МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Лабораторна робота №3

З курсу “Дискретна математика ”

Виконав:
ст.гр. КН-110
Синюк Павло

Викладач:
Мельникова Н. І.

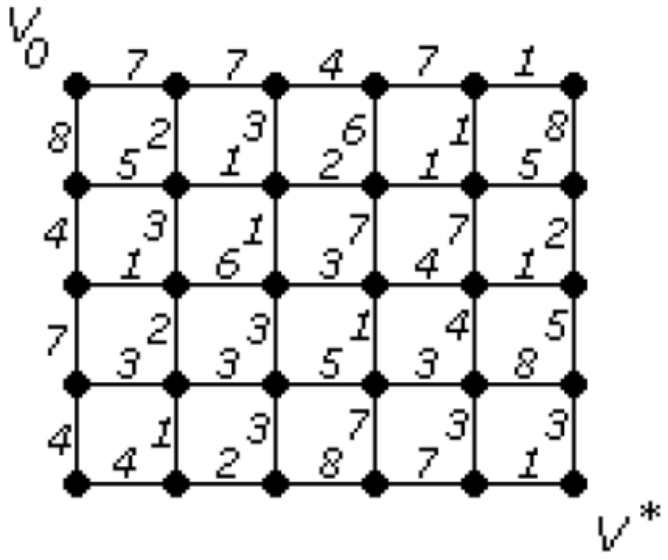
Львів – 2018

ВАРІАНТ №10

Завдання №1

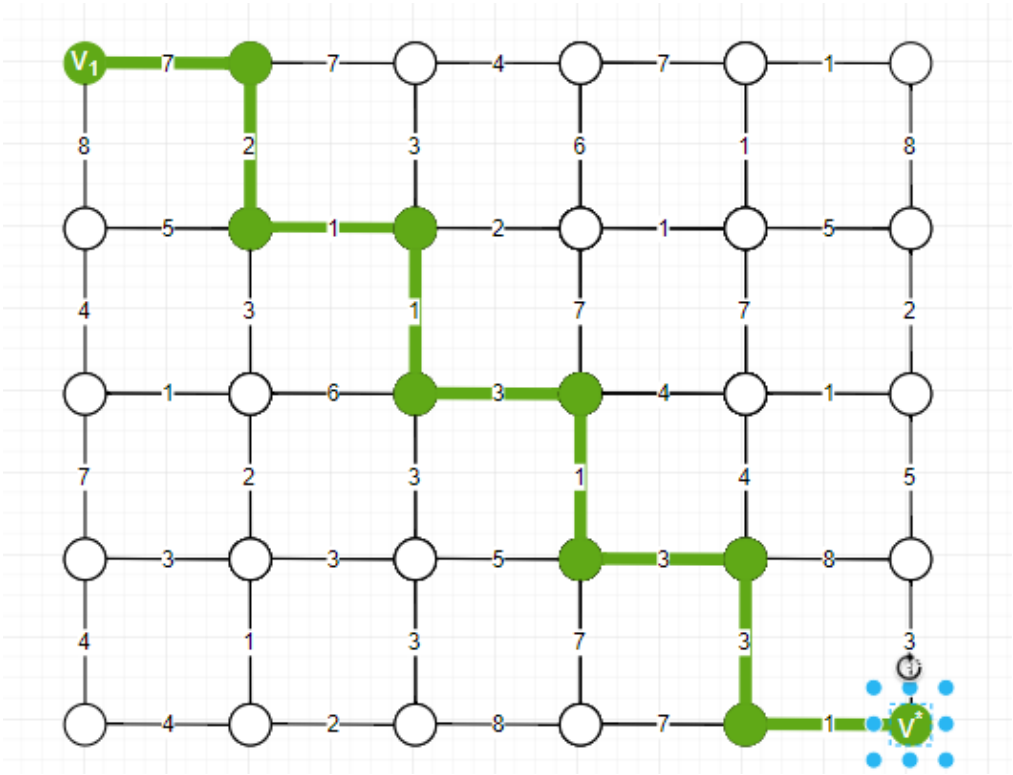
1. За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин V_0 і V^* .

