## 实验一 图搜索与问题求解

### 实验1 启发式搜索

**一 实验目的**

1 熟悉和掌握启发式搜索的定义、估价函数和算法过程；

2 理解和掌握启发式搜索过程，能够用选定的编程语言求解八数码问题，理解求解流程和搜索顺序；

3 比较并分析图搜索策略的实质，通过实验理解启发式搜索的意义。

**二 实验预习内容**

1了解重排九宫/八数码问题、一字棋游戏、八皇后问题；

2 各种图搜索算法及剪枝技术等。

**三 实验内容**

1 八数码问题

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 8 | 3 |  | 1 | 2 | 3 |
| 1 | 6 | 4 | 8 |  | 4 |
| 7 |  | 5 | 7 | 6 | 5 |

初始棋局 目标棋局

图 1 八数码问题示例

在一个3\*3的方格棋盘上放置8个标有1、2、3、4、5、6、7、8数字的将牌，留下一个空格（一般用0表示），规定与空格上下左右相邻的将牌可以移入空格。问题的解是要求寻找一条从某初始状态S0到目标状态Sg的将牌移动路线（如图1所示）

2 问题描述

要求用某种启发式搜索方法（推荐选用A\*算法）求解从给定的初始状态到目标状态的移动路线。

**四 实验要求**

1 自己定义启发式函数,能正确求解出从初始状态到目标状态的移动路线；

A\*算法是一种启发式搜索算法，用于在搜索图中寻找从初始节点到目标节点的最佳路径。在每一步，从所有叶节点中选择一个节点进行扩展。为了尽快找到一条从初始节点到目标节点的路径，A\*算法采用评价函数f(n) = g(n) + h(n)，其中：

- n是待评价的节点。

- g(n)是从初始节点s到节点n的最佳路径耗散值的估计。

- h(n)是从节点n到目标节点t的最佳路径耗散值的估计，称为启发函数。

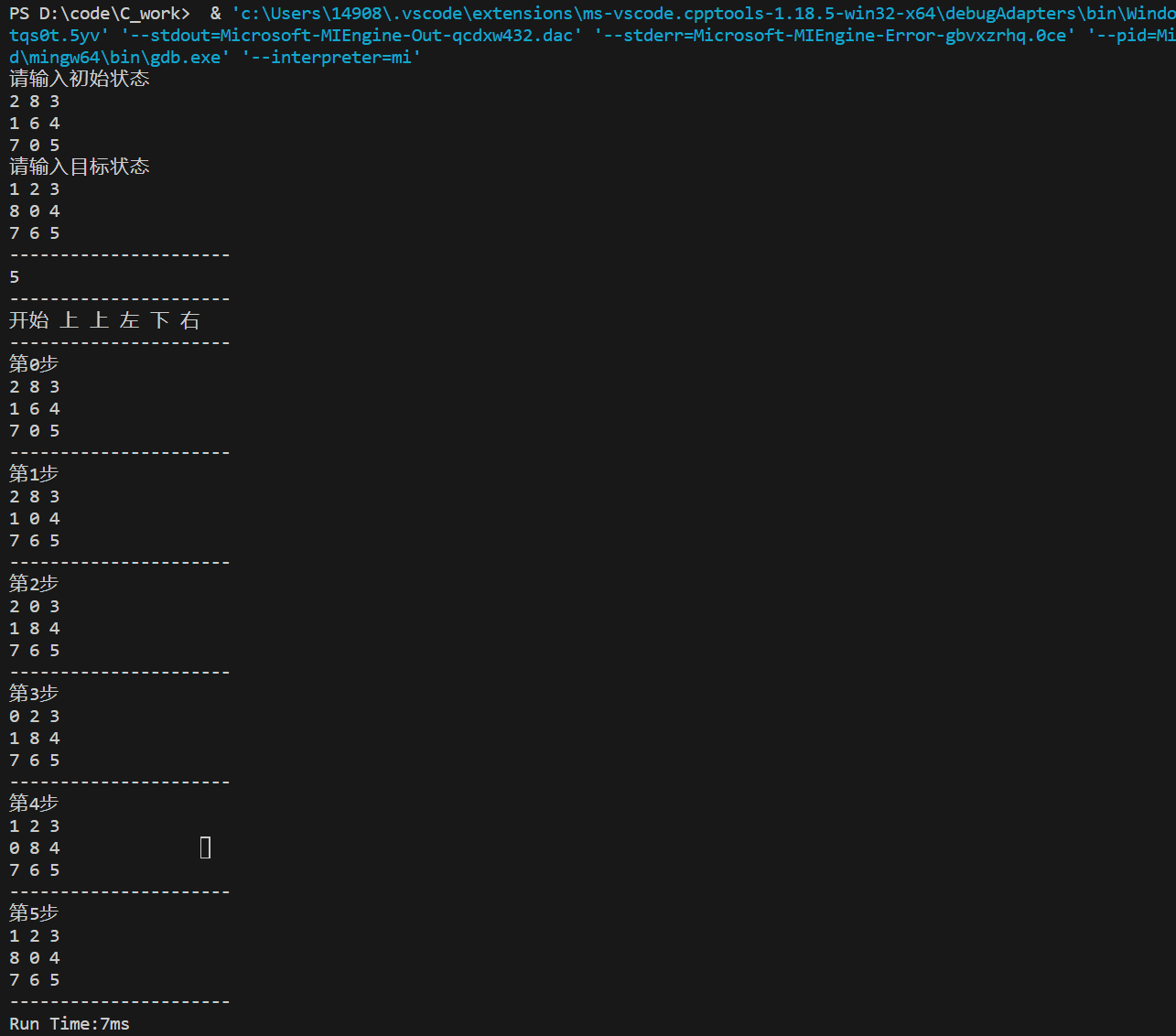
- f(n)是从初始节点s经过节点n到达目标节点的最佳路径耗散值的估计，也是评价函数。

