**八数码三种策略自动求解可视化**

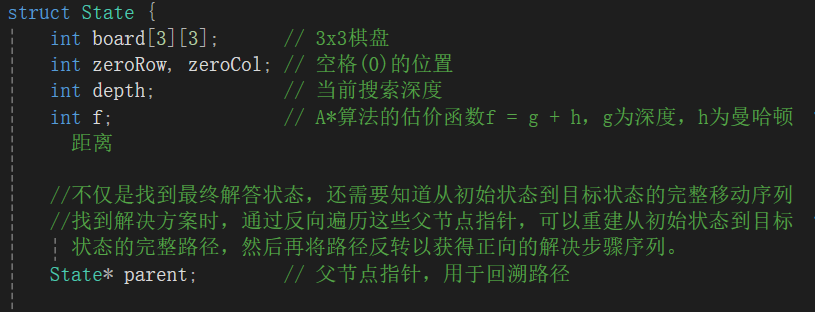
**一、项目概述**

八数码问题是一个经典的人工智能搜索问题，其目标是通过移动空格，将初始的 3×3 数字棋盘布局变换为目标布局。本项目实现了三种常见的搜索策略：A \* 算法、迭代加深搜索（IDS）和贪婪最佳优先搜索，并通过 EasyX 图形库实现可视化展示，方便用户直观地观察不同搜索策略的求解过程。

