**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**Кафедра систем штучного інтелекту**

**Лабораторна робота**

**№5**

з дисципліни

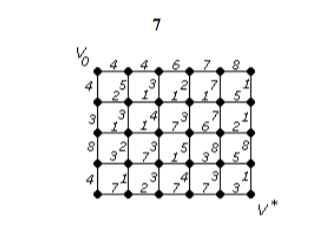
«Дискретна математика»

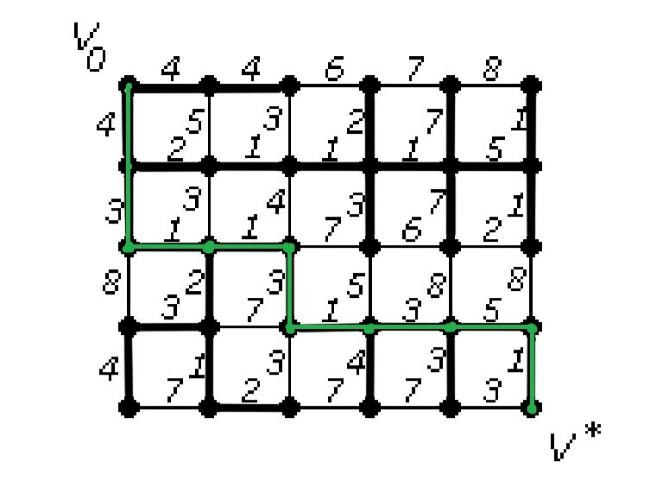
**Виконав:**

студент групи КН-108

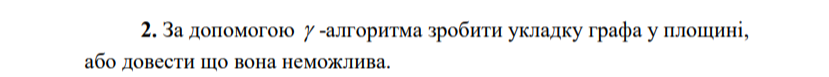
Пулик Максим

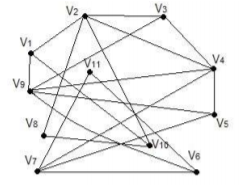
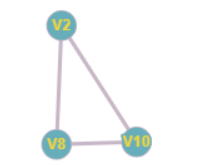
Львів – 2018 р.

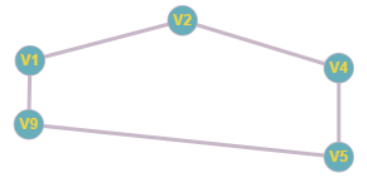
**Частина 1**

Завдання 1

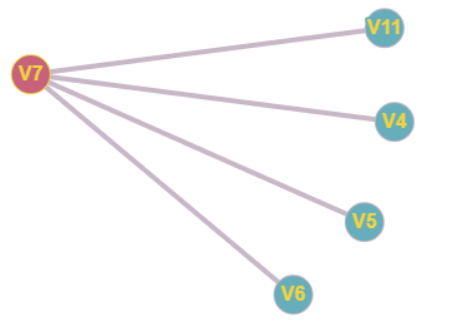
Шуканий ланцюг:{V0 V\*} вага = 22

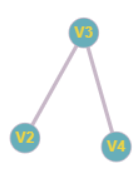
Завдання 2

Розбиваємо на підграфи:

1) 2)

3) 4)

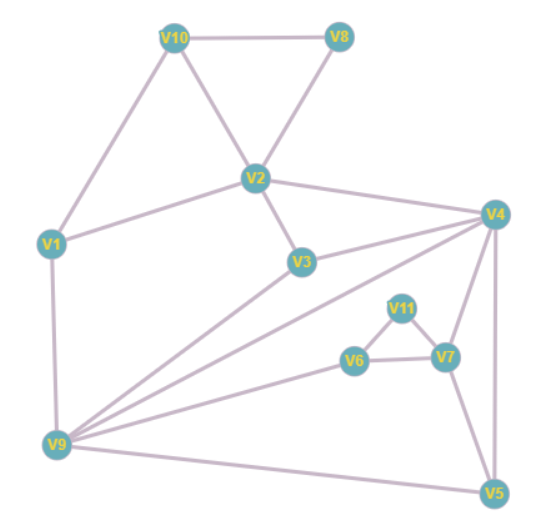


 5)

7)

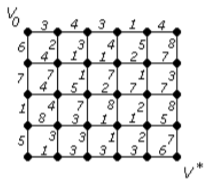
 6)

