

输入数据都是30m×30m的栅格数据

1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
2	1	2	3	3	2	5	4	0	1	2	4	4	4	2
1	2	2	3	2	0	1	0	2	3	2	4	5	2	1
2	5	4	4	2	4	1	4	2	4	0	2	3	1	3
4	4	3	4	2	1	3	2	5	4	3	4	2	4	1
0	0	1	2	3	1	1	3	4	1	2	1	5	1	2
3	4	2	5	1	5	0	4	4	3	4	ന	2	1	1
1	2	1	4	2	3	2	4	3	5	0	3	2	3	1
3	4	5	3	2	5	3	2	4	3	4	3	3	2	5
3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
3	5	3	4	3	3	2	4	3	2	5	3	2	4	0
3	3	2	1	1	5	4	5	3	3	2	4	0	3	2

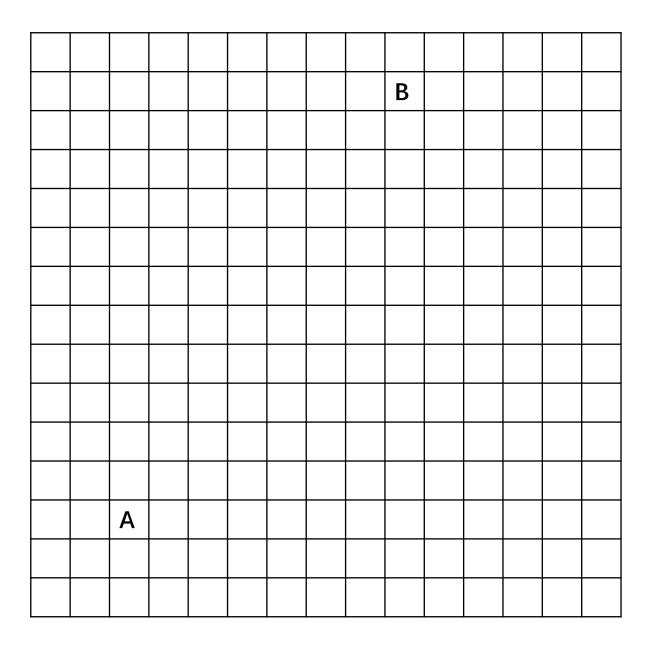
赋值栅格1:

MESV指数 (0-5)

1	2	5	4	80	9	1	3	2	4	10	9	8	5	6
2	4	8	9	1	<b>15</b>	4	3	7	3	5	9	8	5	5
5	4	7	10	2	1	5	6	7	8	8	7	5	8	9
10	8	5	4	6	5	7	1	4	10	2	5	7	6	6
5	8	9	9	7	6	4	2	5	10	3	7	80	9	5
1	1	4	5	2	ന	6	7	8	7	9	10	5	9	7
5	8	7	9	10	<b>1</b> 5	80	9	10	7	80	5	6	5	8
9	7	4	8	15	2	1	3	6	6	5	7	8	9	10
10	2	2	4	<b>L</b> 5	7	80	80	9	5	15	4	2	2	4
1	8	8	7	15	4	2	1	1	2	3	6	9	9	8
2	10	4	4	2	2	1	3	6	5	7	8	9	10	8
2	4	3	5	6	4	2	1	3	7	4	8	5	2	1
1	2	2	3	4	5	4	4	1	5	7	8	9	10	8
2	5	4	1	7	8	7	8	5	2	4	5	4	7	5
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4

赋值栅格2:

ECR值(1-10)



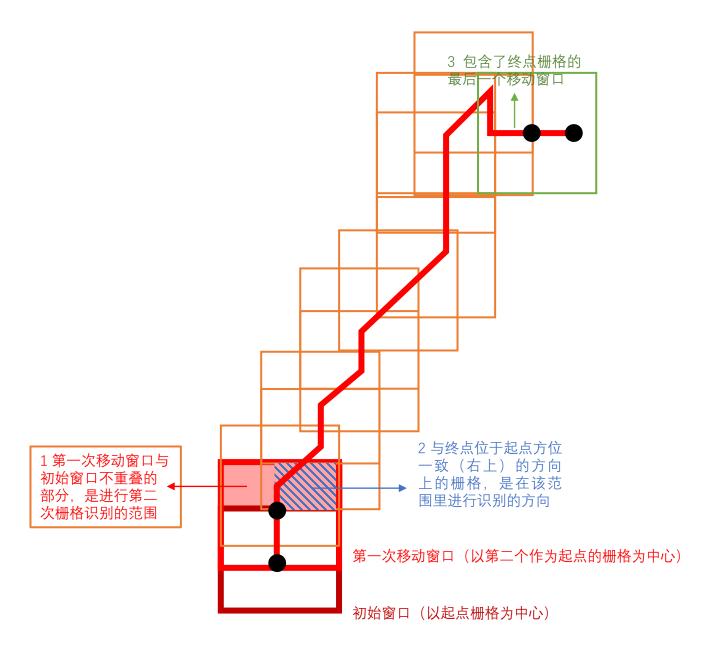
从起点A到终点B的路径搜寻 \_*(左下─右上)*\_

		P1)	芸													
P1+1	1层	1	2	5	3	2	1	4	Z	0	3	2	1	4	5	3
1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	4	2	3	4	2	3	1		0	1	4	2	1	4	5	3
5	4	2	1	2	3	3	2	į	4	0	1	2	4	4	4	2
10	8	1	2	2	3	2	0		0	2	3	2	4	5	2	1
5	8	2	5	4	4	2	J.	1	4	2	4	0	2	3	1	3
1	1	4	4	3	4	7	1	3	2	5	4	3	4	2	4	1
5	8	0	0	1	2		1	1	3	4	1	2	1	5	1	2
9	7	3	4	2	7	1	5	0	4	4	3	4	3	2	1	1
10	2	1	2	1		2	3	2	4	3	5	0	3	2	3	1
1	8	3	4	F	3	2	5	3	2	4	3	4	3	3	2	5
2	10	3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
2	4	1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
1	2	3	5	3	4	3	3	2	4	3	2	5	3	2	4	0
2	5	3	3	2	1	1	5	4	5	3	3	2	4	0	3	2
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		
																レフトコ