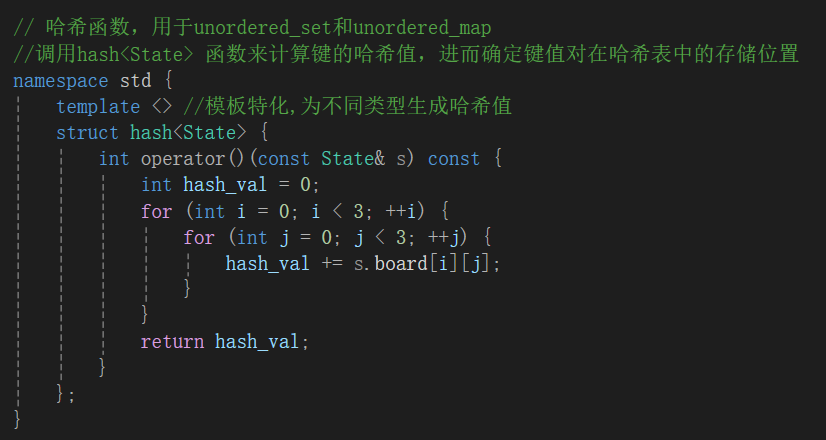
**二、系统设计**

**2.1 数据结构设计**

**State 结构体：**用于表示八数码问题的一个状态，包含棋盘布局、空格位置、搜索深度、估价函数值（仅用于 A \* 和贪婪最佳优先搜索）以及父节点指针（用于回溯路径）。



**哈希函数：**为 State 结构体定义哈希函数，以便在哈希集合和哈希映射中使用。



**2.2 算法设计**

**A \* 算法：**结合当前状态的深度（g 值）和曼哈顿距离（h 值）作为估价函数 f = g + h，通过优先队列优先扩展 f 值最小的状态，确保找到最优解。

**迭代加深搜索（IDS）：**从深度 1 开始逐步增加深度限制，调用深度受限搜索（DLS）函数，在每个深度限制下进行深度优先搜索，直到找到解或达到最大深度。

**贪婪最佳优先搜索：**仅使用曼哈顿距离作为启发式函数，优先扩展当前状态下曼哈顿距离最小的状态，虽然效率高，但不能保证找到最优解。

**2.3 可视化设计**

**图形界面布局：**

窗口大小：600×700 像素

棋盘区域：位于窗口中部，大小为 300×300 像素

信息展示区域：位于棋盘下方，显示求解步数和耗时

操作说明区域：初始状态下显示操作指令

**绘制函数：**

startup()：初始化图形窗口，设置目标状态和初始状态。

show()：绘制棋盘、当前状态、步数信息和操作说明。

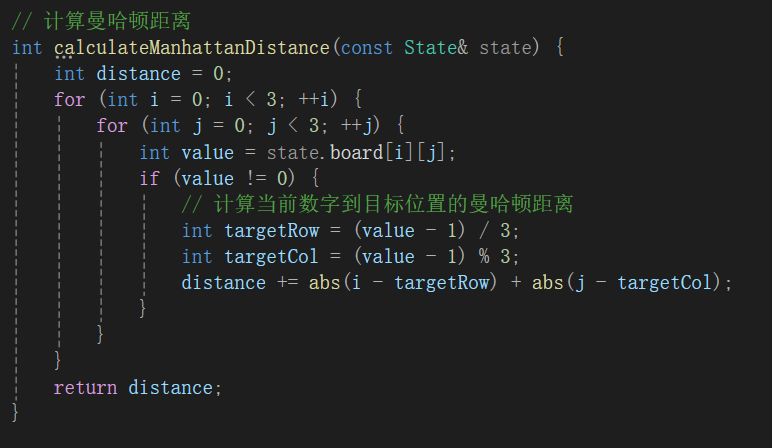
updateWithoutInput()：在求解过程中自动更新当前步骤。

