实验一 ：启发式搜索

**【实验目的】**

启发式搜索就是利用知识来引导搜索，达到减少搜索范围，降低问题复杂度的目的。本实验通过解决八数码问题，帮助学生更好的熟悉和掌握启发式搜索的定义、估价函数和算法过程。

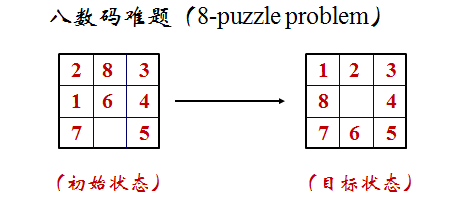
【实验内容】

利用启发式搜索（A\*算法）解决8数码问题

8数码规则：在 3× 3 的方格棋盘上，分别放置了表有数字 1、2、3、4、5、6、7、8 的八张牌，初始状态 S0，目标状态 Sg，如下图所示。

可以使用的操作有空格左移，空格上移，空格右移，空格下移

即只允许把位于空格左、上、右、下方的牌移入空格。请编程实现该问题的解决

、

【实验原理】

一、搜索策略简介

1. 无信息的搜索策略——盲目搜索

在不具有对特定问题的任何有关信息的条件下，按固定的步骤（依次或随即调用操作算子）进行的搜索，它能快速地运用一个操作算子。盲目搜索中，由于没有可参考的信息，因此只要能匹配的操作算子都须运用，这会搜索更多的状态。

2. 有信息的搜索——启发式搜索

考虑特定问题领域可应用的知识，动态地确定调用操作算子的步骤，优先选取较合适的操作算子，尽量减少不必要的搜索，以求尽快地到达结束状态，提高搜索效率。

启发式搜索每使用一个操作算子便需做更多的计算与判断，其性能一般要优于盲目搜索，但不可过于追求更多的甚至完整的启发信息。

解决。

二、搜索算法

1. 辅助用的数据结构

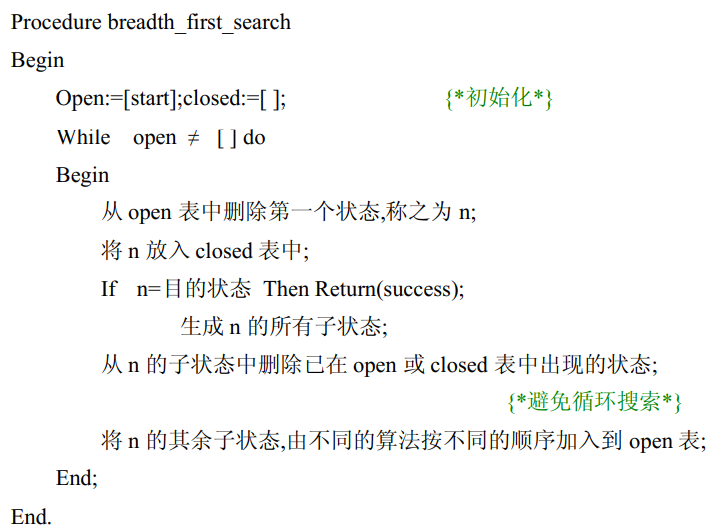
open 表：记录已生成但其子状态未扩展的状态。

closed 表：记录其子状态已被扩展的状态。

(1) 广度优先搜索算法中的 open 表采用的是先进先出的“队列”。

(2) 深度优先搜索算法中的 open 表采用的是后进先出的“栈”。

(3) 启发式搜索算法中的 open 表中的元素是按启发估价函数值的大小排列的一个表，每次从表中优先取出启发估价函数值最小的状态加以扩展。同时并不丢弃其它的状态，而把它们保留在 open 表中。

2. 算法过程

三、启发式策略中估价函数的确定

启发式策略中的“启发” （heuristic）可以体现在三个方面：问题的表达；操作算子的构造；操作算子的选择。这里特别关注操作算子的选择——控制性启发信息。

控制性启发信息往往被反映在估价函数之中。估价函数的任务就是估计待搜索结点的“有希望”程度 (或者说估计操作算子的“性能” )，并依此给它们排定次序。

估价一个结点的价值，必须综合考虑两方面的因素：已经付出的代价(实际)将要付出的代价(估计)

f (n) = g(n) + h(n)

g(n)项保持了搜索的宽度优先成分，有利于搜索的完备性，但会影响搜索的效率，h(n)体现了搜索的启发信息，有利于搜索的效率，但影响搜索的完备性。

定义 h\*(n)为状态 n 到目的状态的最优路径的代价。对一具体问题，只要有解，则一定存在 h\*(n)。于是，当要求估价函数 f(n)中的 h(n)都小于等于 h\*(n)即h(n)≤ h\*(n) 时，A 搜索算法就成为 A\*搜索算法。

【例】八数码问题的搜索树

2 8 3 1 2 3

1 6 4 -- > 8 4

7 5 7 6 5

S0 Sg(0)

其估价函数可以定义为：

f(n) = d(n) + w(n)

其中 d(n)代表状态的深度，每步为单位代价；w(n)表示以“不在位”的将牌数。

估价函数的另一个定义为：

f(n) = d(n) + p(n)

其中 p(n) = 将牌“不在位”的距离和。

采用这两种估价函数的 A 算法都是 A\*算法。附录程序采用的估价函数为(2)式，但其 d(n)=0。

【**实验要求**】

1. 画出八数码解过程的节点状态图。

2.编程解决八数码问题

3.撰写实验报告