图论

question1

设备更新：企业使用一台设备，每年年初，企业就要确定是购置新的， 还是继续使用旧的。若购置新设备，就要支付一定的购置费用；若继续使用，则需支付一定的维修费用。现要制定一个五年之内的设备更新计划， 使得五年内总的支付费用最少。 已知该种设备在每年年初的价格为：

第一年	第二年	第三年	第四年	第五年
11	11	12	12	13

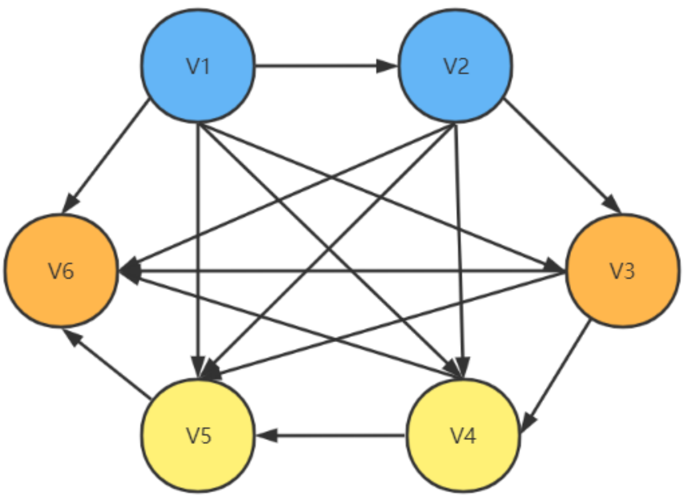
使用不同时间设备所需维修费为：

使用年限	0-1	1-2	2-3	3-4	4-5
维修费	5	6	8	11	18

要求：用最短路算法解决该问题，并编程求解。

解：

建立模型，将该问题转换为图论的最短路径问题，图如下：



- 符号说明

上图中，有 $G=\{V_1,V_2,V_3,V_4,V_5,V_6\}$ ，V_i 表示第 i 年购买新设备的决策。特别的，V_6 表示第5年末购买新设备的决策。

边 $V_i\rightarrow V_k$ 表示第 i 年购买的设备一直用到第 k 年的决策。比如 $\{V_2,V_5\}$ 表示 $V_2\rightarrow V_5$ 选择开始时购买设备，然后之后一直维修用到第五年。总花费为：11+5+6+8 = 30； $V_1\rightarrow V_2$ 花费则为：11+5 = 16；其他的以此类推。可以得到下面的花费矩阵：

	V1	V2	V3	V4	V5	V6
V1	0	16	22	30	41	59
V2	inf	0	16	22	30	41
V3	inf	inf	0	17	23	31
V4	inf	inf	inf	0	17	23
V5	inf	inf	inf	inf	0	18
V6	inf	inf	inf	inf	inf	0