方法规定: 1) 只有在高层目标已满足但未得到唯一结果的情况下,才考虑继续满足底层目标,直到得到唯一结果; 2) 在考虑底层目标时,不允许违背已满足的高层目标后已得到唯一结果,不允许违背已满足高层目标后已得到唯一的法果,便不再继续考虑底层目标被告现实。这种不同层次目标被告,简化计算流程、时间。这种不同层次目标的优先因子P1来表示,以为P1》P1+1,即P1层目标的满足优先于P1+1层目标。

#### 具体执行规则:

- P1层目标为选择**MESV指数**最大的栅格;
- P1+1层目标为选择**ECR值**最小的栅格;
- ▶ P1+2层目标为选择与目的地栅格**欧氏距 离**最近的栅格。



#### 执行三个规则的前提基础:

#### 1) 每一次进行贪婪选择的栅格范围:

以每个起始点栅格为中心,用一个3×3的移动窗口对周边栅格进行覆盖;在新的窗口与上一个窗口(若存在)非重叠的部分(左图浅红色区域),执行多层次目标规则下的贪婪选择,从而防止识别的路径走回头路。(即:前一个窗口中包含过的栅格不参与下一次规则中的比较和识别,从而保证每一次移动后执行规则的栅格都是新的。)

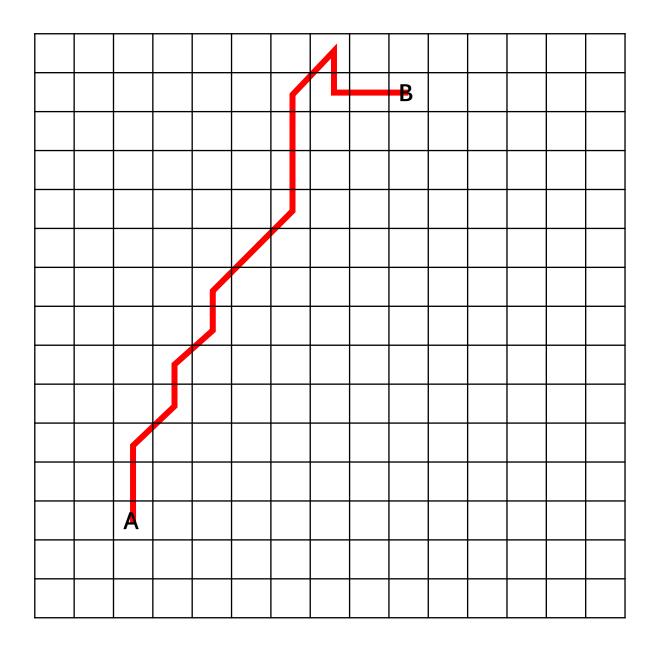
#### 2) 每一次进行贪婪选择的前进方向:

选择位于与终点相对于每一个新的起点方位一致 (左上/左下/右上/右下/左侧/右侧/上侧/下侧) 的 方向上的栅格进行贪婪选择。

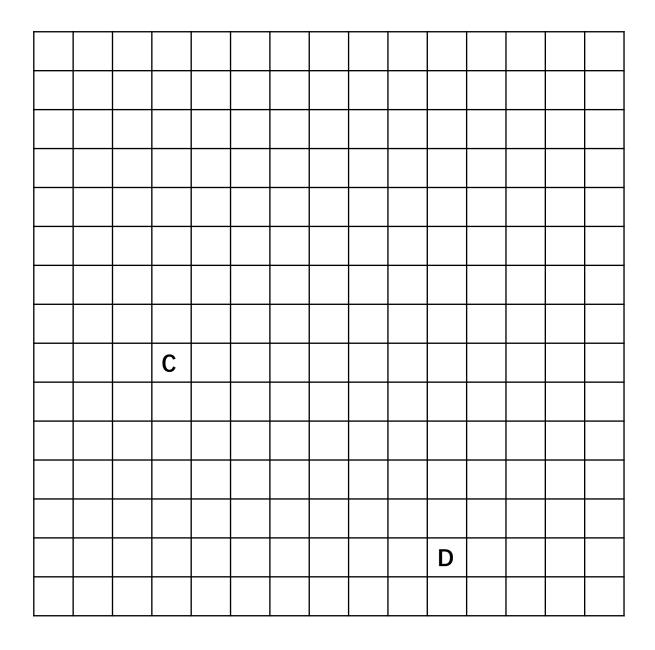
#### 3) 进行贪婪选择的结束标志:

当3×3的移动窗口中包含了属于终点的栅格时,将该终点栅格设置为下一步也就是最后一步前进的栅格,并终止搜索。

疑问: 以上规则如何更好地在编程/规则中表述?



