МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №5

з дисципліни
«Дискретна математика»

Виконав:
студент групи КН-114
Брила Ярослав

Викладач:
Мельникова.Н.І

Львів – 2019 р

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 у знаходженні найкоротших(мається на увазі найоптимальніших за вагою) 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 кроці вибирають оптимальний із можливих варіантів.

Задача про найкоротший ланцюг. Алгоритм Дейкстри.

Дано n-вершинний граф $G=(V,E)$, у якому виділено пару вершин $v_0, v^* \in V$, і кожне ребро зважене числом $w(e) \geq 0$. Нехай $X=\{x\}$ – множина усіх простих ланцюгів, що з'єднують v_0 з v^*, $x=(V_x, E_x)$. Цільова функція $F(x)=\sum_{e \in E_x} w(e) \to \min$. Потрібно знайти найкоротший ланцюг, тобто $x_0 \in X:\ F(x_0)=\min_{x \in X} F(x)$

Перед описом алгоритму Дейкстри подамо визначення термінів "k-а найближча вершина і "дерево найближчих вершин". Перше з цих понять визначається індуктивно так.

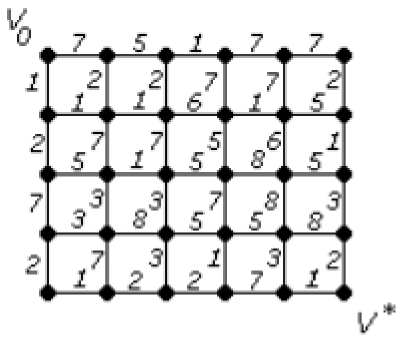
1-й крок індукції. Нехай зафіксовано вершину x_0, E_1 – множина усіх ребер $e \in E$, інцидентних v_0. Серед ребер $e \in E_1$ вибираємо ребро $e(1) = (v_0, v_1)$, що має мінімальну вагу, тобто $w(e(1)) = \min_{e \in E_1} w(e)$. Тоді v_1 називаємо першою найближчою вершиною (НВ), число $w(e(1))$ позначаємо $l(1) = l(v_1)$ і називаємо відстанню до цієї НВ. Позначимо $V_1=\{v_0, v_1\}$ – множину найближчих вершин.

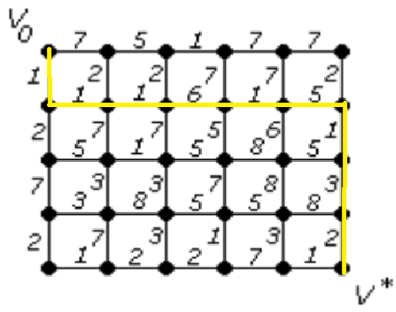
2-й крок індукції. Позначимо E_2 – множину усіх ребер $e=(v', v'')$, $e \in E$, таких що $v' \in V_1$, $v'' \in (V \setminus V_1)$. Найближчим вершинам $v \in V_1$ приписано відстані $l(v)$ до кореня v_0, причому $l(v_0)=0$. Введемо позначення: $\overline{V_1}$ – множина таких вершин $v'' \in (V \setminus V_1)$, що $\exists$ ребра виду $e=(v, v'')$, де $v \in V_1$. Для всіх ребер $e \in E_2$ знаходимо таке ребро $e_2=(v', v_2)$, що величина $l(v')+w(e_2)$ найменша. Тоді v_2 називається другою найближчою вершиною, а ребра e_1, e_2 утворюють зростаюче дерево для виділених найближчих вершин $D_2=\{e_1, e_2\}$.