在每次扩展时，A\*算法选择具有最小评价函数值f(n)的节点。其中，g(n)表示已经走过的路径的实际代价，而h(n)是从当前节点到目标节点的估计代价。当启发函数满足h(n) ≤ h\*(n)时，其中h\*(n)是从节点n到目标节点t的最优实际耗散值，可以证明在问题有解的情况下，A\*算法一定会找到一条耗散值最小的路径。这个条件的满足使得A\*算法成为一种优秀的启发式搜索算法。

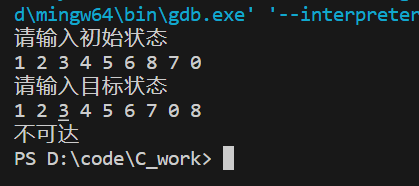
因此我们针对八数码问题使用A\*算法，其中g（n）表示当前结点和初始结点的距离，h(n)为当前状态和目标状态不一致的数字的个数

2 要求界面显示初始状态、目标状态和中间搜索步骤；

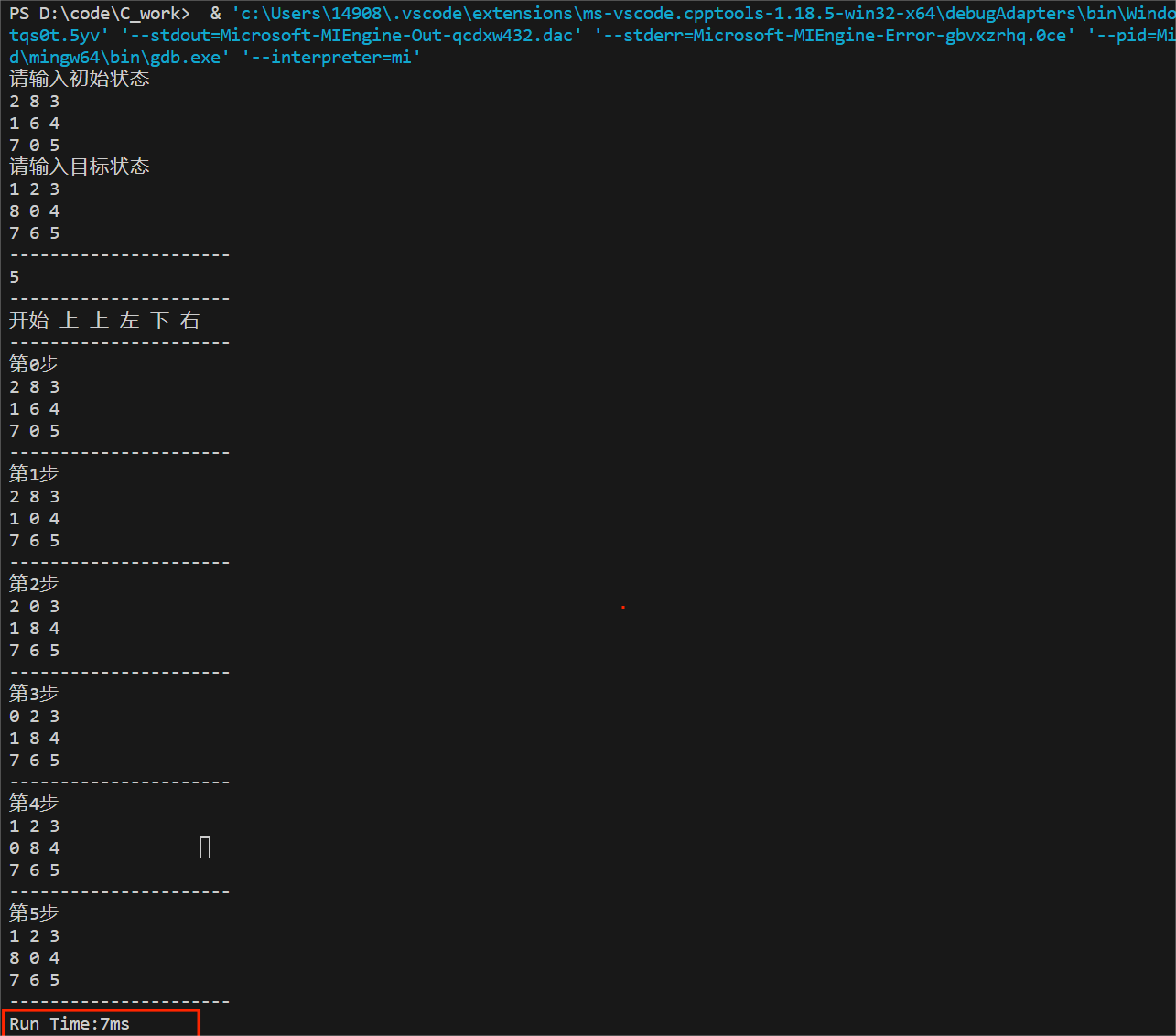
运行结果：



3 对不可达状态能进行正确识别；



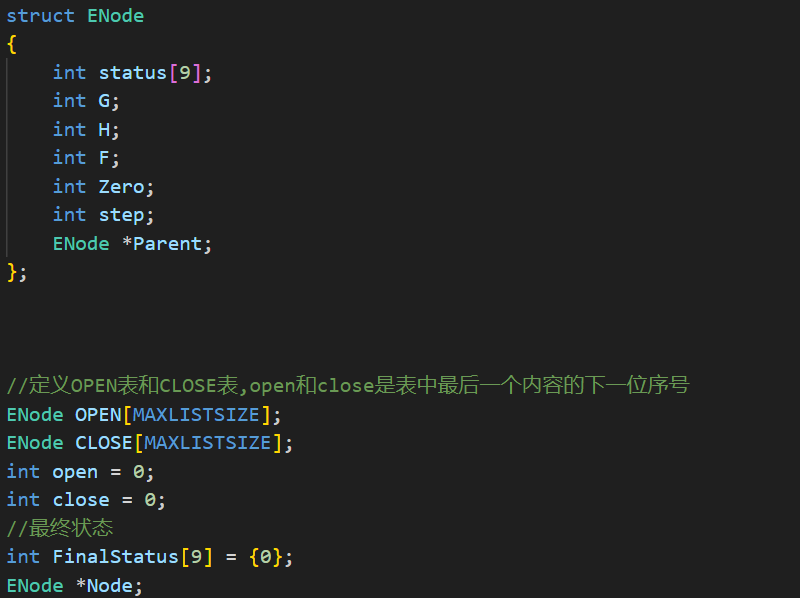
4 对所采用的启发式函数做出性能分析。



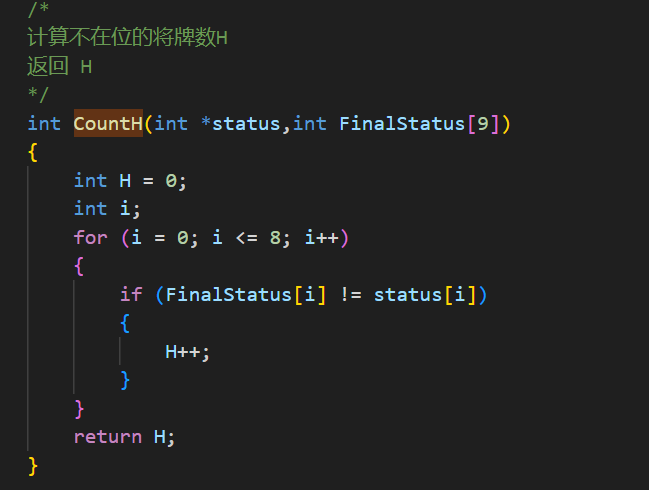
可以看到对于我们实验要求中的示例，我们执行时间仅用了7ms，效率还是非常高的。