对上面的矩阵使用迪杰斯特拉算法得到最短路径为：$\{V_1 \rightarrow V3 \rightarrow V6\}$，即设备更新计划为：第一年购买新设备，第三年购买新设备，第五年末购买新设备。编程程序如下：

```
#include<iostream>
#include<stdio.h>
#include<math.h>
#include<string.h>

using namespace std;

void initial();
void dijkstra(int start);

const int maxInt = 1000000;
int start;
int m,n;            //m个顶点，n条边
int dis[100];       //起始点到点i之间的最短距离，会不断被更新
int book[100];      //标志着i有无被访问
int map[100][100];
int path[100];

int main(){
    initial();
    dijkstra(start);
    return 0;
 }

void initial(){
    int x,y,w;
    memset(dis, 88, sizeof dis);
    memset(map, 88, sizeof map);

    cout<<"请输入顶数和边数"<<endl;
    cin>>m>>n;      //m个顶点,n条边
    cout<<"请输入起始点"<<endl;
    cin>>start;     //求start到各个点的距离
    cout<<"请输入顶点间边的权重，如顶点1和2之间的边权为3，则输入1 2 3"<<endl;
    for(int i = 0; i<n; i++){
        cin>>x>>y>>w;
        map[x][y] = w;
        map[y][x] = w;
        map[i][i] = 0;
        path[i] = -1;
```

```
    }
}

void dijkstra(int start){
    for(int i = 0; i<m; i++){
        dis[i] = dis[i] < map[start][i]? dis[i]:map[start][i];  //先进行一轮初始化
    }

    book[start] = 1;

    for(int i = 0; i<m; i++){
        int min = maxInt;
        int next = 0;
        for(int j = 0; j<m; j++){   //记录当前时刻距离源点最近距离，且未被探索的点，将其作为扩展点
            if(!book[j]&&min>dis[j]){
                min = dis[j];
                next = j;
            }
        }
        book[next] = 1;
        for(int j = 0; j<m; j++){
            if (!book[j] && dis[next] + map[next][j] < dis[j]){
                dis[j] = dis[next] + map[next][j];
                path[j] = next;
            }
        }
    }

    for(int i = start; i<m; i++){
        cout<<"点"<<start<<"到第"<<i<<"个节点的最短距离为："<<dis[i]<<endl;
    }

    cout<<"从终点到起点（逆向）的最短路径为：";
    int p = 6;
    cout<<p;
    while(path[p]!=-1){
        cout<<"-->"<<path[p];
        p = path[p];
    }
    cout<<"-->"<<start<<endl;
 }

```

结果如下:

```
D:\桌面内容\数模_迪杰斯特拉最短路径.exe
请输入顶数和边数
7 15
请输入起始点
1
请输入顶点间边的权重，如顶点1和2之间的边权为3，则输入1 2 3
1 2 16
1 3 22
1 4 30
1 5 41
1 6 59
2 3 16
2 4 22
2 5 30
2 6 41
3 4 17
3 5 23
3 6 31
4 5 17
4 6 23
5 6 18
点1到第1个节点的最短距离为: 0
点1到第2个节点的最短距离为: 16
点1到第3个节点的最短距离为: 22
点1到第4个节点的最短距离为: 30
点1到第5个节点的最短距离为: 41
点1到第6个节点的最短距离为: 53
从终点到起点（逆向）的最短路径为: 6-->3-->1

--------------------------------
Process exited after 4.35 seconds with return value 0
```

question2

某人带着狼、羊以及蔬菜渡河，一小船除需人划外，每次只能载一物过河。而人不在场时，狼要吃羊，羊要吃菜，问此人应该如何渡河。请用最短路求解。

解：

用四维向量来表示状态：

- 第一分量：人；
- 第二分量：狼；
- 第三分量：羊；
- 第四分量：菜；

用0和1表示向量中每个分量的状态：当某物在此岸时记作1，在对岸时记作0.

如 $(0,1,1,1)$表示人不在此岸，狼羊菜都在对岸。

首先，根据题目的限制，我们列举出所有的允许的状态，共有10种，列举如下：

$$\begin{aligned}&(1,1,1,1)\\&(1,0,1,1)\\&(1,1,0,1)\\&(1,1,1,0)\\&(1,0,1,0)\\&(0,1,0,1)\\&(0,1,0,0)\\&(0,0,1,0)\\&(0,0,0,1)\\&(0,0,0,0)\end{aligned}$$