(s+1)-й крок індукції. Нехай у результаті s кроків виділено множину найближчих вершин $Vs=\{v_0, v_1, \ldots, v_s\}$ і відповідне їй зростаюче дерево $D_s=\{e_1, e_2, \ldots, e_s\}$... Для кожної вершини $v \in V_s$ обчислена відстань $l(v)$ від кореня v_0 до v; $\overline{V_s}$ – множина вершин $v \in (V \setminus V_s)$, для яких існують ребра вигляду $e=(v_r, v)$, де $v_r \in V_s$, $v \in (V \setminus V_s)$. На кроці s+1 для кожної вершини $v_r \in V_s$ обчислюємо відстань до вершини v_r: $L(s+1)(v_r) = l(v_r) + \min\limits_{v^* \in V_s} w(v_r, v^*)$, де min береться по всіх ребрах $e=(v_r, v_*)$, $v^* \in \overline{V}_s$, після чого знаходимо min серед величин $L(s+1)(v_r)$. Нехай цей min досягнуто для вершин v_{r0} і відповідної їй $v^* \in \overline{V}_s$, що назвемо v_{s+1}. Тоді вершину v_{s+1} називаємо (s+1)-ю НВ, одержуємо множину $V_{s+1} = V_s \cup v_{s+1}$ і зростаюче дерево $D_{s+1} = D_s \cup (v_{r0}, v_{s+1})$. (s+1)-й крок завершується перевіркою: чи є чергова НВ v_{s+1} відзначеною вершиною, що повинна бути за умовою задачі зв'язано найкоротшим ланцюгом з вершиною v_0. Якщо так, то довжина шуканого ланцюга дорівнює $l(v_{s+1})=l(v_{r0})+w(v_{r0}, v_{s+1})$; при цьому шуканий ланцюг однозначно відновлюється з ребер зростаючого дерева D_{s+1}. У противному випадку випливає перехід до кроку s+2.

Варіант 2

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* .

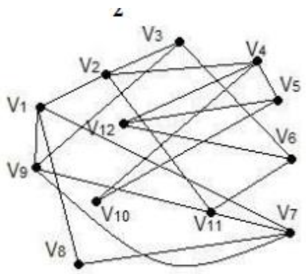


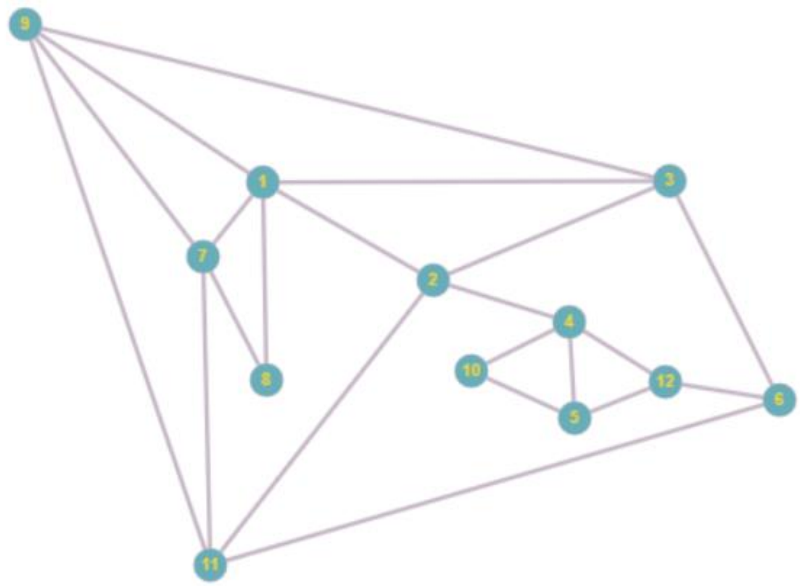


Мінімальна довжина :21

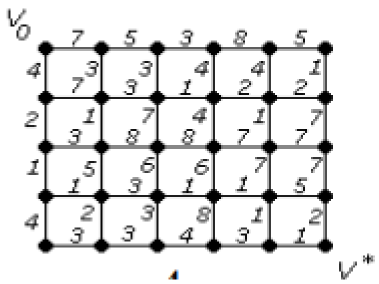
1-7-8-9-10-11-12-18-24-30

Завдання №2.За допомогою γ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.