从起点A到终点B的路径搜寻结果 \_*(左下─右上)*\_

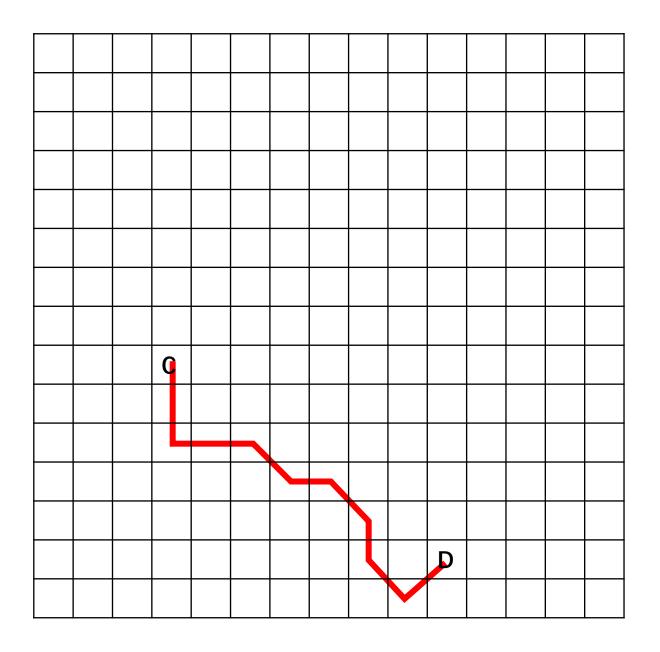


从起点C到终点D的路径搜寻 \_*(左上→右下)*\_ P1层

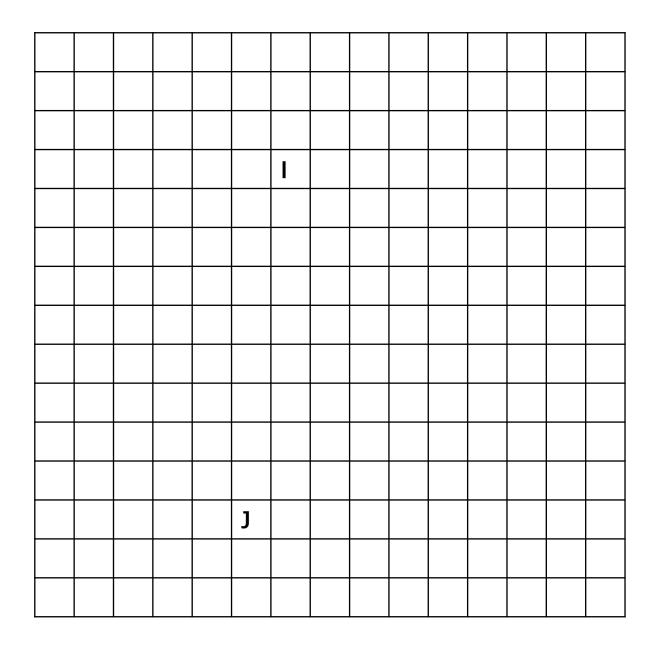
P1+	1层	1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	4	2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
5	4	2	1	2	3	3	2	5	4	0	1	2	4	4	4	2
10	8	1	2	2	3	2	0	1	0	2	3	2	4	5	2	1
5	8	2	5	4	4	2	4	1	4	2	4	0	2	3	1	3
1	1	4	4	3	4	2	1	3	2	5	4	3	4	2	4	1
5	8	0	0	1	2	3	1	1	3	4	1	2	1	5	1	2
9	7	3	4	2	5	1	5	0	4	4	3	4	3	2	1	1
10	2	1	2	1	4	2	3	2	4	3	5	0	3	2	3	1
1	8	3	4	5	1	2	5	3	2	4	3	4	3	3	2	5
2	10	3	3	5	2	2	3	4	-5	3	3	3	2	2	1	1
2	4	1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
1	2	3	5	3	4	3	3	2	4	1	2	5	3	2	4	0
2	5	3	3	2	1	1	5	4	5	3	8	2	4	0	3	2
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		

