**深 圳 大 学 实 验 报 告**

|  |
| --- |
| **课程名称 人工智能概述**  **项目名称 实验一：八数码难题**  **学 院 计算机与软件学院**  **专 业 软件工程（腾班）**  **指导教师 高 灿**  **报 告 人 黄亮铭 学号 2022155028**  **实验时间 2023年9月20日至2023年10月15日**  **实验报告提交时间 2023年10月22日** |

**教务处制**

# 一、实验目的与要求

**实验目的：**

1．熟悉状态空间表示法；

2．掌握深度优先、广度优先和等代价无信息搜索算法；

3．掌握启发式函数设计，实现面向实际问题的A\*搜索算法；

**实验要求：**

1. 实验提交文件为实验报告和相关程序代码，以压缩包的形式提交，命名规则为“学号数字+姓名+Task1”，如2021154099张三Task1；

2. 所有素材和参考材料需列明出处，实验报告中的图片和程序代码建议标注个人水印或标识信息：姓名，班级，学号信息；

# 二、实验内容与方法

**实验内容：**

1. 利用至少一种无信息搜索算法实现八数码难题求解（可选多种）；

2. 设计至少一种启发式信息函数，利用A\*搜索实现八数码难题求解（选做题）；

# 三、实验步骤与过程

1）利用无信息搜索算法实现八数码难题求解：

1.BFS实现。

I．思路：

①首先定义BFS类，在类内定义以下成员；

start：存储八数码初始情况

end：存储八数码目标情况

queue：搜索队列

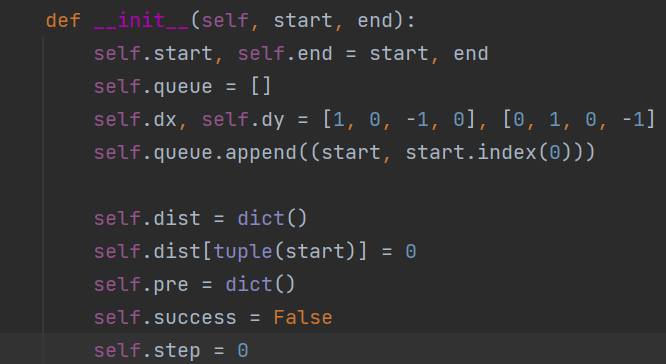
dx、dy：偏移量，用于控制空的位置的移动

dist：存储当前情况与初始情况的距离（需要的步数）

pre：存储当前情况由哪个情况转移而来

success：用于判断是否有解

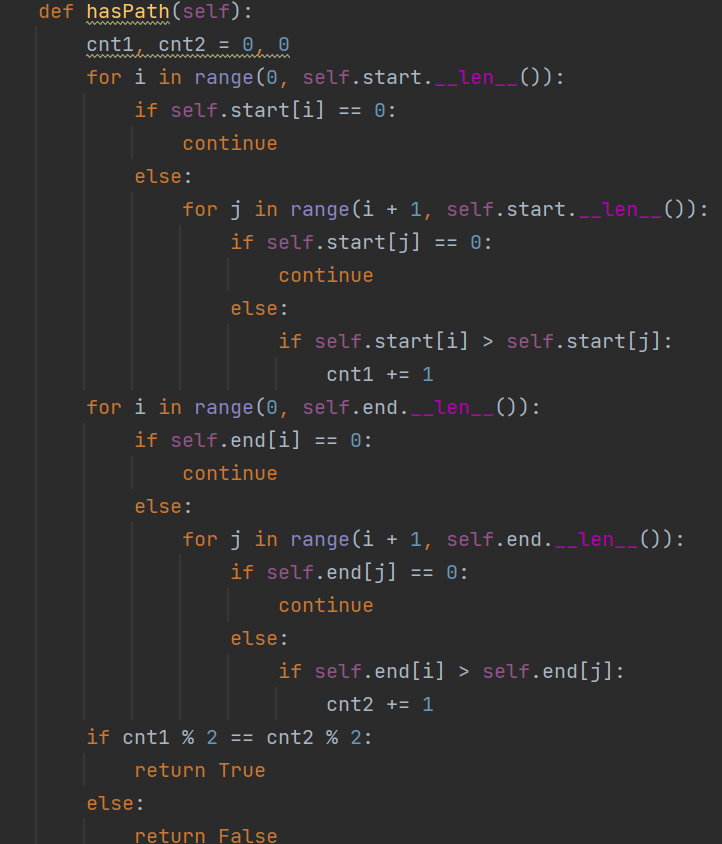
step：存储最短步数



\_\_init\_\_方法

②定义hasPath方法，用于判断是否有解，即判断初始状态和最终状态的逆序对的对数的奇偶性是否相同，相同则有解，否则无解。

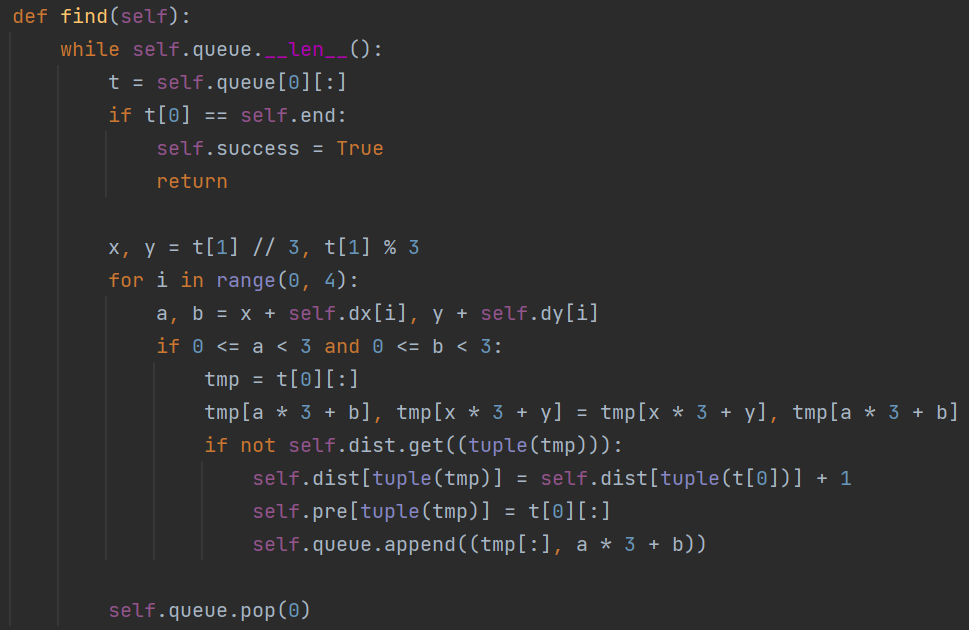
以初始状态为例，先定义cnt1用于存储逆序对对数；考虑到数据较少，我们直接使用嵌套循环查找逆序对，第一重循环i的范围为0~len(list)，第二重循环j的范围为i+1~len(list)，当start[i]或start[j]的数据为0时直接跳过，否则判断start[i]是否大于start[j]，如果大于，则cnt1加1。