Завдання №3 Написати програму, яка реалізує алгоритм Дейкстри знаходження найкоротшого шляху між парою вершин у графі. Протестувати розроблену програму на графі згідно свого варіанту.




```
#include <iostream>
using namespace std;
int n,g[50][50]={0},dist[50],pred[50];
bool visit[50];

void way(int j)
{
    if (pred[j] == -1)
        return;
    way(pred[j]);
    cout << "Top" << j+1 << " -> ";
}
int distance()
{
    int minimum = 10000, minD;
    for (int z = 0; z < n; z++)
        if (visit[z] == false && dist[z] <= minimum)
        {
            minimum = dist[z];
            minD = z;
        }
    return minD;
}
void algo(int g[50][50])
{
    for (int i = 0; i < n; i++)
    {
        pred[0] = -1;
        dist[i] = 10000;
        visit[i] = false;
    }
    dist[0] = 0;
    for (int j = 0; j < n - 1; j++)
    {
        int u = distance();
        visit[u] = true;
        for (int z = 0; z < n; z++)
            if (!visit[z] && g[u][z] && dist[u] + g[u][z] < dist[z])
```

```
                {
                    pred[z] = u;
                    dist[z] = dist[u] + g[u][z];
                }
        }
        cout << "The least way is: ";
        cout << dist[29] << endl;
        cout << "The way is: ";
        cout << "Top -> ";
        way(29);
        cout << "finish" << endl;
}
int main()
{
n=30;
int v1, v2;
cout<<"Number of rows and columns ";
cin>>v1>>v2;
for (int i=0; i<n; i++){
for(int j=i+1; j<n; j++){
if(j==i+1 || j==i+v1){
cout<<"top "<<i+1<<" to top "<<j+1<<" :";
cin>>g[i][j];
}
else
g[i][j]=0;
}
}
algo(g);
}
```

```
Number of rows and columns6 5
top 1 to top 2 :7
top 1 to top 7 :4
top 2 to top 3 :5
top 2 to top 8 :3
top 3 to top 4 :3
top 3 to top 9 :3
top 4 to top 5 :8
top 4 to top 10 :4
top 5 to top 6 :4
top 5 to top 11 :4
top 6 to top 7 :0
top 6 to top 12 :1
top 7 to top 8 :7
top 7 to top 13 :2
top 8 to top 9 :3
top 8 to top 14 :1
top 9 to top 10 :1
top 9 to top 15 :7
top 10 to top 11 :2
top 10 to top 16 :4
top 11 to top 12 :2
top 11 to top 17 :1
top 12 to top 13 :0
top 12 to top 18 :7
top 13 to top 14 :3
top 13 to top 19 :1
top 14 to top 15 :8
top 14 to top 20 :5
top 15 to top 16 :8
top 15 to top 21 :6
top 16 to top 17 :7
top 16 to top 22 :6
top 17 to top 18 :7
top 17 to top 23 :7
top 18 to top 19 :0
top 18 to top 24 :7
top 19 to top 20 :1
top 19 to top 25 :4
top 20 to top 21 :3
top 20 to top 26 :2
top 21 to top 22 :1
top 21 to top 27 :3
top 22 to top 23 :1
top 22 to top 28 :8
top 23 to top 24 :5
top 23 to top 29 :1
top 24 to top 25 :0
top 24 to top 30 :2
top 25 to top 26 :3
top 26 to top 27 :3
top 27 to top 28 :4
top 28 to top 29 :3
top 29 to top 30 :1
The way is: Top1 -> Top7 -> Top13 -> Top19 -> Top20 -> Top21 -> Top22 -> Top23 -> Top29 -> Top30 -> Return 0;
The least way is: 15
```

Висновок: Виконуючи цю лабораторну роботу, я закріпив знання алгоритму Дейкстри, а також гамма-вкладки.