updateWithInput()：处理用户按键输入，启动不同算法或重新生成初始状态。

1. **核心代码实现**

**3.1 状态操作**

**计算曼哈顿距离：**

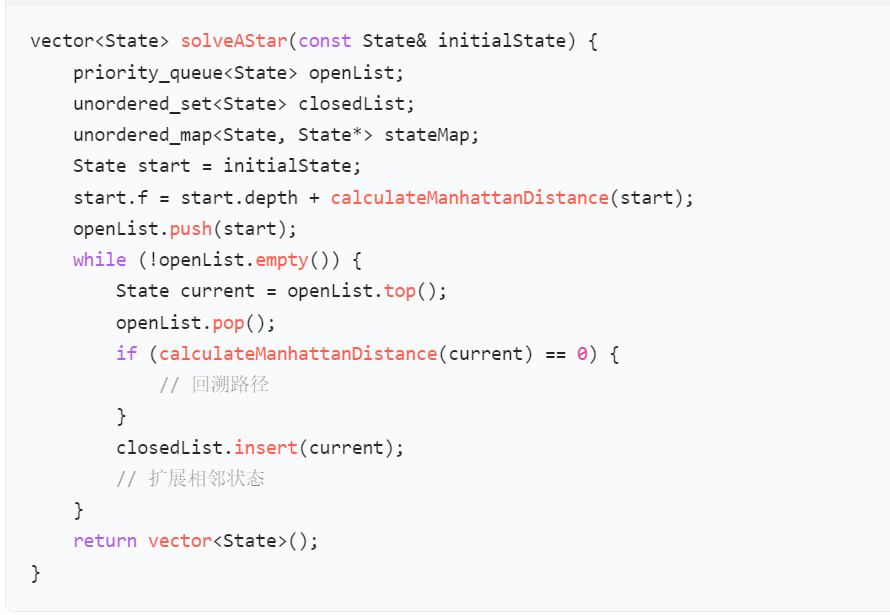


**检查状态是否可解：**

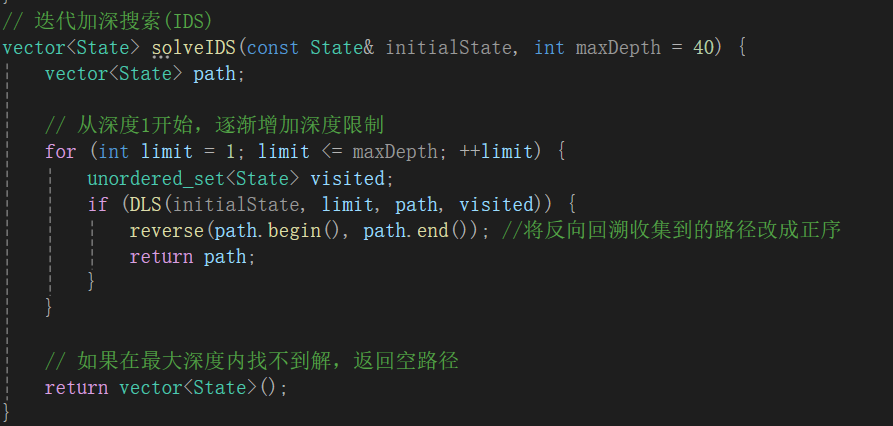


**3.2 搜索算法**

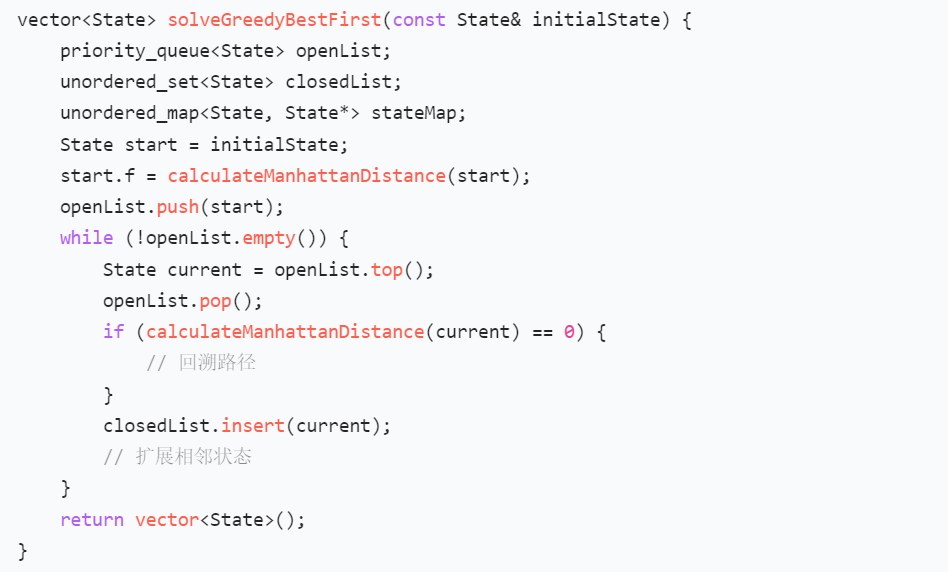
**A\*算法：**



**迭代加深搜索：**



**贪婪最佳优先：**

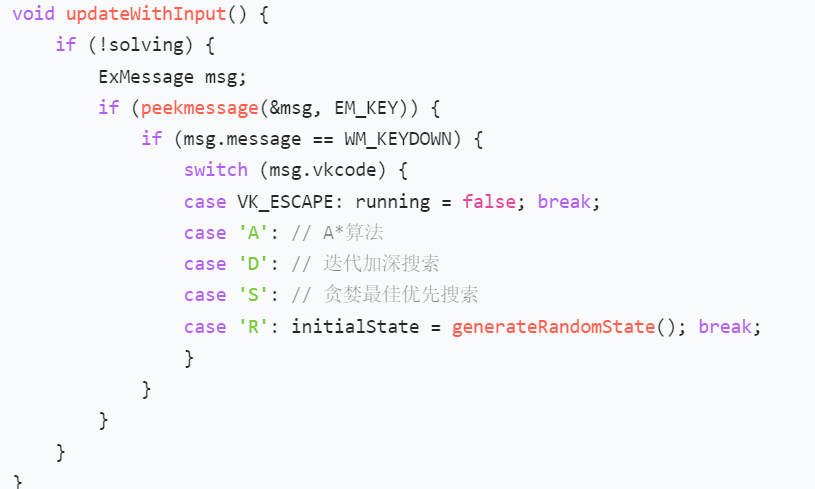


**3.3 可视化**

**绘制棋盘状态：**



**用户交互：**



1. **使用说明**

1、运行程序：启动可执行文件，显示初始状态和操作说明。

2、操作指令：

A：使用 A \* 算法求解

D：使用迭代加深搜索求解

S：使用贪婪最佳优先搜索求解

R：重新生成随机初始状态

ESC：退出程序

1. 观察结果：求解过程中，界面自动展示每一步移动，完成后显示总步数和耗时。
2. **性能分析**

**A \* 算法：**保证找到最优解，但时间复杂度较高，适用于对解的最优性要求严格的场景。

**迭代加深搜索：**在一定深度限制内找到解，时间复杂度介于广度优先搜索和深度优先搜索之间。

**贪婪最佳优先搜索：**求解速度快，但不能保证最优解，适用于对时间要求较高且对解的质量要求不严格的场景。