人工智能基础

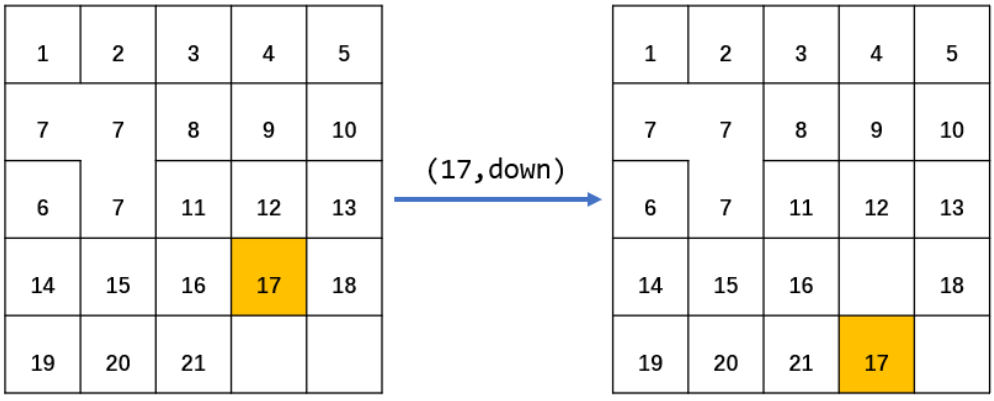
编程作业 1

完成截止时间：**2020/5/17**
提交方式：**bb 系统中提交**
助教：　**郭俊良**　[guojunll@mail.ustc.edu.cn]
　　李丹　[lidan528@mail.ustc.edu.cn]
　　李继权　[lijiquan@mail.ustc.edu.cn]
　　刘苏源　[lsysue@mail.ustc.edu.cn]
　　张铁　[tiezhang@mail.ustc.edu.cn]

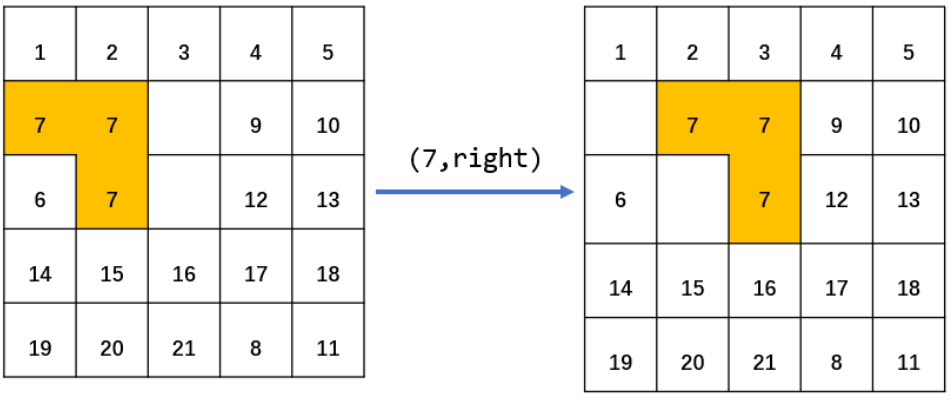
P1：数码问题（50%）

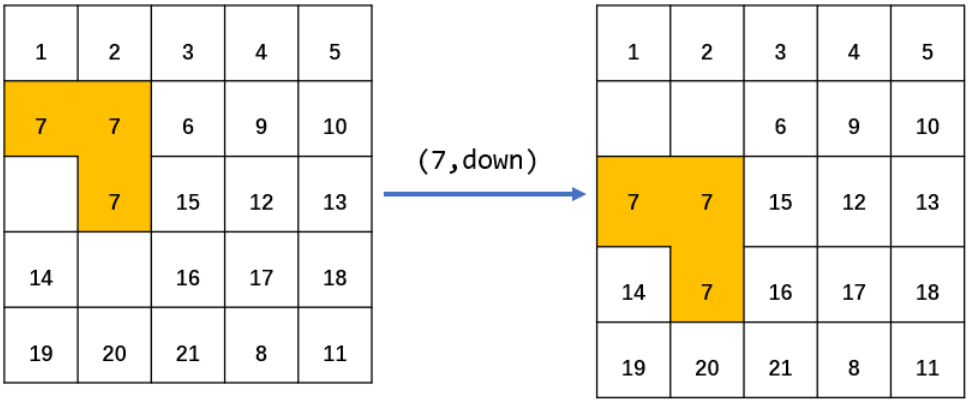
问题描述：
在一个 5*5 的网格中，23 个小方块写有数字“1-21”，剩下的两个空白方块代表空位，特别地，有三个写有“7”的方块被绑定在一起，组成了一个“7”型块。与空位上、下、左、右相邻的方块可以移动到空位中，记为一次行动；方块的移动规则为：

1. 当方块为独立的小方块（即非“数字 7”）时，若其上下左右的任一位置存在空位，其可移动到对应的空位中。



2. 当待移动方块为“7”号方块时，为表述方便，依次记其左上角部分，右上角部分，右下角部分为 a、b、c。若 b、c 的正右方均存在空位，则其可以右移；若 a、b 的正上方均存在空位，则其可以上移；若 a、c 的下方均存在空位，则其可以下移；若 a、c 的正左方均存在空位，则其可以左移；“7”型块需要作为一个整体进行移动，无法分割。下图为“7”型方块的右移与下移示例：