**五 代码分析**

1. **定义八数码结构体和全局变量**



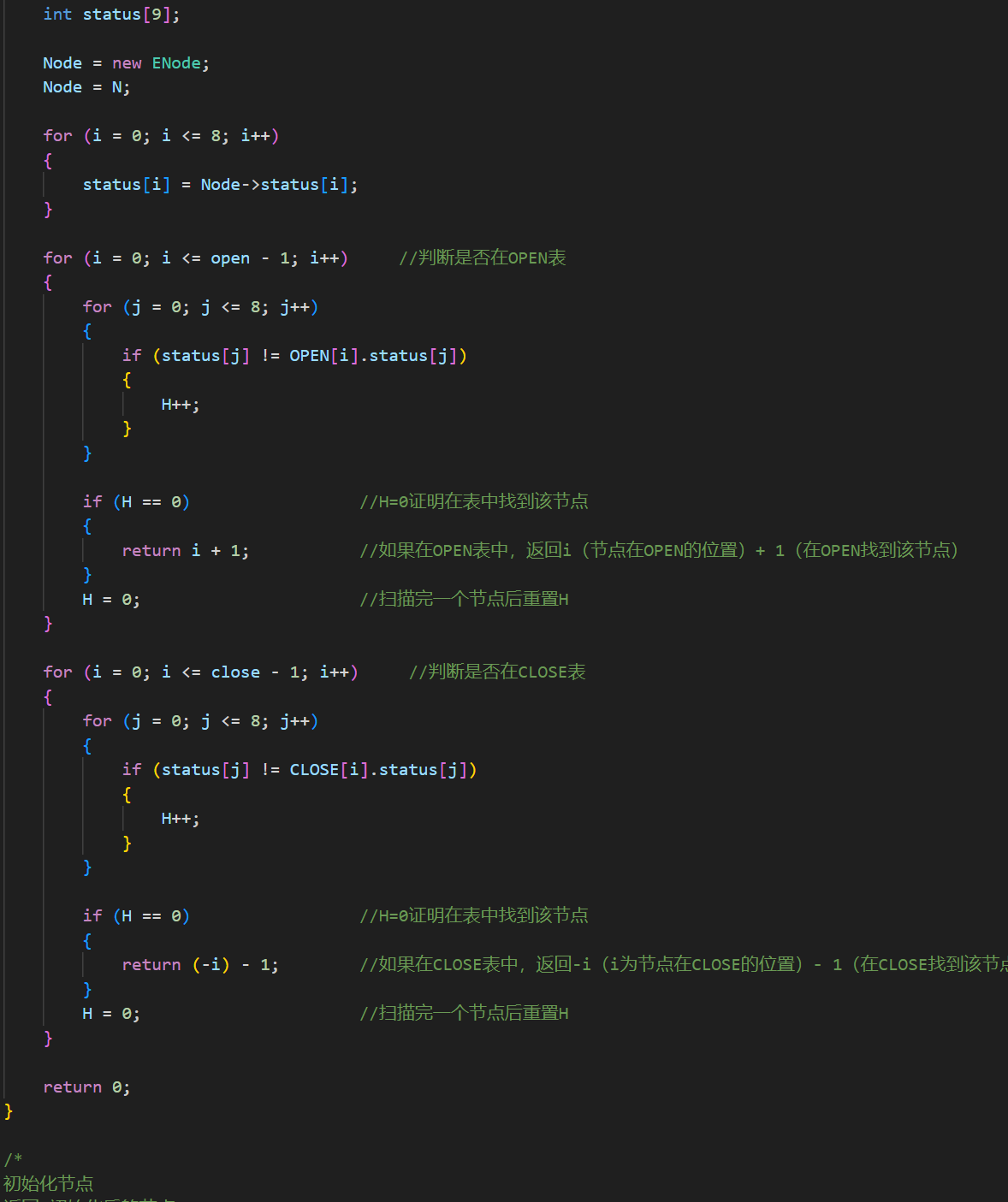
1. ENode 结构体表示八数码问题中的节点，包括状态、步数、不在位的牌数等信息。
2. OPEN 和 CLOSE 分别是存储待扩展和已扩展节点的数组。
3. open 和 close 是记录 OPEN 和 CLOSE 数组的末尾元素位置的指针。
4. FinalStatus 存储目标状态。
5. Node 是一个全局指针，用于辅助操作节点。
6. 计算启发式函数H的数值CountH