 最后利用选择语句返回结果。

hasPath方法

③定义find方法，用于寻找最短路径。

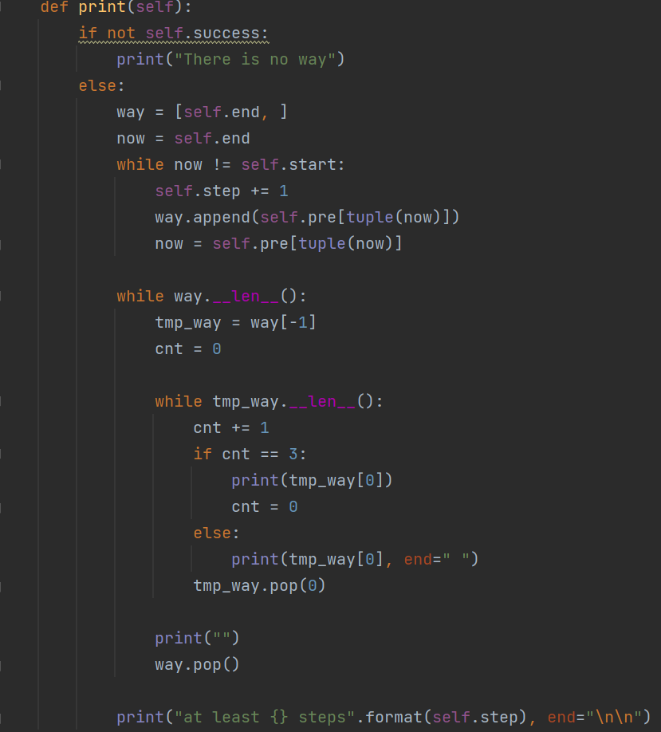
流程如下：首先进入循环（结束条件是queue的长度为0）；然后从队头取出元素，判断是否为最终情况，如是结束循环，否则继续；利用循环将空的位置向四个方向移动，如果合法且没有遍历过则更新信息，入队，最后将队头元素出队；重复上述步骤直到达到最终状态或queue空。



Find方法

④定于print方法，用于输出路径。

具体流程：首先利用success判断是否有解；然后定义临时列表用于存储路径，利用pre存储的信息将路径从最终状态到初始状态依次存到临时列表中；最后逆序输出列表即为初始状态到最终状态的路径。