Заданий граф



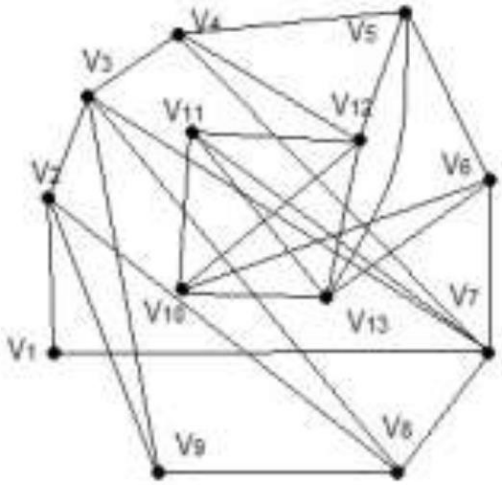
За алгоритмом Дейкстри ми обираємо по черзі усі точки що є найближчі до заданої.

Відповідь виглядатиме так:

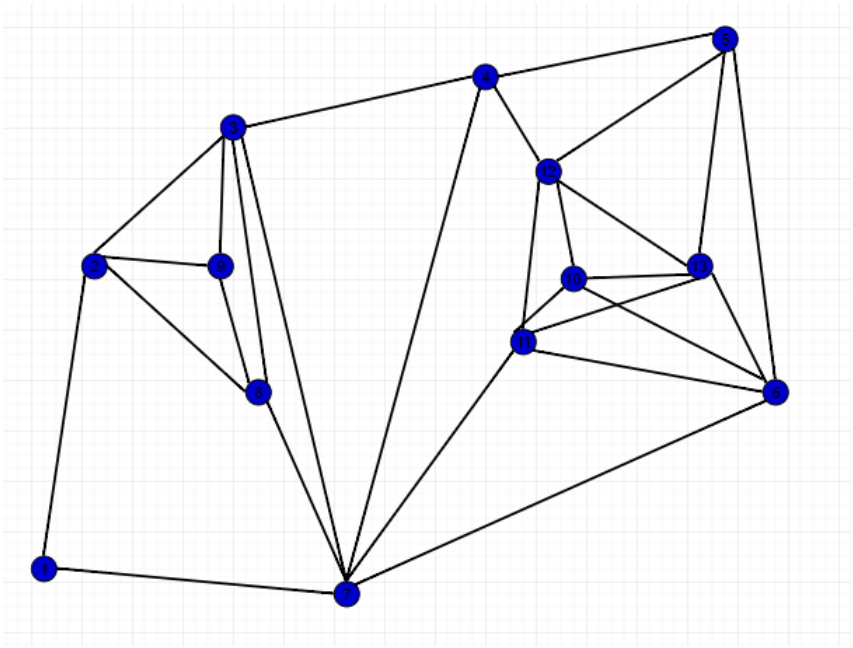


Відстань від вершини V_1 до вершини V^* дорівнює 22.

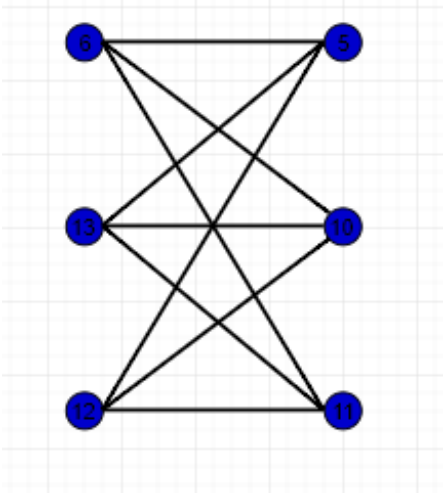
2. За допомогою -алгоритма зробити укладку графа у площині, або довести що вона неможлива.



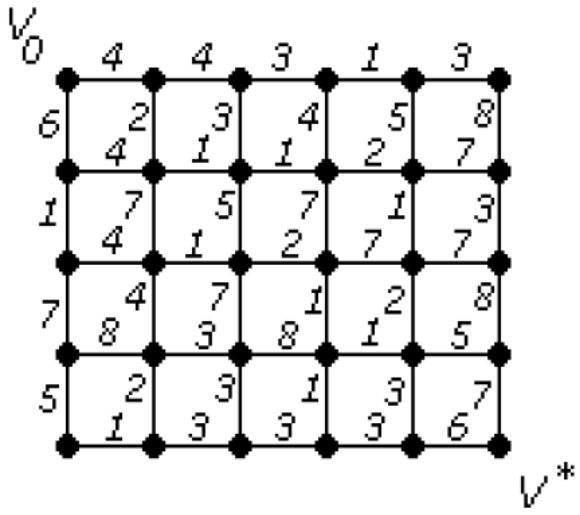
Після виконання перетворень ми спростили граф до вигляду:



Укладка графу не можлива адже його під графом є граф G1, який не можливо розкласти:



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



Код програми:

```
#include <stdio.h>

#include <cs50.h>


typedef struct EDGE{


    int first;

    int second;

    int weight;

    int check;


}EDGE;


typedef struct POINTS{
```

```
    int busho;
    int korito;
    int check;

}POINTS;

EDGE re[49];
POINTS ss[30];

void GetValue( EDGE re[49], POINTS ss[30]);
void FindCls( EDGE re[49], POINTS ss[30]);
void PrintWay( EDGE re[49], POINTS ss[30]);
int main(){
    GetValue(re, ss);
    for(int i=0; i<49; i++){
        printf("%2d-->%2d         %2d\n", re[i].first, re[i].second, re[i].weight);
    }
    FindCls(re, ss);
    PrintWay(re, ss);
}

void FindCls(EDGE re[49], POINTS ss[30]){

    int j=0, i=0, k=0;
```

```
    for(i=0; i<30; i++){
        if(ss[i].check==1){
            for(j=0; j<49; j++){
                if(ss[i].busho==re[j].first && re[j].check==0){
                    for(k=0; k<30; k++){
                        if(re[j].second==ss[k].busho &&
ss[k].korito>ss[i].korito+re[j].weight){
                            ss[k].korito=ss[i].korito+re[j].weight;
                            re[j].check=1;
                            ss[k].check=1;
                        }
                    }
                }
                /* else if(ss[i].busho==re[j].second && re[j].check==0){
                    for(k=0; k<30; k++){
                        if(re[j].first==ss[k].busho &&
ss[k].korito>ss[i].korito+re[j].weight){
                            ss[k].korito=ss[i].korito+re[j].weight;
                            re[j].check=1;
                            ss[k].check=1;
                        }
                    }
                }break;*/
            }
        }
```

```
    }
   // for(i=0; i<30; i++)
     //   printf("%2d     %2d\n", ss[i].busho, ss[i].korito);
}


void PrintWay( EDGE re[49], POINTS ss[30]){
    int i=0, j=0;

    for(j=0; j<49; j++)
        re[j].check=0;

    for(i=0; i<30; i++){
        ss[i].check=0;
    }
    ss[29].check=1;


    for(int i=29; i>=0; i--){
        if(ss[i].check==1){
            for(int j=49;j>=0; j--){
                if(ss[i].busho==re[j].second && re[j].check==0){
                    for(int k=29; k>=0; k--){
                        if((ss[i].korito-re[j].weight==ss[k].korito) &&
(ss[k].busho==re[j].first)) {
```

```
                    ss[k].check=1;
                    re[j].check=1;
                    break;
                    }
                }
            }
        }
    }


    for(i=0; i<30; i++){
        for(j=0;j<49;j++){
            if(ss[i].check==1 && re[j].first==ss[i].busho && re[j].check==1){
                printf("%2d-->%2d      %2d\n", re[j].first, re[j].second, ss[i+1].korito);
            }
        }
    }
}


void GetValue(EDGE re[49], POINTS ss[30]){

    //horisontal
    int i=0, j=0;
```

```
re[i].first=j+1;re[i].second=j+2;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=2;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=7;i++;j++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=2;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=7;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=7;i++;j++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=8;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=8;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=5;i++;j++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+2;re[i].weight=6;i++;j++;j++;
```

```
j=0;
re[i].first=j+1;re[i].second=j+7;re[i].weight=6;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=2;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=5;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=8;i++;j++;//
re[i].first=j+1;re[i].second=j+7;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=7;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=5;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=7;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=3;i++;j++;//
re[i].first=j+1;re[i].second=j+7;re[i].weight=7;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=4;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=7;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=2;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=8;i++;j++;//
re[i].first=j+1;re[i].second=j+7;re[i].weight=5;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=2;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=3;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=1;i++;j++;
re[i].first=j+1;re[i].second=j+7;re[i].weight=3;i++;j++;
```

```
    re[i].first=j+1;re[i].second=j+7;re[i].weight=7;i++;j++;//



    for(i=0; i<30; i++){
        ss[i].busho=i+1;
        ss[i].korito=100000;
        ss[i].check=0;
    }


    for(j=0; j<49; j++)
        re[j].check=0;
        ss[0].korito=0;
    ss[0].check=1;
}
```

Вивід програми:

```
 1--> 2       4
 2--> 8       8
 8--> 9       7
 9-->10       8
10-->11      10
11-->17      17
17-->23      18
23-->29      18
29-->30      22
jharvard@appliance (~/Projects/Discret):
```

Висновок:

Я ознайомився з методом Дейкстри з обходом в шир і в довжину графа. Програмно виконав завдання №2, де за методом Дейкстри я знайшов найкоротший шлях між двома вершинами графу.