;//вага першого заходу
	visited[start] = 1;//вказали значення по індексу для того , щоб відслідковувати точки (індекси це точки)
	counter = 0;
	while (counter < number_of_dots - 1) {
		mindistance = INFINITY;
		for (int i = 0; i < number_of_dots; i++) {//підбираємо мінімальну найменшу відстань у всьому ряді
			if ((distance[i] < mindistance) && (visited[i]==0)) {
				mindistance = distance[i];
				next_dot = i;//наступна точка шляху
			}
		}

		visited[next_dot] = 1;//вказали , що пройшли цю точку

		for (int i = 0; i < number_of_dots; i++) {
			if (visited[i]==0) {
				if (mindistance + Graph[next_dot][i] < distance[i]) {//всіх точок попередню вагу
					distance[i] = mindistance + Graph[next_dot][i];
					pred[i] = next_dot;//записуємо шлях
				}
			}
		}
		counter++;
	}
	int I;
	for (int i = 0; i < number_of_dots; i++) {
		if (i != start) {
			if (i == 29) {
				cout << endl << "Way to dot N" << i+1 << " = " << distance[i] << endl;
				cout << "Way: " << i + 1;
				I = i;
				do {
					I = pred[I];
					cout << " - " << I + 1;
				} while (I != start);
			}
		}
	}
}
int main() {
	int Graph[MAX][MAX] = { {0,4,0,0,0,0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{4,0,4,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{0,4,0,3,0,0,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{0,0,3,0,1,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{0,0,0,1,0,3,0,0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{0,0,0,0,3,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
	{6,0,0,0,0,0,0,4,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
```

```
        {0,2,0,0,0,0,4,0,1,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
        {0,0,3,0,0,0,0,1,0,1,0,0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0},
        {0,0,0,4,0,0,0,0,1,0,2,0,0,0,0,7,0,0,0,0,0,0,0,0,0,0,0,0},
        {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},
        {0,0,0,0,0,8,0,0,0,0,7,0,0,0,0,0,0,3,0,0,0,0,0,0,0,0,0,0},
        {0,0,0,0,0,0,1,0,0,0,0,0,0,4,0,0,0,0,7,0,0,0,0,0,0,0,0,0},
        {0,0,0,0,0,0,0,4,0,0,0,0,4,0,1,0,0,0,0,4,0,0,0,0,0,0,0,0},
        {0,0,0,0,0,0,0,0,5,0,0,0,0,1,0,2,0,0,0,0,7,0,0,0,0,0,0,0},
        {0,0,0,0,0,0,0,0,0,7,0,0,0,0,2,0,7,0,0,0,0,1,0,0,0,0,0,0},
        {0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,7,0,7,0,0,0,0,2,0,0,0,0,0},
        {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},
        {0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,0,0,8,0,0,0,0,5,0,0,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,8,0,3,0,0,0,0,2,0,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,3,0,8,0,0,0,0,3,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,8,0,1,0,0,0,0,1},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,1,0,5,0,0,0,0,3},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,8,0,0,0,0,5,0,0,0,0,0,7},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,0,0,0,0,0,1,0,0,0,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,1,0,3,0,0,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,3,0,3,0,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,3,0,3,0},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,3,0,6},
        {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,6,0},
        };
        int number_of_dots = 30;
        int start = 0;//початкова точка (нумерація починається з 0)
        Algo_Deikstra(Graph, number_of_dots, start);
}
```

Результат:

Висновок: на цій лабораторній роботі я навчився знаходити найкоротший шлях за алгоритмом Дейкстри та укладати граф за допомогою алгоритму у-укладання графа.