Print方法

2.DFS实现：

I．思路

①首先定义DFS类，在类内定义以下成员；

start：存储八数码初始情况

end：存储八数码目标情况

dx、dy：偏移量，用于控制空的位置的移动

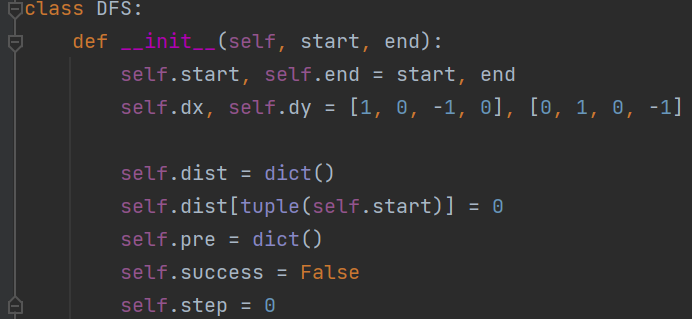
dist：存储当前情况与初始情况的距离（需要的步数）

pre：存储当前情况由哪个情况转移而来

success：用于判断是否有解

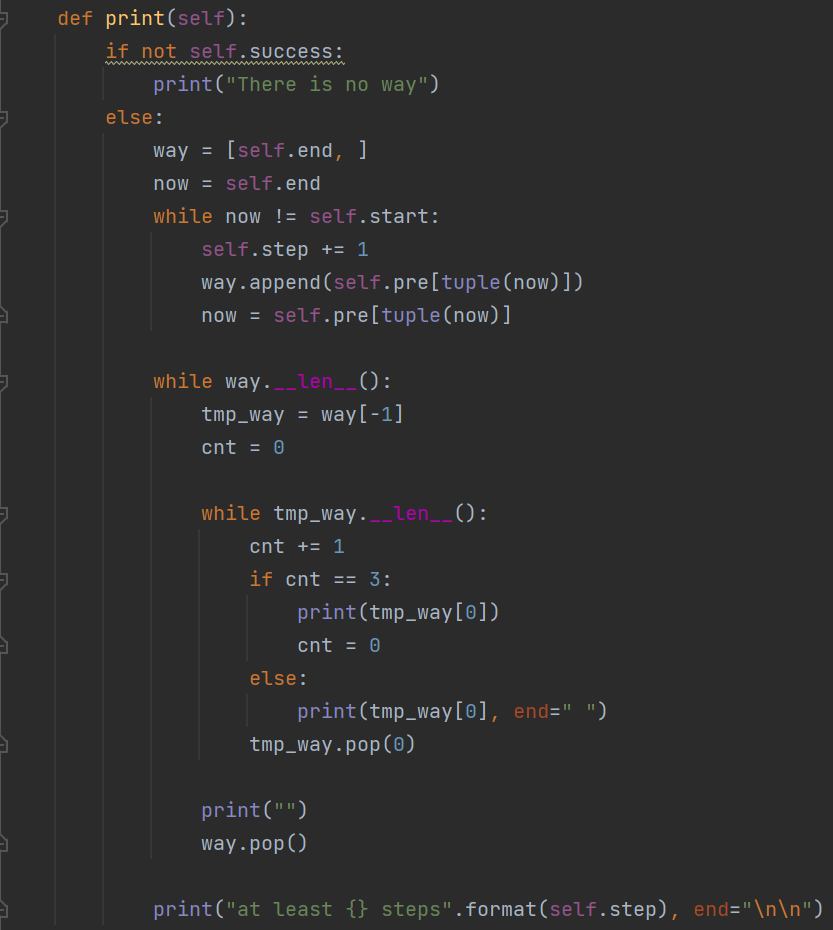
step：存储最短步数

与BFS类相比，缺少了queue队列，因为DFS采用递归的方式实现



\_\_init\_\_方法