案例2

从起点C到终点D的路径搜寻结果 \_*(左上→右下)*\_



从起点C到终点D的路径搜寻结果 \_*(左上→右下)*\_

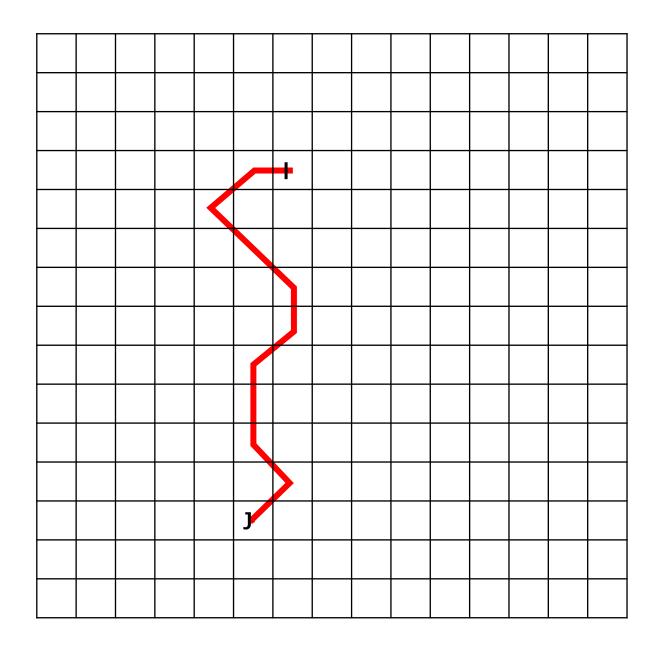


从起点I到终点J的路径搜寻 *(右上→左下,几乎在一列)*  P1层

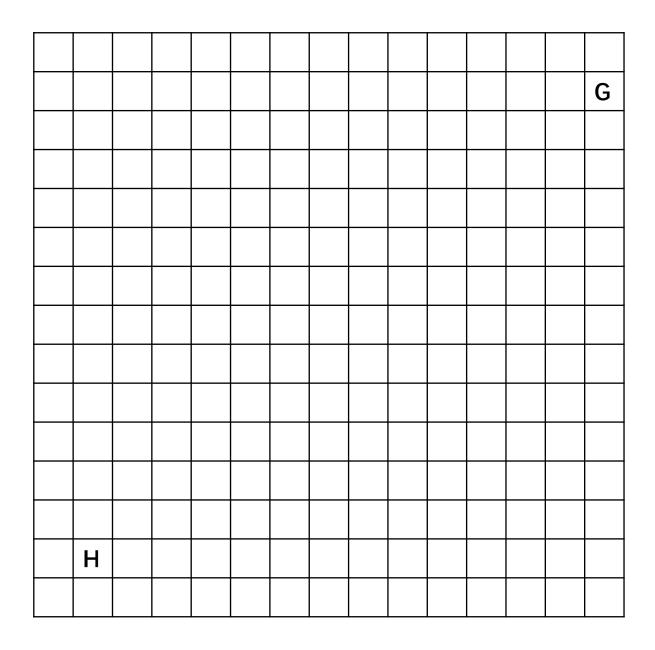
P1+	1层	1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	4	2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
5	4	2	1	2	3	3	2	-5	4	0	1	2	4	4	4	2
10	8	1	2	2	3		0	1	0	2	3	2	4	5	2	1
5	8	2	5	4	4	2	M	1	4	2	4	0	2	3	1	3
1	1	4	4	3	4	2	1	3	2	5	4	3	4	2	4	1
5	8	0	0	1	2	3	1		3	4	1	2	1	5	1	2
9	7	3	4	2	5	1		0	4	4	3	4	3	2	1	1
10	2	1	2	1	4	2		2	4	3	5	0	3	2	3	1
1	8	3	4	5	3	2	T	3	2	4	3	4	3	3	2	5
2	10	3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
2	4	1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
1	2	3	5	3	4	3	3	2	4	3	2	5	3	2	4	0
2	5	3	3	2	1	1	5	4	5	3	3	2	4	0	3	2
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		