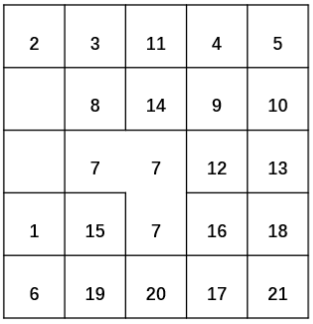


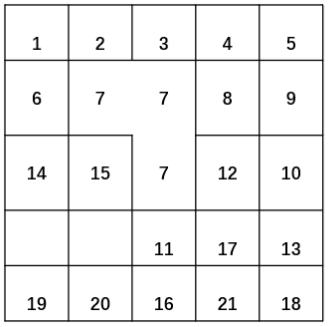
现给定初始状态与目标状态，要求获得从初始状态到目标状态的合法移动序列。

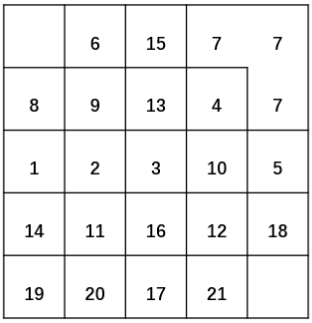
实验任务：

1. 为该问题寻找一个可采纳的启发式函数，并证明你的结论。
2. 依照你设计的启发式函数，分别实现 A*搜索算法和迭代 A*搜索算法（迭代 A*搜索算法见附页）进行以下 3 个不同难度的初始状态求解，并输出合法的解。
3. 问题的求解并不一定需要严格服从以上两个算法的流程，可以根据问题的性质加入你自己策略对算法进行微调，但算法的大致框架仍要属于 A*搜索与迭代 A*搜索算法。例如，对于该问题的一个可行但并不一定最优的解法可以为：a.采用搜索策略将“7”型数字块移动到正确的位置；b.将“7”型数字块固定视为障碍物，采用搜索策略把其他数字块移动到对应的位置。

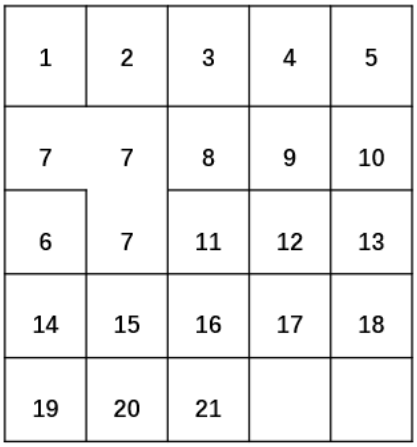
初始状态：







目标状态：



提交要求：

1. 输入输出：输入文件为 txt 格式，文件内容用于描述初始状态。每一行对应网格中的一行，行内数字代表网格中方块的数字，数字之间用英文逗号分隔，空位用“0”表示。例如对于初始状态 1，1.txt 文件的内容应如下：
   2,3,11,4,5
   0,8,14,9,10
   0,7,7,12,13
   1,15,7,16,18
   6,19,20,17,21
   输出文件为 txt 格式，文件名应与输入文件对应。内容包含了从初始状态到目标状态所经的合法移动序列。其中一次移动用 (number, direction) 表示，number∈[1,21],direction∈{u,d,l,r}，分别代表将标号为 number 的数字块上移、下移、左移、右移。移动序列之间用英文分号分隔。例如，可能的移动序列为：(1,r); (2,l); (3,u)。

2. 编程语言限定为 C/C++，请确保你的代码可正常编译运行，并在 README 文件中给出代码编译运行的方式。代码具有可读性，必要的地方应给出注释。
3. 实验报告中需要用伪代码对你的算法进行简要概括，同时给出算法的复杂度，以及不同初始状态、不同算法下的实际执行时间、得到解的步数；我们将对时间、空间复杂度优化以及解的步数优化进行相应加分。

附页：迭代 A*搜索算法

迭代 A*搜索算法的提出是为了解决 A*搜索在空间复杂度上的缺点，将迭代深入的思想用在启发式搜索上。IDA*和典型的迭代深入算法最主要的区别就是所用的截断值是 f 耗散值（g+h）而不是搜索深度；每次迭代，截断值是超过上一次迭代阶段值的节点中最小的 f 耗散值。以下为迭代 A*搜索算法。

Algorithm 3 Iterative deepening A* search (IDA*)

```
1: d̂_limit ← d̂(s_0)
2: while d̂_limit < ∞ do
3:    next_d̂_limit ← ∞
4:    list ← {s_0}
5:    while list is not empty do
6:       s ← head(list)
7:       list ← rest(list)
8:       if d̂(s) > d̂_limit then
9:          next_d̂_limit ← min(next_d̂_limit, d̂(s))
10:      else
11:         if s is a goal then
12:            return s
13:         end if
14:         newstates ← apply actions to s
15:         list ← prepend(newstates, list)
16:      end if
17:   end while
18:   d̂_limit ← next_d̂_limit
19: end while
20: return fail
```