②定义hasPath方法，用于判断是否有解，即判断初始状态和最终状态的逆序对的对数的奇偶性是否相同，相同则有解，否则无解。

 与BFS中判断是否有解的方法一模一样。

hasPath方法

③定义find方法，用于寻找最短路径。

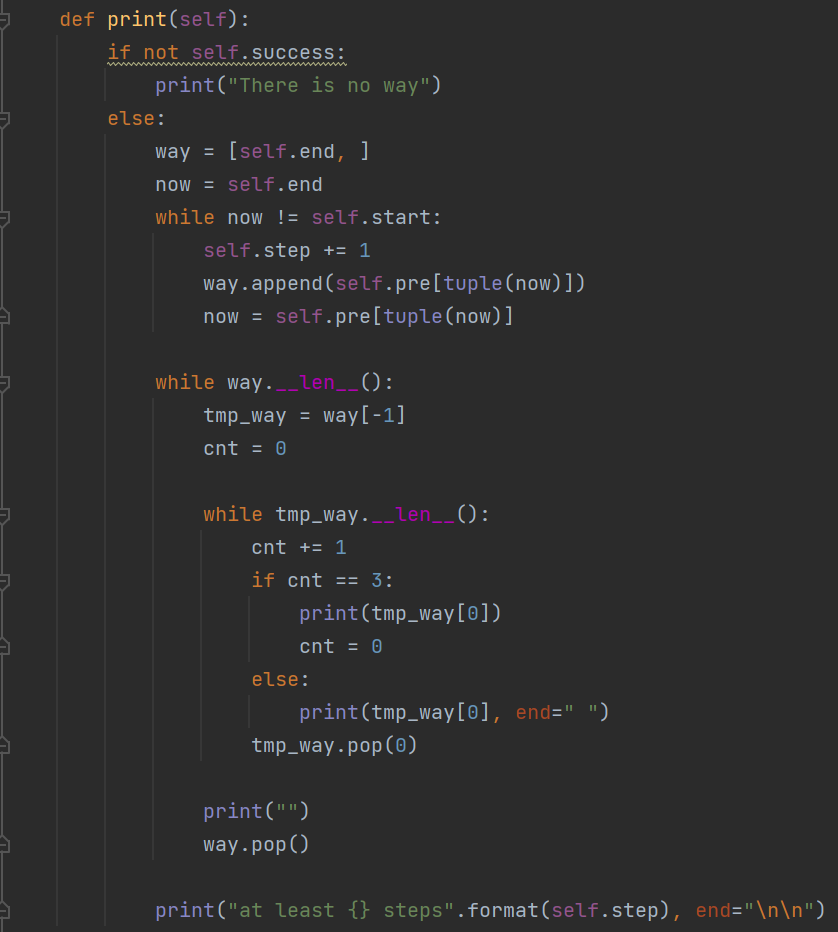
具体流程：首先用st判断状态是否初始化；然后判断当前状态是否为最终状态，同时判断搜索深度是否大于50（大于50就认为没有路径）；利用循环将空的位置向四个方向移动，如果合法且没有遍历过或者当前到该状态的步数小于以前到该状态的步数，则更新信息并沿着当前状态继续往下搜；重复上述步骤，直到搜到最终状态或者搜索完所有状态。（假设有路径，最后到最终状态一定是最短路径，因为不是更短的路径不会更新当前路径）