# 案例3

从起点I到终点J的路径搜寻结果 \_*(右上→左下,几乎在一列)* 



从起点I到终点J的路径搜寻结果 *(右上一左下,几乎在一列)* 

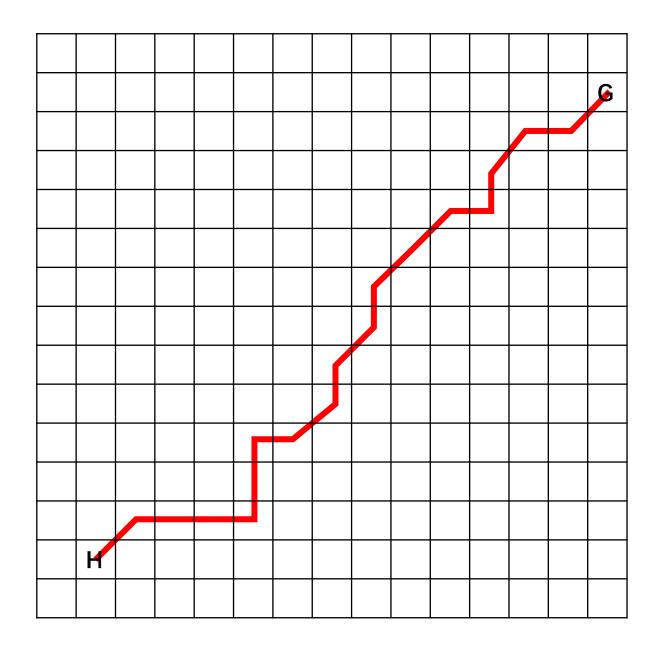


从起点G到终点H的路径搜寻 *(右上→左下,长距离)*  P1层

1   2   3   4   5   3   2   2   3   1   1   5   0   0   4   3   5     2   4   2   3   4   2   3   1   0   0   1   4   2   1   6   3     5   4   2   1   2   3   2   5   4   0   1   2   4   4   2     10   8   1   2   2   3   2   0   1   0   2   3   2   1   1   4   4   4   4   1   4   2   3   1   1   4   4   4   4   1   4   2   4   1   4   2   4   1   4   2   4   1   4   2   4   1   4   2   4   1   4   2   4   1   4   2   4   1   4   2   1   4   4   3   4   3   4   3<			1 1/														
2   4   2   3   4   2   3   1   0   0   1   4   2   1   4   2     5   4   2   1   2   3   3   2   5   4   0   1   2   4   4   2     10   8   1   2   2   3   2   0   1   0   2   3   2   4   1     5   8   2   5   4   4   2   4   1   4   2   4   1     5   8   0   0   1   2   3   1   1   3   2   4   3   4   2   4   1     5   8   0   0   1   2   3   1   1   3   4   3   4   2   4   1     9   7   3   4   2   5   1   5   0   4   4   3   4   3   2   1   1     10	P1+		1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
5   4   2   1   2   3   3   2   5   4   0   1   2   1   4   4   2     10   8   1   2   2   3   2   0   1   0   2   3   2   1   1     5   8   2   5   4   4   2   4   1   4   2   4   3   4   2   4   1   4   2   4   3   4   2   4   1   3   2   6   4   3   4   2   4   1   3   2   6   4   3   4   2   4   1   4   2   4   3   4   2   4   1   3   4   3   4   2   4   1   3   4   3   4   2   4   1   4   3   4   3   2   1   1   1   3   4   3   4   3   2   1   1   1   3   4   3 <td>1</td> <td>2</td> <td>3</td> <td>4</td> <td>5</td> <td>3</td> <td>2</td> <td>2</td> <td>3</td> <td>1</td> <td>1</td> <td>5</td> <td>0</td> <td>0</td> <td>4</td> <td>3</td> <td>5</td>	1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
5   4   2   1   2   3   3   2   5   4   0   1   2   4   4   2     10   8   1   2   2   3   2   0   1   0   2   3   2   1   1     5   8   2   5   4   4   2   4   1   4   2   4   1     5   8   0   0   1   2   3   1   1   3   4   2   4   1   4   3   4   2   4   1   4   2   4   1   3   4   3   4   2   1   3   4   2   1   1   3   4   3   4   2   4   1   1   3   4   3   4   2   4   1   1   3   4   3   4   3   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1<	2	4	2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
5   8   2   5   4   4   2   4   1   4   2   6   1   4   2   6   1   4   2   6   1   4   1   4   1   4   2   1   3   1   1   4   2   3   1   3   2   6   4   3   4   2   4   1     5   8   0   0   1   2   3   1   1   3   1   1   2   1   5   1   2     9   7   3   4   2   5   1   5   0   4   4   3   4   3   2   1   1     10   2   1   2   1   4   2   3   2   4   3   4   3   2   1   1     1   8   3   4   5   3   2   4   3   3   3   2   1   1     2   4   1   1   4 <t< td=""><td>5</td><td>4</td><td>2</td><td>1</td><td>2</td><td>3</td><td>3</td><td>2</td><td>5</td><td>4</td><td>0</td><td>1</td><td>2</td><td>4</td><td>4</td><td></td><td>2</td></t<>	5	4	2	1	2	3	3	2	5	4	0	1	2	4	4		2
5   8   2   5   4   4   2   4   1   4   2   6   0   2   3   1   3     1   1   4   4   3   4   2   1   3   2   4   3   4   2   4   1     5   8   0   0   1   2   3   1   1   3   4   3   4   2   4   1     9   7   3   4   2   5   1   5   0   4   4   3   4   3   2   1   1     10   2   1   2   1   4   2   3   2   4   3   4   3   2   1   1     1   8   3   4   5   3   2   4   3   3   2   3   1     1   8   3   4   5   3   2   4   3   3   3   2   1   1     2   4	10	8	1	2	2	3	2	0	1	0	2	3	2		5	2	1
5   8   0   0   1   2   3   1   1   3   2   1   3   4   2   4   1     9   7   3   4   2   5   1   5   0   4   4   3   4   3   2   1   1     10   2   1   2   1   4   2   3   2   2   3   5   0   3   2   3   1     1   8   3   4   5   3   2   4   3   4   3   3   2   3   1     2   10   3   3   5   2   2   4   4   5   3   3   2   2   1   1   3   3   2   2   1   1   3   3   2   1   1   3   3   2   1   1   3   3   2   4   3   3   2   4   0   3   2   4   0   3   2   4 <td< td=""><td>5</td><td>8</td><td>2</td><td>5</td><td>4</td><td>4</td><td>2</td><td>4</td><td>1</td><td>4</td><td>2</td><td>1</td><td></td><td>2</td><td>3</td><td>1</td><td>3</td></td<>	5	8	2	5	4	4	2	4	1	4	2	1		2	3	1	3
9   7   3   4   2   5   1   5   0   4   4   3   4   3   2   1	1	1	4	4	3	4	2	1	3	2	5	4	3	4	2	4	1
10   2   1   2   1   4   2   3   2   1   3   5   0   3   2   3   1     1   8   3   4   5   3   2   5   3   2   4   3   4   3   3   2   5     2   10   3   3   5   2   2   2   4   5   3   3   2   2   1   1     2   4   1   1   5   5   4   3   4   3   3   2   4   2   1   1   3     1   2   3   3   4   3   3   2   4   3   2   4   2   1   1   3     1   2   3   3   4   3   3   2   4   3   2   4   0   3   2   4   0   3   2   4   0   3   2   4   0   3   2   4   0   3   <	5	8	0	0	1	2	3	1	1	3		1	2	1	5	1	2
1   8   3   4   5   3   2   3   2   4   3   4   3   3   2   5     2   10   3   3   5   2   2   4   5   3   3   3   2   1   1     2   4   1   1   5   4   3   4   3   3   2   4   1   1   3     1   2   3   3   4   3   3   2   4   3   2   4   2   1   1   3     2   5   3   3   2   1   1   5   4   5   3   3   2   4   0     2   5   3   3   2   1   1   5   4   5   3   3   2   4   0   3   2     3   3   2   1   1   5   4   5   3   3   2   4   0   3   2   4   0   3	9	7	3	4	2	5	1	5	0	4	4	3	4	3	2	1	1
2 10 3 3 5 2 2 1 4 5 3 3 3 2 2 1 1   2 4 1 1 5 4 3 4 3 3 2 4 2 1 1   1 2 3 3 4 3 3 2 4 3 2 4 2 1 1 3   2 5 3 3 2 1 1 5 4 5 3 3 2 4 0   2 5 3 3 2 1 1 5 4 5 3 3 2 4 0	10	2	1	2	1	4	2	3	2		3	5	0	3	2	3	1
2 4 1 1 5 4 3 3 2 4 2 1 1 3   1 2 3 3 4 3 3 2 4 3 2 4 0   2 5 3 2 1 1 5 4 5 3 3 2 4 0   2 5 3 3 2 1 1 5 4 5 3 3 2 4 0 3 2	1	8	3	4	5	3	2	5	3	2	4	3	4	3	3	2	5
1 2 3 3 4 3 3 2 4 3 2 4 2 1 1 3   1 2 3 3 4 3 3 2 4 3 2 5 3 2 4 0   2 5 3 3 2 1 1 5 4 5 3 3 2 4 0 3 2	2	10	3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
1 2 3 5 3 4 3 3 2 4 3 2 5 3 2 4 0   2 5 3 3 2 1 1 5 4 5 3 3 2 4 0 3 2	2	4	1	1	5	5	4		4	3	3	2	4	2	1	1	3
3 3 2 1 1 5 4 5 3 3 2 4 0 3 2	1	2	3	5				3	2	4	3	2	5	3	2	4	0
	2	5	3	3	2	1	1	5	4	5	3	3	2	4	0	3	2
	9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		

