1. **MAZE\_NODE**

Top-wall

Bottom-wall

Right-wall

Left-wall

1. **Initialize Maze**

15

14

13

12

11

10

9

8

7

6

5

4

3

2

0

1

1. 构建M\*N的网格
2. 记录每个节点的邻接节点和邻接关系
3. 构建Maze的算法（DFS）

将0号节点设为当前节点

While（存在未被访问的节点）

{

将当前节点设为已被访问，并且push到记录访问路径的栈中。

如果当前节点有未被访问的邻接节点，则随机访问当前节点的一个未被访问的邻接节点，推倒当前节点和邻接节点之间的墙，将邻接节点设为当前节点。

否则，{

将记录访问路径的栈顶元素出栈。

如果栈不为空，则将栈顶元素设为当前节点。

否则，随机选择一个未被访问过的节点设为当前节点。

}

}

1. 求解Maze的算法（DFS）

1将0号节点设为当前节点，将最后一个节点设为出口节点。

2.while（当前节点 ！= 出口节点）

{

将当前节点设为已被访问，并push到求解路径的栈中。

如果当前节点存在可到达的未被访问的邻接节点，则选择

一个设为当前节点。

否则，{

将栈顶元素出栈。

如果栈不为空，取栈顶元素设为当前节点。

否则，返回求解失败！

}

}