P2：X 数独问题（50%）

问题描述：

原始的数独问题：在 9*9 格的方格中，玩家需要根据已知数字，推理出所有剩余空格的数字，并满足每一行、每一列、每一个粗线宫（3*3）内的数字均含 1-9，不重复。

下图是一个原始数独的题目与答案的例子。

* 8 * 4 * 9 6 5 3	7 8 1 4 2 9 6 5 3
6 4 2 8 * * * 7 *	6 4 2 8 5 3 9 7 1
* * * * * * 8 * *	9 3 5 1 7 6 8 2 4
* * 7 * * 5 * 4 2	1 6 7 9 8 5 3 4 2
* * * 7 * 1 * * *	3 2 9 7 4 1 5 6 8
8 5 * 6 * * 1 * *	8 5 4 6 3 2 1 9 7
* * 6 * * * * * *	4 9 6 3 1 7 2 8 5
* 1 * * * 4 7 3 6	5 1 8 2 9 4 7 3 6
2 7 3 5 * 8 * 1 *	2 7 3 5 6 8 4 1 9

现在将原始的数度问题扩展为 X 数独问题，即给它额外加上一个约束，需要使得数独九宫格的两个对角线上的各 9 个数字也分别满足包含 1-9 且不重复。下图是一个 X 数独的例子。

* * 7 4 * * * * *	5 3 7 4 9 2 8 6 1
* * * 3 * * * * 2	1 9 6 3 7 8 4 5 2
* * 2 5 * * * * 3	4 8 2 5 1 6 7 9 3
* * 5 1 * 9 * 2 *	7 4 5 1 6 9 3 2 8
* * * * * * * * *	8 2 9 7 3 5 1 4 6
* 1 * 8 * 4 5 * *	6 1 3 8 2 4 5 7 9
9 * * * * 1 6 * *	9 7 4 2 8 1 6 3 5
3 * * * * 7 * * *	3 6 1 9 5 7 2 8 4
* * * * * 3 9 * *	2 5 8 6 4 3 9 1 7

总的约束即为：

1) 每一行、每一列、每一个粗线宫（3*3）内的数字均含 1-9，不重复。
2) 九宫格的两条对角线内的数字也均含 1-9，不重复。

实验任务：

1. 实现一个 CSP 问题的**回溯搜索算法**（backtracking search）来解给定的 **X 数独问题**。
2. 对上述实现的算法进行优化，包括：
   a) 设计一个**启发式**来决定选择变量的顺序，如度启发式。
   b) 利用约束条件来**提前减小搜索空间**，如前向检验方法。
   c) 或者其他一些你认为可行的优化方法。
3. 将优化过的算法与原始的算法结果进行比较分析，分析角度：
   a) 搜索遍历的节点数
   b) 搜索所花的具体时间
   c) …
4. 思考题
   a) X 数独这个问题是否可以通过爬山算法、模拟退火算法或是遗传算法等算法来解决？如果能的话，请给出大致的思路。
   b) 如果使用爬山算法、模拟退火算法或是遗传算法等算法来解决，可能会遇到哪些问

题?

上述问题不对这三个算法的具体实现做要求。

作业要求：

1. 使用 C/C++实现上述算法。
2. 实验输入与输出：
   a) 实验输入给定三个 X 数独题目（sudoku01.txt，sudoku02.txt 与 sudoku03.txt），以 9 行，每一行 9 个数字，用一个空格分隔的方式表示，其中未给定的数用 0 来代替，存储在.txt 文件中。
   b) 实验输出要求与输入文件的格式相同，只是其中的原有的 0 用算法找出来的结果代替。输出文件**与对应的输入文件同名**，文件结构见后文描述。
   c) NOTE：最后评分的时候除验证上交的输出结果外，还会更换输入的题目测试。
   d) 请提交源代码与可执行文件。若使用命令行编译请给出编译命令。若用命令行执行并需要命令行参数请给出执行命令。
3. 实验报告使用 PDF 格式提交，实验报告包含以下几点：
   4. 算法思想，以及你的具体的优化方法。
   5. 实验结果说明与分析。
   6. 要求回答思考题。

实验提交：

1. 提交方式：**bb 系统中提交**
2. **请组织好文件结构**，提交的目录结构树应如下例所示：

```
PBXXXXXXXX__张三_exp1\
    |---digit\
        |---src\
            |---{your_code}
        |---input\
            |---{your_input_file}
        |---output\
            |---{your_output_file}
    |---sudoku\
        |---src\
            |---{your_code}
        |---input\
            |---{your_input_file}
        |---output\
            |---{your_output_file}
    |---report.pdf
```

   将文件夹 **PBXXXXXXXX__张三_exp1** 压缩为 **PBXXXXXXXX__张三_exp1.zip**，将压缩包提交
3. **请务必按时完成实验，不接受逾期提交的实验。**
4. 实验中有任何问题请联系助教，数码问题请优先联系助教李丹，数独问题请优先联系助教李继权。