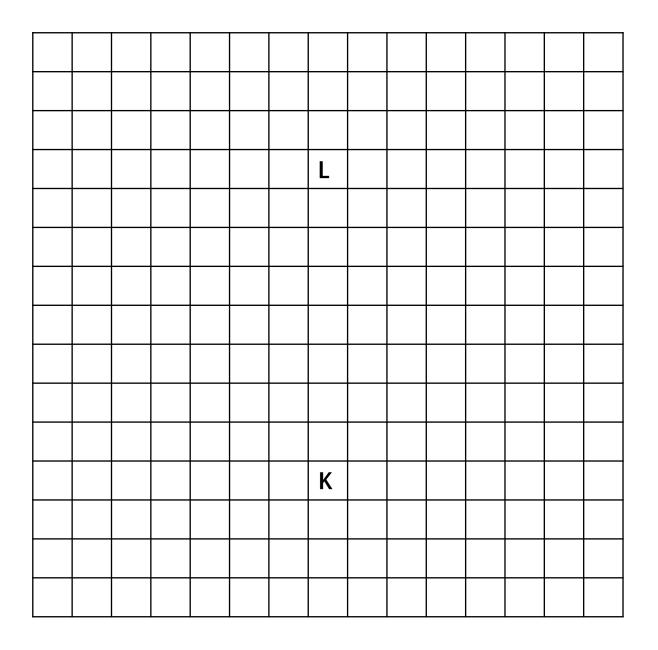
#### 案例4

从起点G到终点H的路径搜寻结果 *(右上→左下,长距离)* 



案例4

从起点G到终点H的路径搜寻结果 *(右上→左下,长距离)* 

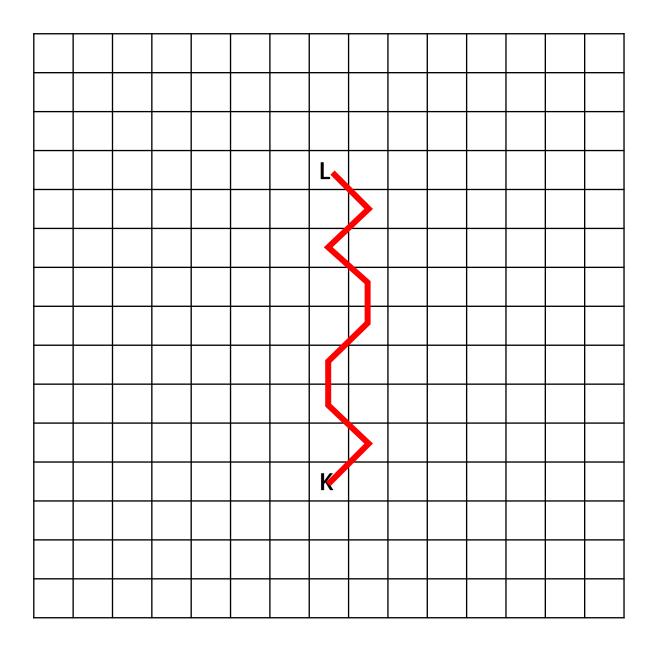


从起点K到终点L的路径搜寻 \_*(下→上,在一列)*\_ P1层

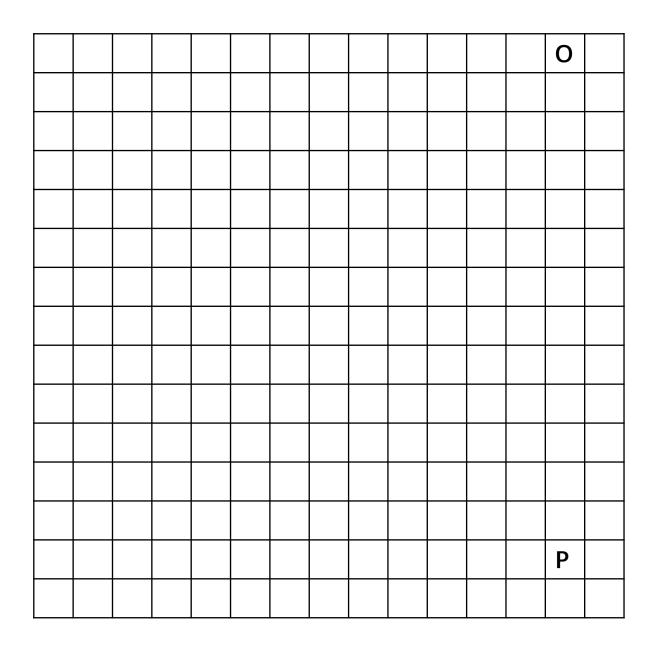
		1 1/	_													
P1+1		1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	4	2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
5	4	2	1	2	3	3	2	5	4	0	1	2	4	4	4	2
10	8	1	2	2	3	2	0	1	0	2	3	2	4	5	2	1
5	8	2	5	4	4	2	4	1		2	4	0	2	3	1	3
1	1	4	4	3	4	2	1	3	2		4	3	4	2	4	1
5	8	0	0	1	2	3	1	1	3	4	1	2	1	5	1	2
9	7	3	4	2	5	1	5	0	1	4	3	4	3	2	1	1
10	2	1	2	1	4	2	3	2	1	3	5	0	3	2	3	1
1	8	3	4	5	3	2	5	3	2	*	3	4	3	3	2	5
2	10	3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
2	4	1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
1	2	3	5	3	4	3	3	2	4	3	2	5	3	2	4	0
2	5	3	3	2	1	1	5	4	5	3	3	2	4	0	3	2
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		