Загалом отримуємо граф, що укладається в площині, вигляду:



Частина 2

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



#include <stdio.h>

#include <stdlib.h>

#define SIZE 30 // Кількість вершин, розмір матриці

int Array[SIZE][SIZE]; // Масив матриці графа

int distance[SIZE]; // Масив мінімальних відстаней

int vertex[SIZE]; // Масив вершин

// Заповнення матриці нулями

void zeroArray()

{

for(int i=0; i<SIZE; i++)

for(int j=0; j<SIZE; j++)

Array[i][j]=0;

}

// Запис дуг в матрицю

void enterEdges()

{

printf("Enter edges:\n");

int r, c, n;

for(int i=0; i<49; i++)

{

scanf("%d %d %d", &r, &c, &n);

Array[r-1][c-1]=n;

Array[c-1][r-1]=n;

}

}

// Вивід матриці графа

void printArray()

{

for(int i=0; i<SIZE; i++)

{

for(int j=0; j<SIZE; j++)

{

printf("%d ", Array[i][j]);

}

printf("\n");

}

}

// Ініціалізація масивів відстаней та вершин

void initArray()

{

for(int i=0; i<SIZE; i++)

{

distance[i]=10000; // Всі відстані поки що є невизначеними, тому 10000

vertex[i]=1; // Всі вершини є необійденими, мають значення 1

}

distance[0]= 0; // Відстань до першої вершини 0

}

// Вивід найкоротших відстаней до вершин 1-30

void printDistance()

{

printf("\nShortest path to every vertex(1-30): \n");

for(int i=0; i<SIZE; i++)

{

printf("%d) %d; ", i+1, distance[i]);

if((i+1)%5==0 && i!=0)

printf("\n");

}

}

// Головна функція

int main(void)

{

int temp; // Тимчасова змінна для запису матриці графа

int minindex; // Змінна для ітерацій циклу основного алгоритму

int min;

zeroArray(); // Заповнення матриці нулями

enterEdges(); // Ввід ребер та їхньої ваги

printArray(); // Вивід матриці

initArray(); // Ініціалізація масивів відстаней та вершин

// Основний алгоритм

do

{

minindex=10000;

min=10000;

for (int i=0; i<SIZE; i++)

{

if ((vertex[i]==1) && (distance[i]<min)) // Якщо вершину ще не обійшли і вага менша за min

{

min=distance[i]; // Встановлюємо змінній min мінімальне значення

minindex=i; // Визначаємо позицію мінімальної відстані

}

}

if(min!=10000) // Якщо попередня умова виконалась

{

for(int i=0; i<SIZE; i++)

{

if(Array[minindex][i]>0)

{

temp=min+Array[minindex][i];

if(temp<distance[i])

{

distance[i]=temp; // Відстань записуємо в масив відстаней

}

}

}

vertex[minindex]=0; // Помічаємо вершину пройденою

}

}

while(minindex < 10000); // Допоки не знайдемо всіх відстаней

printDistance(); // Вивід найкоротших відстаней

// Відновлення шляху

int ver[SIZE]; // Масив відвіданих вершин

int end = 29; // Індекс кінцевої вершини 30-1=29

ver[0] = end + 1; // Перший елемент -- остання вершина

int k = 1; // Індекс попередньої вершини

int weight = distance[end]; // Вага кінцевої вершини

while (end > 0) // Поки не дійшли до початкової вершини

{

for(int i=0; i<SIZE; i++) // Переглядаємо всі вершини

if (Array[end][i] != 0) // Якщо зв'язок є

{

temp = weight - Array[end][i]; // Визначаємо вагу шляху з попередньої вершини

if (temp == distance[i]) // Якщо вага співпала з вирахуваною

{ // Значить, з цієї вершини був здійснений перехід

weight = temp; // Зберігаємо нову вагу

end = i; // Зберігаємо попередню вершину

ver[k] = i + 1; // І записуємо її в масив

k++;

}

}

}

// Вивід шляху (початкова вершина опинилась в кінці масиву з k елементів)

printf("\nOutput of the shortest path:\n");

for(int i = k-1; i>=0; i--)

printf("%3d ", ver[i]);

scanf("%d", &temp);

return 0;

}

