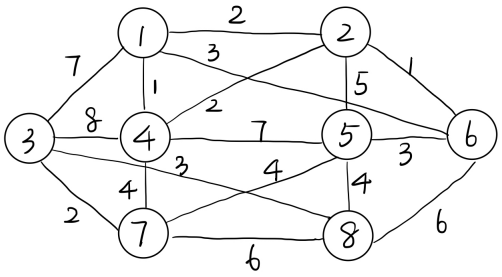
实　验　报　告

评分:

数学　系　2019　级　姓名　赵宁双　日期　20210408　No.　PB19010444

1　最小支撑树的实现

【实验题目】给出求最小支撑树的程序实现，并以书上图 11-39（a）为例验证。



	1	2	3	4	5	6	7	8
1	∞	2	7	1	∞	3	∞	∞
2	2	∞	∞	2	5	1	∞	∞
3	7	∞	∞	8	∞	∞	2	3
4	1	2	8	∞	7	∞	4	∞
5	∞	5	∞	7	∞	3	4	4
6	3	1	∞	∞	3	∞	∞	6
7	∞	∞	2	4	4	∞	∞	6
8	∞	∞	3	∞	4	6	6	∞

【实验代码】 函数部分

```
function Prim(A)%输入要求的矩阵A
[m,m]=size(A); %m为A的行数和列数，也就是生成树的点的数量
B=A;%修改B不修改A，从B中取出代价最小的边
C=ones(m,m);
INF=99;%用99表示正无穷大
C=INF*C;%将代价最小的边放入C中
Min=INF;Min_i=1;Min_j=1;
D=zeros(1,m);%找到代价最小的边后，对应的点做标记
D(1,1)=1;
for k=1:m-1%共有m-1条边
    Min=INF;%每一轮寻找都作初始化
for i=1:m
    if (D(1,i)==1)
        for j=1:m
            if(B(i,j)<Min)
                Min=B(i,j);Min_i=i;Min_j=j;%找出代价最小的边
            end
        end
    end
end
for t=1:m
    if(D(1,t)==1)
        B(t,Min_j)=INF;B(Min_j,t)=INF;%取出该点，将已经生成树之间的节点之间的
        权设为无穷大
    end
end
C(Min_i,Min_j)=Min;C(Min_j,Min_i)=Min;%将代价最小边放入C中
D(1,Min_j)=1;%将对应的点做标记
```

```
29  end
30  disp(C)
31  end
```

输入部分

```
1  >>A=[99,2,7,1,99,3,99,99;2,99,99,2,5,1,99,99;7,99,99,8,99,99,2,3;1,2,8,99,7,99,4,99];
2  >>Prim(A)
```

【运行结果】

```
1     99     2    99     1    99    99    99    99
2      2    99    99    99    99     1    99    99
3     99    99    99    99    99    99     2     3
4      1    99    99    99    99    99     4    99
5     99    99    99    99    99     3    99    99
6     99     1    99    99     3    99    99    99
7     99    99     2     4    99    99    99    99
8     99    99     3    99    99    99    99    99
```

【实验总结】

本实验利用了 Prim 算法，用矩阵 C 作为存放代价最小的边和对应点，同时在取出对应的边后将原矩阵 B 中已被取出点的路径改为无穷大。具体的思想已经在注释中体现。经过验证，输出的矩阵确实为最小生成树。

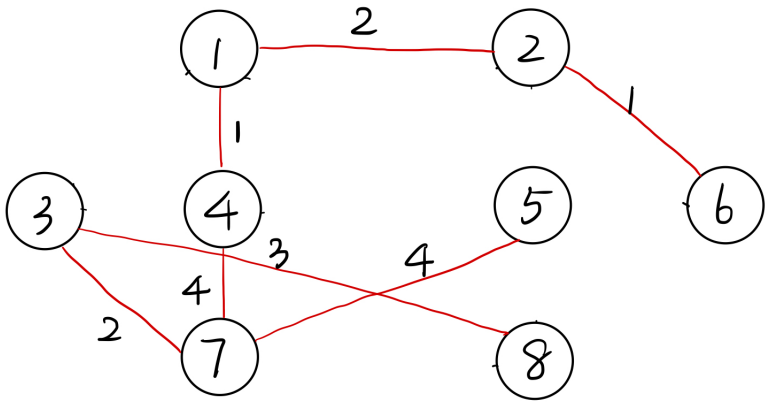


Figure 1: 实验结果

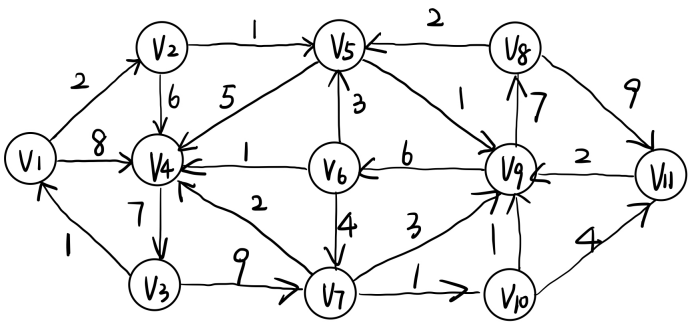
实　验　报　告

评分:

数学　系　2019　级　姓名　赵宁双　日期　20210408　No.　PB19010444

2　最短路问题

【实验题目】给出最短路算法的程序实现（不局限于书上介绍的 Dijkstra 方法），并求出书上图 11-41 中从 v1 到 v11 的最短路。



	1	2	3	4	5	6	7	8	9	10	11
1		2		8							
2				6	1						
3	1						9				
4			7								
5				5					1		
6				1	3		4				
7				2					3	1	
8					2						9
9						6		7			
10									1		4
11									2		

【实验代码】

函数部分

```
function DIJ(A)
[m,m]=size(A);
INF=999;%设999为无穷大
P=zeros(m,m);%P（v，w）若为1，则w是v1到v当前求得最短路径上的顶点之一。
%设空路径
F=zeros(1,m); %F矩阵表示v1->v的最短路径是否已经找到
D=zeros(1,m); %D矩阵表示v1->v的路径长短
%下面进行初始化操作
for v=1:m
    F(1,v)=0;%v1到所有的v的最短路径都还没有找到
    D(1,v)=A(1,v);%v1到所有v的最短路径都保持初始状态
    if(D(1,v)<INF)
        P(v,1) = 1;%只要v和v1之间有直接路径，则v1必定在路径上
        P(v,v) = 1;%只要v和v1之间有直接路径，则v必定在路径上
    end
end
%下面开始主循环，每次求得v1到某个定点v的最短路径，并将v加到S集合
for i=1:m
    min=INF;%先设最小值为无穷大
    for w=1:m
        if(F(1,w)==0)%如果v1到w的路径还没有被找到
            if(D(1,w)<min)
                v=w;
                min=D(1,w);
            end
        end
    end
```

```
28        F(1,v)= 1;%1到v的最短路径暂时被找到了
29        for w=1:m
30            if(F(1,w)==0 && (min+A(v,w)<D(1,w)))
31                D(1,w)=min+A(v,w);%比较1->w的路径和1->v->w的路径长短，更新长度数据
32                for j=1:m
33                    P(w,j) = P(v,j);
34                    P(w,w)=1;%更新节点数据
35                end
36            end
37        end
38    end
39    disp(P)
40    end
```

输入部分

```
1  >> A=ones(11,11);
2  >> A=999*A;
3  >> A(1,2)=2;A(1,4)=8;
4  >> A(2,4)=6;A(2,5)=1;
5  >> A(3,1)=1;A(3,7)=9;
6  >> A(4,3)=7;
7  >> A(5,4)=5;A(5,9)=1;
8  >> A(6,4)=1;A(6,5)=3;A(6,7)=4;
9  >> A(7,4)=2;A(7,9)=3;A(7,10)=1;
10 >> A(8,5)=2;A(8,11)=9;
11 >> A(9,6)=6;A(9,8)=7;
12 >> A(10,9)=1;A(10,11)=4;A(11,9)=2;
13 >> DIJ(A)
```

【运行结果】

```
1     1     0     1     1     0     0     0     0     0     0     0
2     1     1     0     0     0     0     0     0     0     0     0
3     1     0     1     1     0     0     0     0     0     0     0
4     1     0     0     1     0     0     0     0     0     0     0
5     1     1     0     0     1     0     0     0     0     0     0
6     1     1     0     0     1     1     0     0     1     0     0
7     1     1     0     0     1     1     1     0     1     0     0
8     1     1     0     0     1     0     0     1     1     0     0
9     1     1     0     0     1     0     0     0     1     0     0
10    1     1     0     0     1     1     1     0     1     1     0
11    1     1     0     0     1     1     1     0     1     1     1
```

【实验总结】

由运行结果可知，v_1 到 v_11 的最短路径是 $v_1->v_2->v_5->v_9->v_6->v_7->v_{10}->v_{11}$。本实验的具体思想已经体现在注释中。其中关键的矩阵为 P、F、D，P（v，w）用于标记 v1 到 v 当前求得的最短路径上包含了点 w，F（1，v）用于标记 v1 到 v 的最短路径是否已经找到，而 D（1，v）用于标记 v1 到 v 的最短路径。接下来通过循环、比较依次求出到各条路的最短路径。

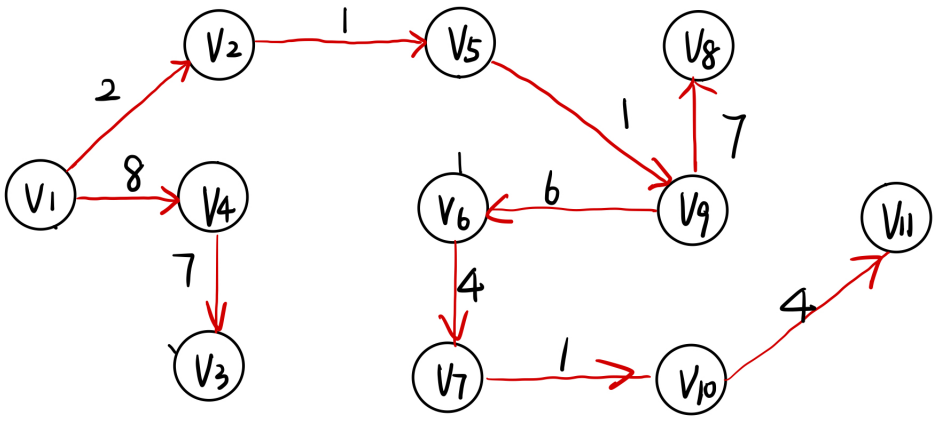


Figure 2: 实验结果