案例5

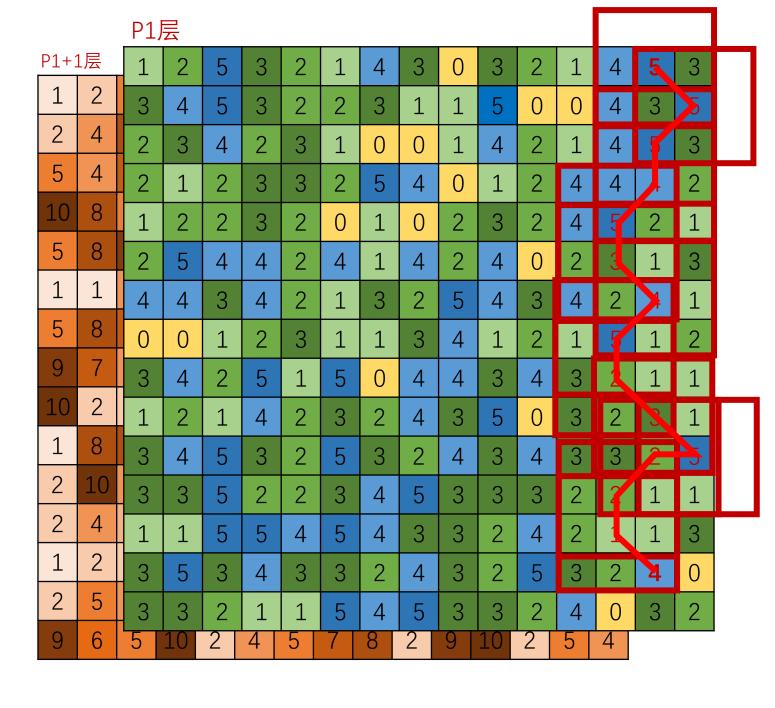
从起点K到终点L的路径搜寻结果 \_*(下→上,在一列)*\_



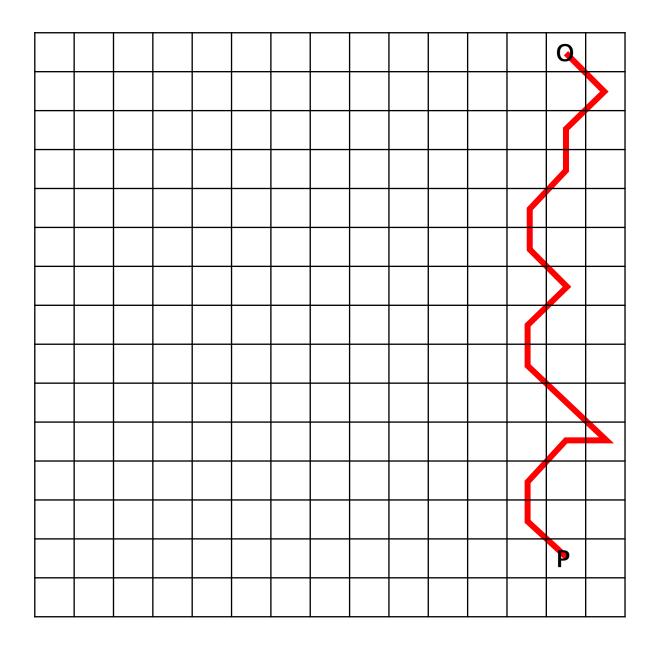
从起点K到终点L的路径搜寻结果 \_*(下→上,在一列)*\_



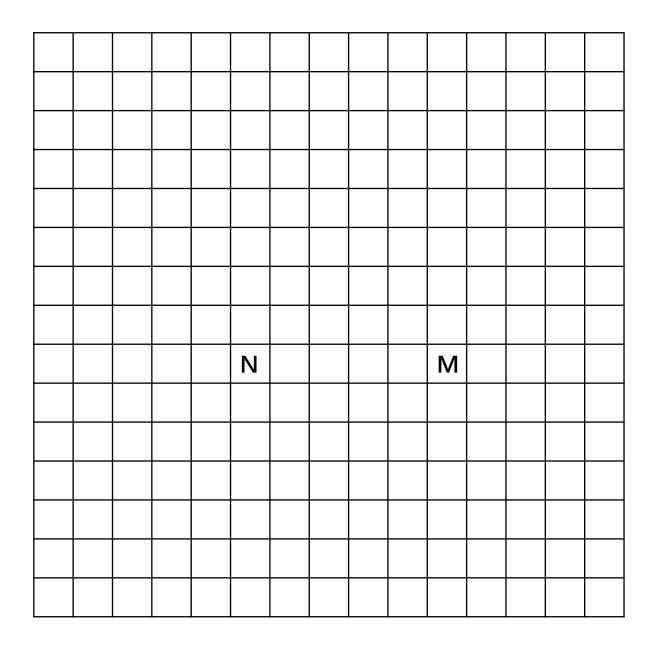
从起点O到终点P的路径搜寻 *(上→下,长距离)* 



从起点O到终点P的路径搜寻结果 *(上→下,长距离)* 



从起点O到终点P的路径搜寻结果 \_*(上→下,长距离)*\_

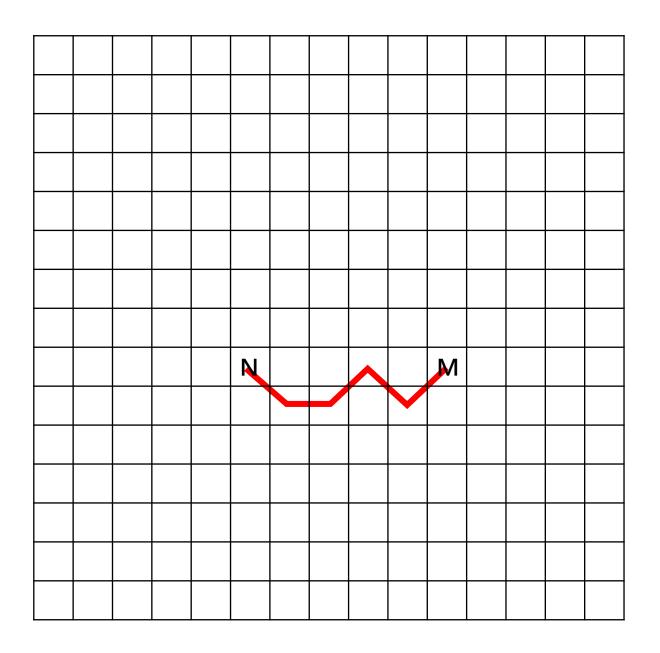


从起点M到终点N的路径搜寻 *(右→左,近距离)*  P1层

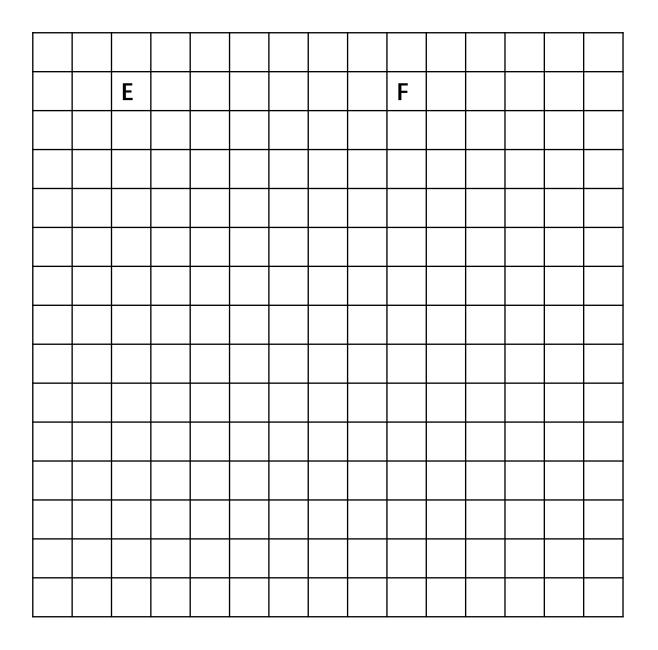
P1+	1层	1	2	5	3	2	1	4	3	0	3	2	1	4	5	3
1	2	3	4	5	3	2	2	3	1	1	5	0	0	4	3	5
2	4	2	3	4	2	3	1	0	0	1	4	2	1	4	5	3
5	4	2	1	2	3	3	2	5	4	0	1	2	4	4	4	2
10	8	1	2	2	3	2	0	1	0	2	3	2	4	5	2	1
5	8	2	5	4	4	2	4	1	4	2	4	0	2	3	1	3
1	1	4	4	3	4	2	1	3	2	5	4	3	4	2	4	1