Find方法

④定义print方法，用于输出路径。

具体流程：首先利用success判断是否有解；然后定义临时列表用于存储路径，利用pre存储的信息将路径从最终状态到初始状态依次存到临时列表中；最后逆序输出列表即为初始状态到最终状态的路径。

 该方法与BFS中的print方法相同。

print方法

2）设计启发式信息函数，利用A\*搜索实现八数码难题求解。

1.AStar实现

I．思路

①首先定义AStar类，在类内定义以下成员；

start：存储八数码初始情况

end：存储八数码目标情况

heap：与BFS类中的queue类似，但是heap每次取出的状态是预估步数最小的状态

dx、dy：偏移量，用于控制空的位置的移动

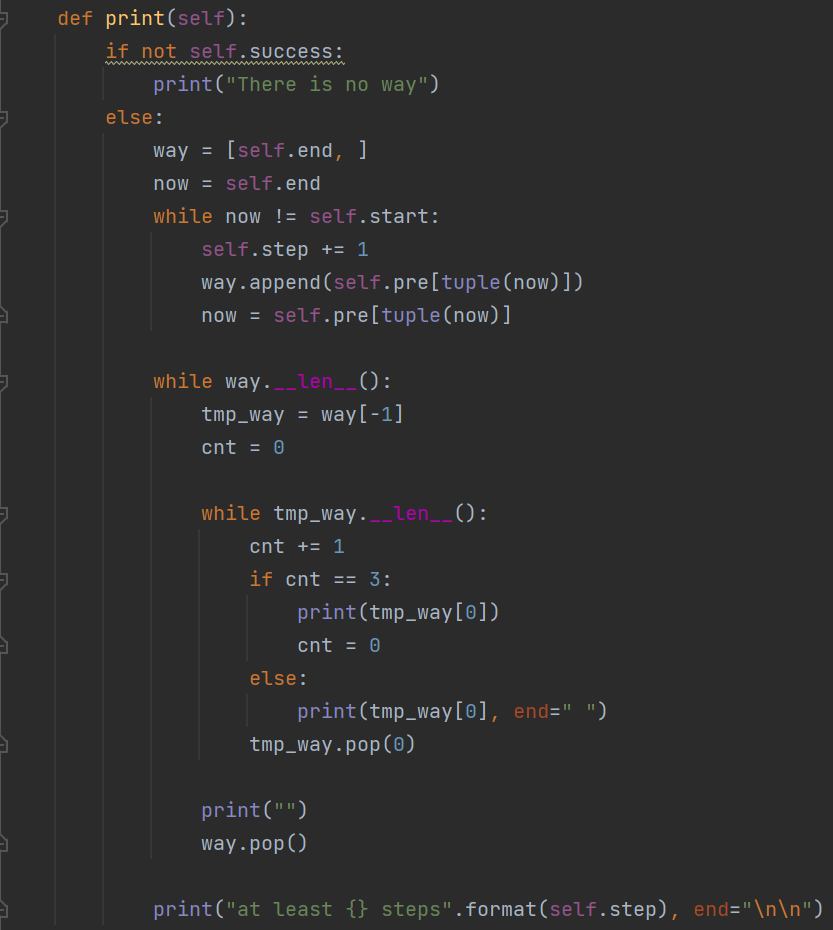