同时，状态之间的转移通过定义转移向量来实现，转移向量有四种：

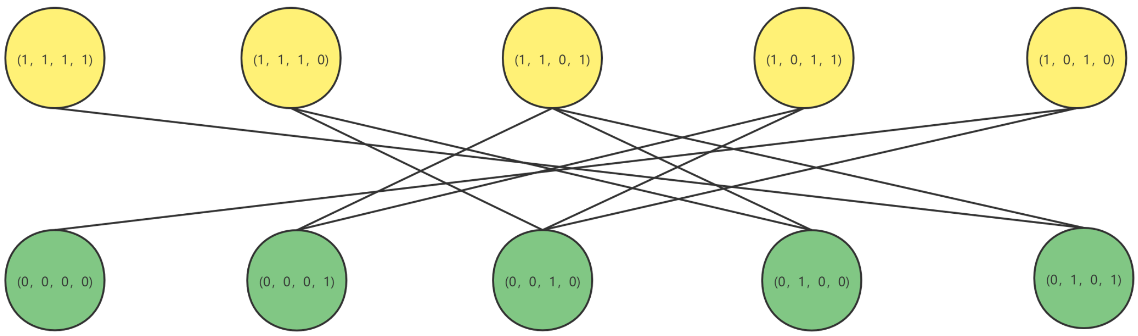
$$\begin{cases}(1,0,0,0):\text{人单独过}\\(1,0,0,1):\text{人带菜过}\\(1,0,1,0):\text{人带羊过}\\(1,1,0,0):\text{人带狼过}\end{cases}$$

状态之间通过与转移向量相加，可以得到下一状态，只要状态是允许的状态，那么就满足转移的题意。

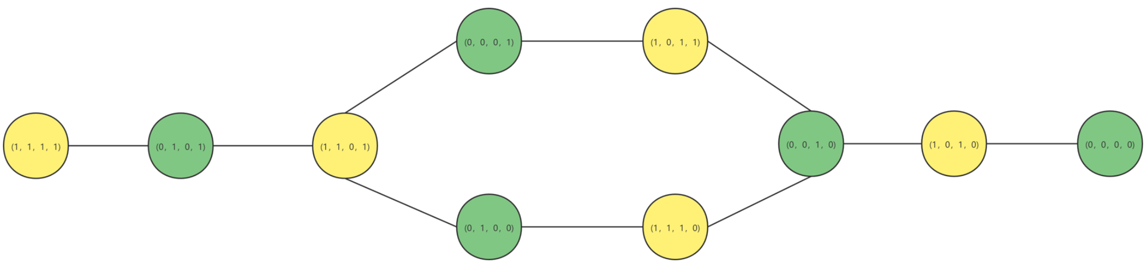
相加的加法运算规则如下：

$$0+0=0 \quad 1+0=1 \quad 0+1=1 \quad 1+1=0$$

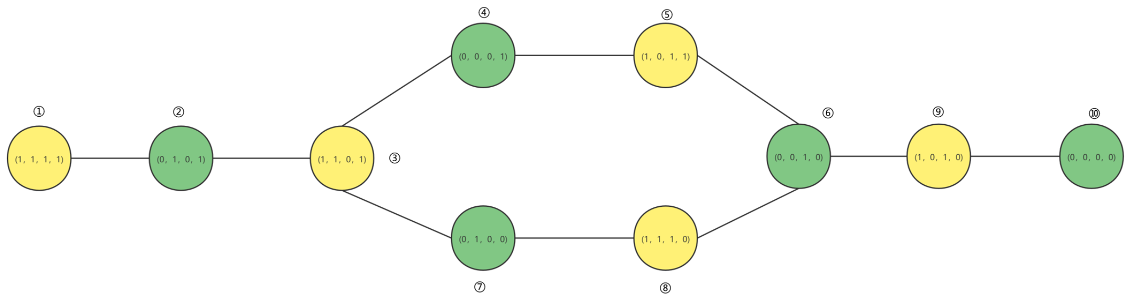
这样，题目的要求就转变为：从 $(1,1,1,1)$ 通过有限次的与转移向量相加转化到状态 $(0,0,0,0)$。在上面的相加定义下，假设从 $A\to B$ 可由加上转移向量①实现，那么从 $B\to A$ 也可由加上向量①实现，即状态的转移具有对称性。状态的转移可以通过下面的状态转移图表示：



由于这种对称性的存在，建立的图模型可以用无向图表示，整理上面的图为平面图，如下：



每条边赋权值为1，可以得到两种方案。



方案①：$1->2->3->4->5->6->9->10$

方案②：$1->2->3->7->8->6->9->10$