5	8	0	0	1	2	3	1	1	3	4	1	2	1	5	1	2
9	7	3	4	2	5	1	5	0	4	A	3	4	3	2	1	1
10	2	1	2	1	4	2	3	2	4	3	3	0	3	2	3	1
1	8	3	4	5	3	2	5	3	2	4	3	4	3	3	2	5
2	10	3	3	5	2	2	3	4	5	3	3	3	2	2	1	1
2	4	1	1	5	5	4	5	4	3	3	2	4	2	1	1	3
1	2	3	5	3	4	3	3	2	4	3	2	5	3	2	4	0
2	5	3	3	2	1	1	5	4	5	3	3	2	4	0	3	2
9	6	5	10	2	4	5	7	8	2	9	10	2	5	4		

案例7

从起点M到终点N的路径搜寻结果 *(右→左,近距离)* 



从起点M到终点N的路径搜寻结果 *(右→左,近距离)* 



从起点E到终点F的路径搜寻 *(左一右,在一行)* 

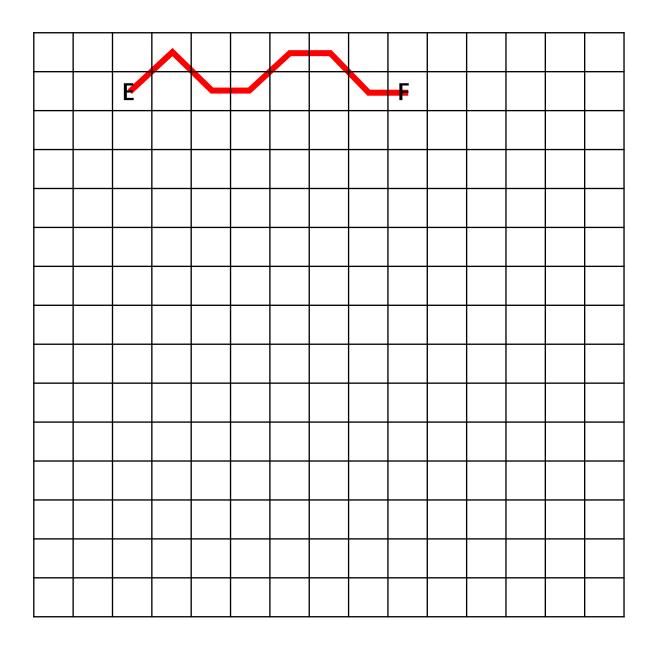
# P1层 P1+1层

#### 案例8

从起点E到终点F的路径搜寻结果 *(左一右,在一行)* 

<u>当终点栅格位于起点栅格正右方时,</u> 栅格识别前进的方向是包含上中下 的右侧栅格;

<u>当移动窗口覆盖超出栅格边界时,</u> <u>仅对于范围内有赋值的栅格执行贪</u> <u>婪选择的规则。</u>



从起点E到终点F的路径搜寻结果 \_*(左→右,在一行)*\_