dist：存储当前情况与初始情况的距离（需要的步数）

pre：存储当前情况由哪个情况转移而来

success：用于判断是否有解

step：存储最短步数

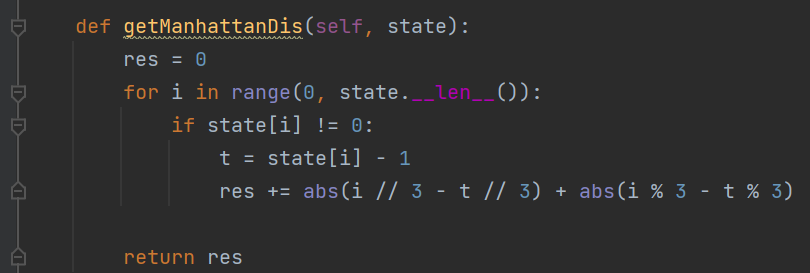
②定义hasPath方法，用于判断是否有解，即判断初始状态和最终状态的逆序对的对数的奇偶性是否相同，相同则有解，否则无解。

 与BFS中判断是否有解的方法相同。

hasPath方法

③定义估价函数getManttanDis，用于计算当前状态预估步数。

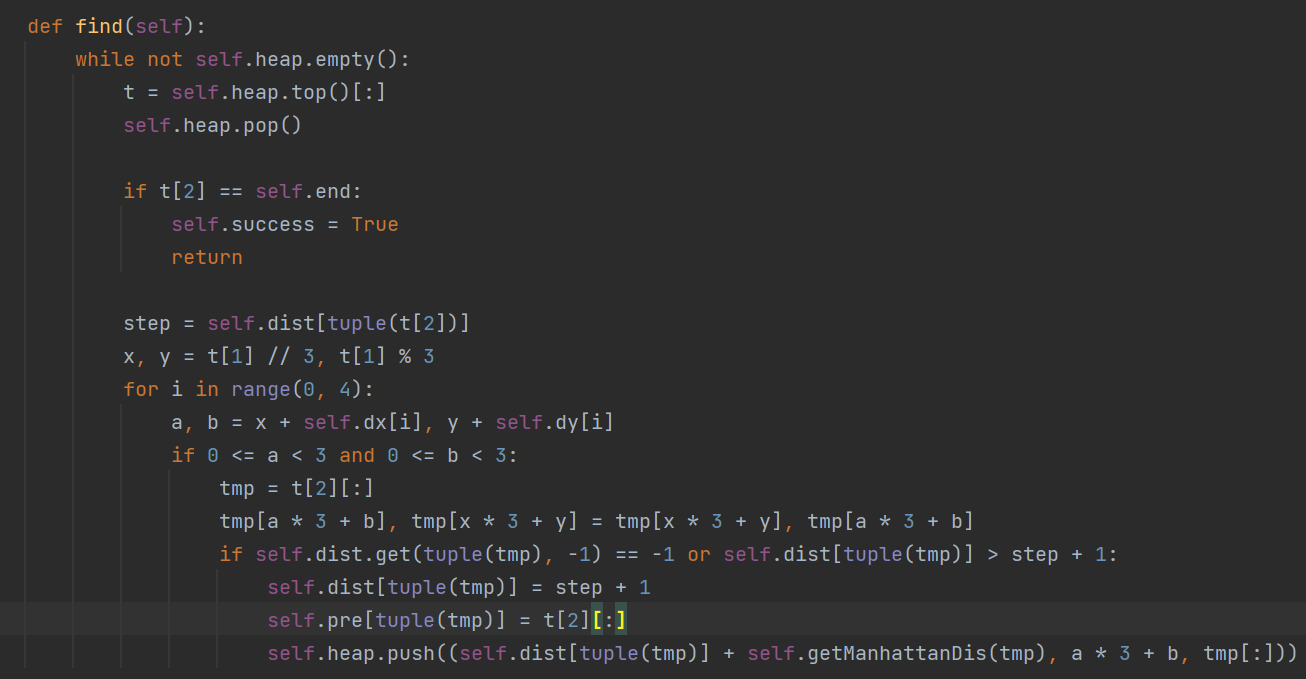
因为一共8个数字和1个空格，数据量较小，所以每次我们都重新计算每个数字当前位置到应到的位置的曼哈顿距离。