CountH 函数计算当前状态与目标状态之间的不在位的牌数，即 H 值。

通过比较每个位置上的牌，不同的数量即为不在位的牌数。

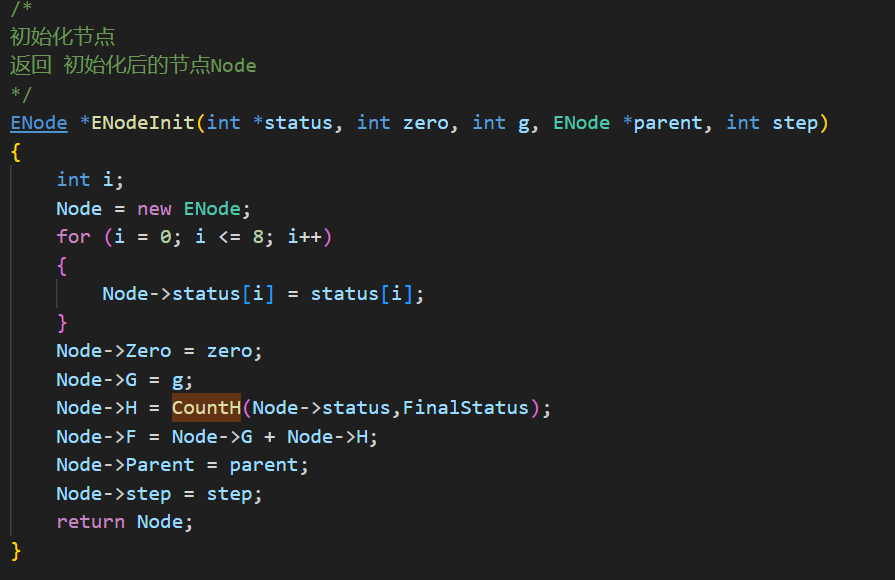
1. 判断结点是否存在与open与close表中

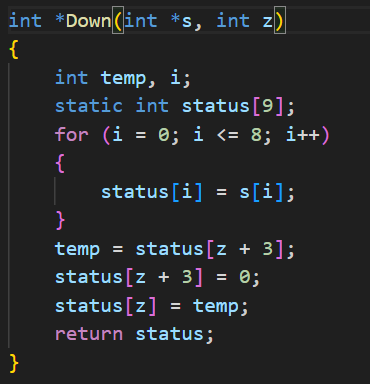
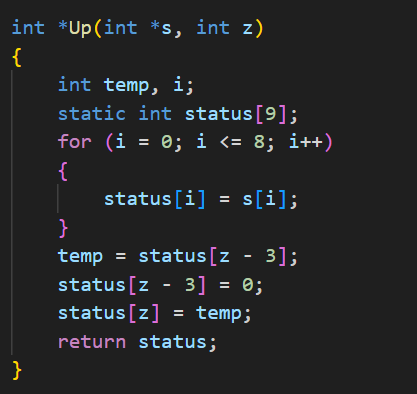
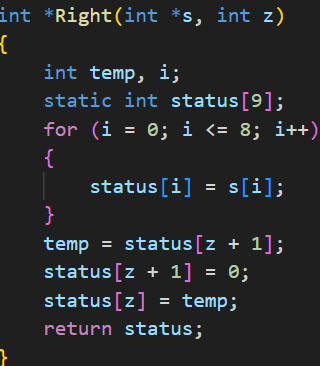
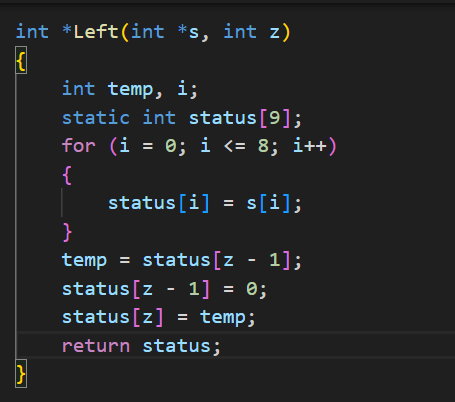


