

## 广度优先搜索 Breadth First Search

基本模型：给定起点和终点，寻找可通行路线，如下图所示@代表起点，\*代表终点，求一共需要多少步才能从起点走到终点。

	1	2	3	4	5	6
1	#	.	.	.	#	#
2	#	.	@	.	.	#
3	.	#	.	.	.	.
4	.	#	.	.	#	.
5	.	.	#	.	*	#

### 数据准备：

- 1、建立Node类型结构体，用来存储每一个节点的信息，用于解答题目
- 2、建立char类型地图数据表map[行][列]，记录整个地图信息
- 3、建立bool类型访问记录表vis[行][列]，记录某个节点是否已经走过
- 4、建立Node类型一维数组或者队列searchQueue，存储后续将要访问的节点

### 遍历逻辑：

- 1、将起点加入searchQueue，并记录已访问

(2,3)					
-------	--	--	--	--	--

访问记录表

	1	2	3	4	5	6
1						
2						
3						
4						
5						

- 2、当searchQueue内存在未访问节点时，持续进行循环搜索。

- 3、从searchQueue中取出第一个节点(2,3)，对其四个相邻的点进行判断。

地图表

	1	2	3	4	5	6
1	#	.	.	.	#	#
2	#	.	@	.	.	#
3	.	#	.	.	.	.
4	.	#	.	.	#	.
5	.	.	#	.	*	#

访问记录表

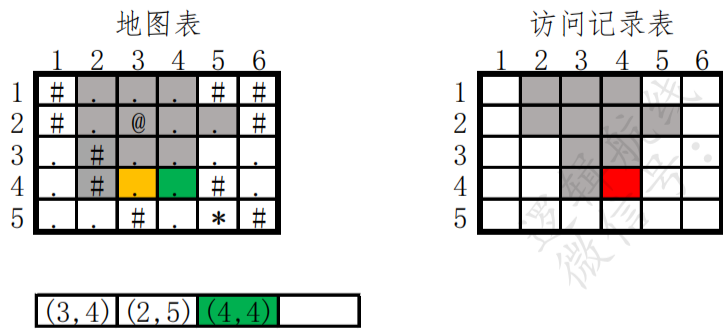
	1	2	3	4	5	6
1						
2						
3						
4						
5						

对于当前数据，其四个点都可以通行，同时，它们又不等于终点，因此，我们依次将其加入searchQueue，并将其记录为已访问。

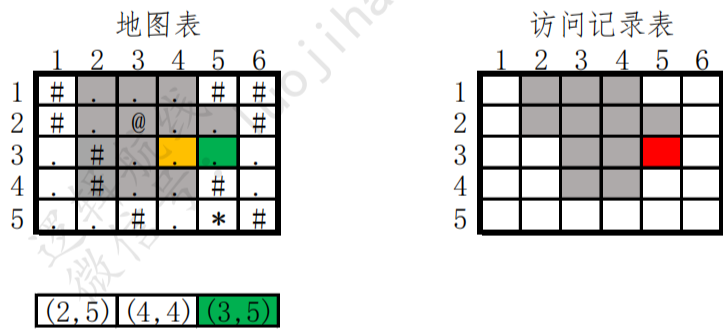
(1,3)	(2,2)	(3,3)	(2,4)			
-------	-------	-------	-------	--	--	--



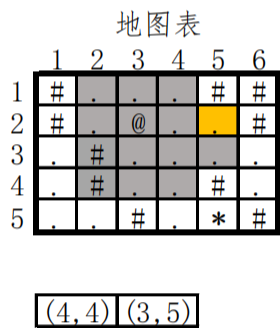
4-6 继续搜索，以(4,3)为起点，结果如下：



4-6 继续搜索，以(4,3)为起点，结果如下：



4-7 继续搜索，以(2,5)为起点，无路可走，结果如下



4-8 继续搜索，以(4,4)为起点，增加新路点，结果如下