估价函数

④定义find方法，用于寻找最短路径。

流程如下：首先进入循环（结束条件是heap为空）；然后从堆顶取出元素，同时将其从堆中删除，判断是否为最终情况，如是结束循环，否则继续；利用循环将空的位置向四个方向移动，如果合法且没有遍历过或当前状态到该状态的距离更短，则更新信息，入堆；重复上述步骤直到达到最终状态或堆空。



find方法

⑤定义print方法，用于输出路径。

具体流程：首先利用success判断是否有解；然后定义临时列表用于存储路径，利用pre存储的信息将路径从最终状态到初始状态依次存到临时列表中；最后逆序输出列表即为初始状态到最终状态的路径。

该方法与BFS中的print方法相同。

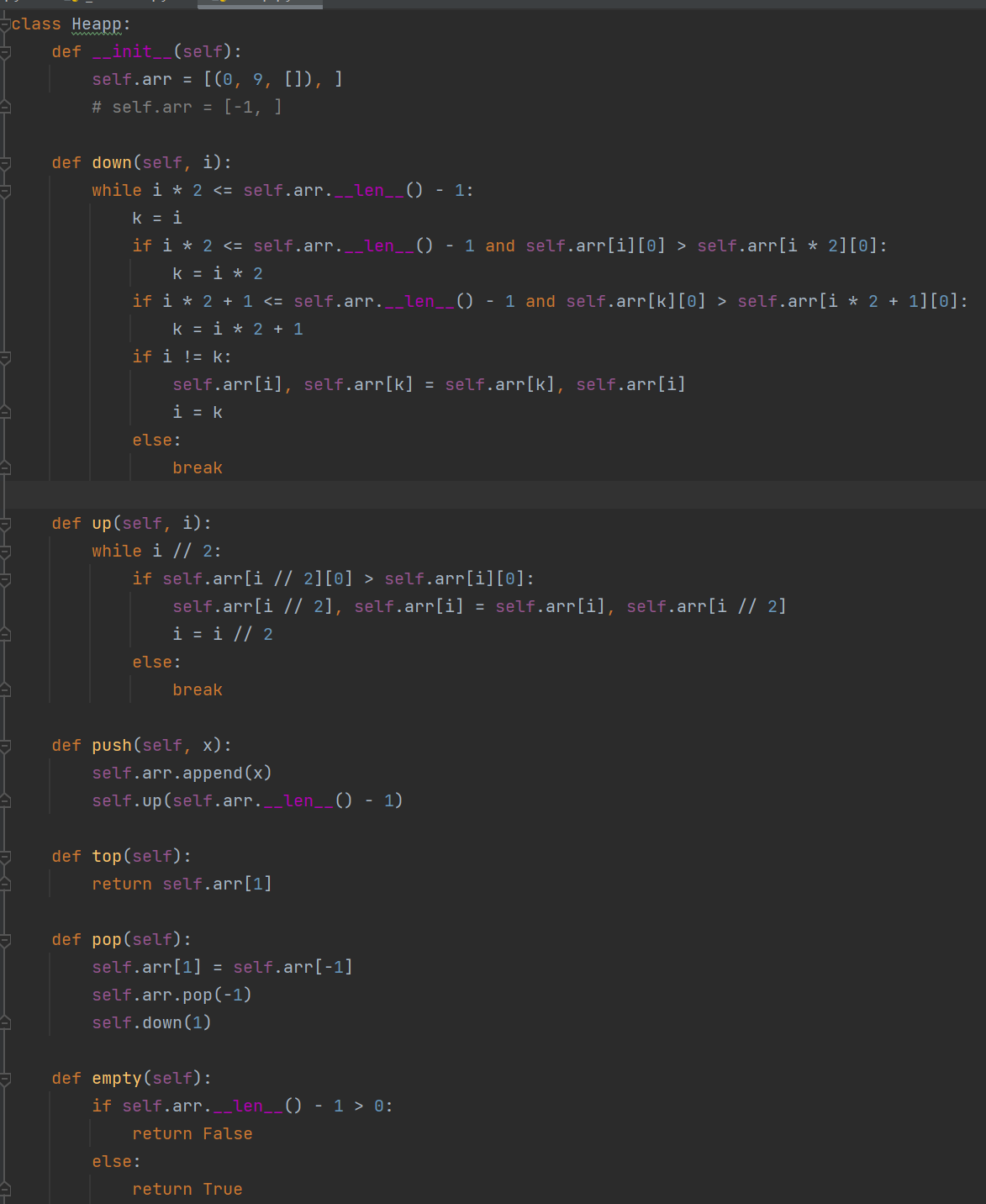


Print方法

II．Heap类的实现

1. 类内定义列表arr存储数据；
2. down方法：将堆顶元素出队时，同时将列表最末的元素插入到堆顶，然后执行该操作，从而维持堆的基本性质。
3. up方法：将元素插入列表时，执行该操作，目的也是为了维持堆的基本性质。
4. push方法：将元素插入堆中。
5. top方法：查询堆顶元素。
6. pop方法：将堆顶元素弹出。
7. empty方法：查询堆是否为空，为空则返回True，否则返回False。

具体实现如下图：

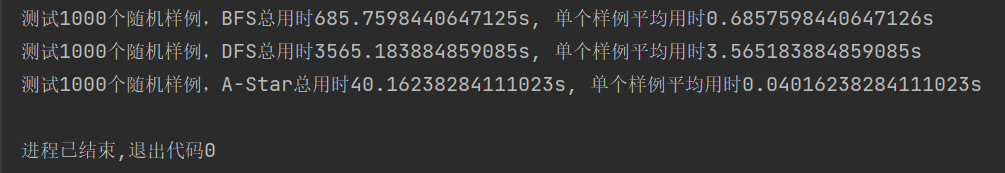


Heap类

3）实验结果：

我采取的测试方法是固定最终状态，然后使用random.shuffle打乱列表，作为初始

状态重复上述1000次，比较三者所花费的时间，从而得出三者的性能差异。



测试结果

由图可知，AStart所用时间最少，BFS次之，DFS所用时间最多；并且AStart和BFS、

BFS和DFS两两之间都有一个数量级的差异。