Exist 函数用于判断新生成的节点是否已存在于 OPEN 表或 CLOSE 表中。

通过比较状态数组判断节点是否相同，返回值表示存在于 OPEN 表（正数）、CLOSE 表（负数）或均不在表中（0）。

1. 初始化与状态扩展

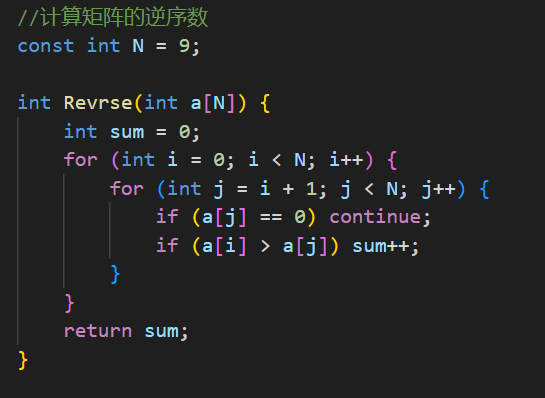




ENodeInit 函数用于初始化新的节点，设置状态、零牌位置、步数、父节点等信息。

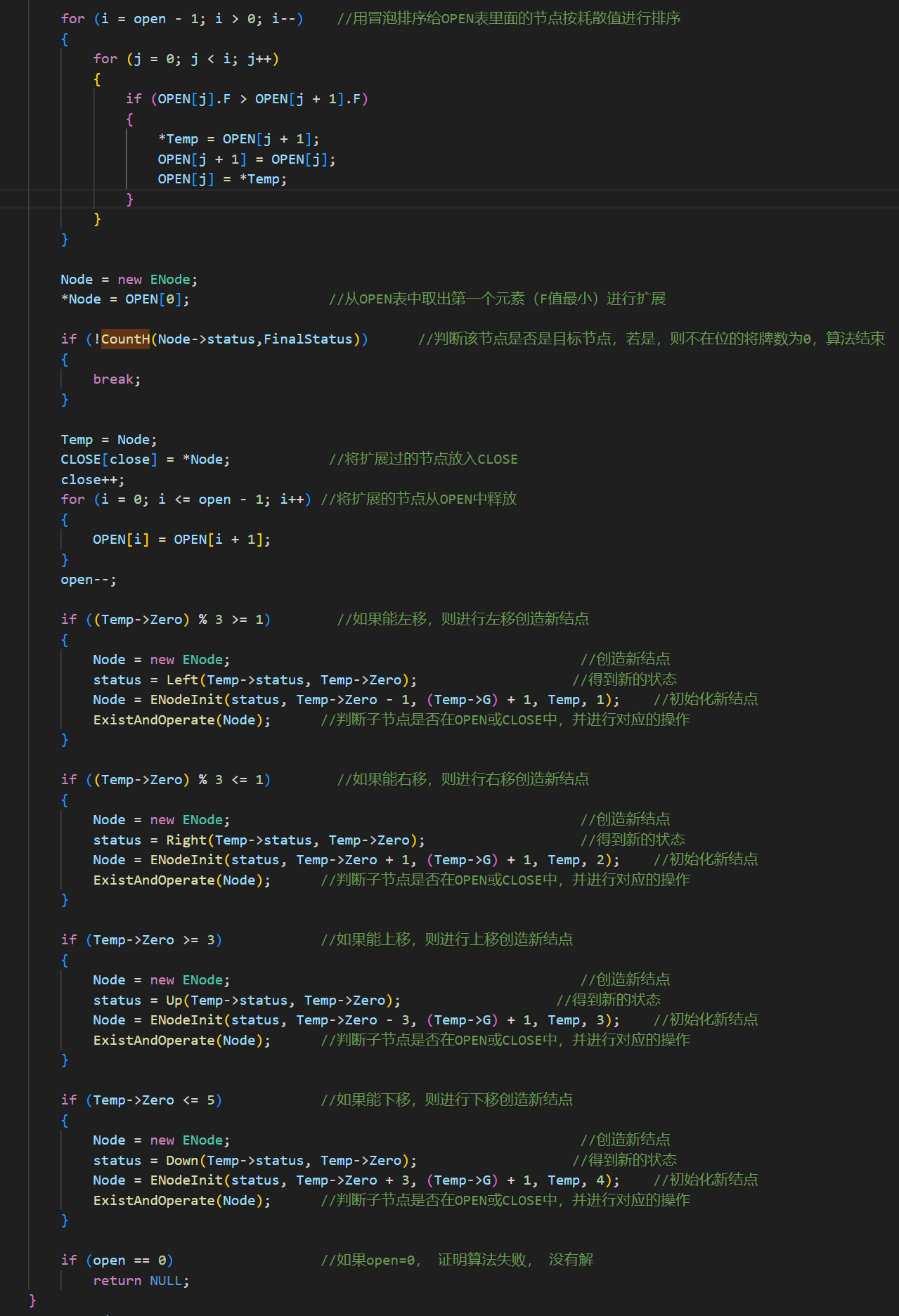
上下左右这些函数分别实现了左移、右移、上移和下移的操作，返回移动后的状态数组。

1. 判断是否存在解，根据数列的奇偶性



Revrse 函数用于计算状态数组的逆序数，用于判断是否有解

1. A\*算法



Astar 函数是 A\* 算法的主体，通过循环选择 F 值最小的节点进行扩展，将扩展后的结点放到open表中进行排序，将H值最小的放在最前面并取出给到closed表中，然后再根据新的结点进行扩展，循环，直到找到目标状态或 OPEN 表为空。

1. **实验总结**