因此,AStar的性能最优;BFS次之;DFS性能最差,但代码实现较为简单且逻辑清晰。

# 四、实验结论或体会

1. 将BFS、DFS、AStar三个类写完时，发现三个类中许多方法相似，甚至一模一样，应当将三个共有的性质提取出来，作为三者的父类，被三者继承。
2. 我在测试代码的时候,没有改变最终状态,没有改变测试样例的个数,不是十分严谨。
3. 在手写Heap类的down方法的时候，忘记将k的值重新赋值给i，导致堆运行错误，浪费较长debug时间。

# 五、思考题

分析深度优先、广度优化、贪婪、A/A\*各算法的优缺点。

深度优先搜索：

优点：使用递归实现，逻辑较为清晰；占用内存较少，只需要存储当前路径上的节

点（但容易进入死循环导致爆栈）

缺点：可能进入死循环，需要使用特殊标记处理；第一次找到最终状态时不能保证

是最短路径，也因此一般不适用于求解全局最优解的问题。

广度优先搜索：

优点：保证第一次搜索到该状态即为该状态的最优解；不会陷入无限循环中；一般

适用于求解全局最优解问题。

缺点：需要占用大量的内存存储节点；

贪婪：

优点：实现简单；适用于某些问题，如启发式搜索、局部最优解；

缺点：不能保证找到全局最优解，容易找到局部最优解；容易忽略某些重要信息，

导致错误的决策。

A\*搜索：

优点：结合了贪婪搜索和广度优先搜索的优点，使用启发式值来估计最优解；保证找到全局最优解；适用于大多数搜索问题，包括路径规划和游戏人工智能。

缺点：需要选取合适的启发式函数，否则会退化为贪婪算法；需要存储大量节点信息，

占用内存较多。

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：高灿  2023 年 4月 8 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。