A*搜索问题实验报告

徐煜森 PB16110173

本报告包含两部分：A*算法和 IDA*算法。

一. 实验环境

本次实验所有代码均在 Windows 10 系统下使用 Visual Studio 2017 完成。若需要编译源代码可以使用以下两种方法之一：

1. （推荐）使用 Visual Studio（版本最好大于等于 2017）创建与源代码同名的 Visual C++项目 -> Windows 控制台应用程序，将源代码导入；修改项目属性 -> C/C++ -> 在预处理器定义中添加一行参数 _CRT_SECURE_NO_WARNINGS，之后即可完成编译。
2. 在源代码中删除 Visual C++预编译头 `#include "stdafx.h"` 之后在 Windows 系统下使用任意 C++编译器编译源代码。

注意：

1. 因为在源代码中使用了 system 函数，所以可能无法在 Linux 系统中编译。
2. Visual C++默认不允许使用 C 中不安全的库函数，因此需要按照上述步骤添加参数_CRT_SECURE_NO_WARNINGS，来禁止编译时的警告。

与源代码一起提交的可执行文件（.exe 文件）可以在 Windows 系统下运行，输入文件请放置在于.exe 文件相同目录下。

二. 实验输入输出说明

输入文件名默认为”input2.txt”，输入文件应放置于可执行文件同一目录下，输入文件格式为：第一行输入迷宫行数 n 和列数 m，其中$0 \leq n \leq 30; 0 \leq m \leq 60$ n 和 m 由空格分隔。接下来 n 行，每行 m 个由空格分隔的整数，整数为 0 或 1 用于表示迷宫，其中 0 表示可以通行的格子，1 表示不可通行的格子。接下来两行分别是起点和终点的坐标，行坐标在前，列坐标在后，行列坐标由空格分隔，坐标从 0 开始。

例如下图输入表示一个 30*60 的迷宫，搜索起点为（1,0），终点为（28,59）

input2.txt

```
30 60
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0
28 59
```

A* 算法输出文件名为“output_A.txt”，IDA* 算法输出文件名为“output_IDA.txt”，均输出至可执行文件同一目录下。对于输出文件推荐使用 Notepad++打开，若使用其他软件打开可能出现行列不对齐或缺少换行的情况。

上述输入文件的输出结果如下：

input2.txt output_A.txt

```
迷宫大小为30*60
起点为(1, 0), 终点为(28, 59)
花费了0.000229秒
使用曼哈顿距离作为启发式函数
从起点到终点所需步数为:116
走法如下 $是起点, *是路径, #是终点:
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ * * * * * * * * * * * * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * * * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 * * * * * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 * * * * * * * * * * * * * * * * * * * * * * 1 0 0 0 0 * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * * * 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 * * * 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 * * 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 * 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 * * 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 0 * 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * * * * * * * 1 0 0 0 0 * * * * * 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 * * * * * * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 * 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 * * * * * * #
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

在 Notepad++中选中其中一个‘1’，会像上图中出现相同高亮，使输出文件更易读。输出文件中将包含迷宫的基本信息、搜索花费时间、使用的启发式函数、从起点到终点的步数和在迷宫中画出的从起点到终点的路径。其中‘$’表示起点，‘#’表示终点，‘*’表示路径。

注：

1. 如需调整输入迷宫大小的限制，可在源代码中找到下图宏定义，直接修改后编译即可，其中 MAXROW 表示最大行数，MAXCOL 表示最大列数。

```
15      #define MAXROW 30
16      #define MAXCOL 60
```

2. 如需将使用的启发式函数由曼哈顿距离修改为欧氏距离，可在源代码中找到下图宏定义，修改为“false”，之后编译即可。

```
13      #define USEMANHATTAN true
```

3. 如需修改输入输出文件名或目录，可在源代码中找到下图字符串常量，可直接修改为所需目录和文件名。

```
21      const string infile = "input2.txt";
22      const string outfile = "output_A.txt";
```

三. A*算法

1. 算法分析

A*算法维护一个优先队列作为 Open List，优先队列中的元素为节点，以节点的 F 值（G+H）为 Key，最小值优先。另外也需要一个记录节点是否在开启列表中的列表和节点的 G 值列表，前者用于记录每个点是否进入过优先队列，即一个点使用处于待扩展状态，G 值列表记录从起点到每个点的最少步数。

因为一个点可能在之前已被发现并添加进优先队列，还未出队，之后又从另一条路发现到该点的更优的路，此时我们真正需要的是后者，然而优先队列无法修改或删除非队头元素，所以通过维护开启列表来判断一个点是否已经进入优先队列，维护 G 值列表来判断哪个点更优。

对于 A*算法使用优先队列，一定是优先扩展最优的点，G 值列表在之前会被更新为较优值，也同时保证了后续扩展的最优性。所以在 F 值更大的相同点出队时，可以通过查看 G 值列表判断出该点为较差的点，且已被扩展，从而放弃对其的扩展。

A*算法实现：首先将起点加入优先队列 Open List，之后每次从优先队列中取出一个点，若为较优点则计算其四个方向能走的点的 F 值并加入优先队列，若该点已被扩展则舍弃。不断迭代直到优先队列为空。

优化：我的 A*算法的实现没有维护关闭列表，因为 A*搜索总是能先扩展最优的点，关闭列表蕴含的信息已被 G 值列表所包含，可以通过 G 值列表来判断一个点是否已被扩展，因此不维护关闭列表可为空间复杂度做出了常数级别的优化。

2. 启发式函数

本次实验中尝试使用曼哈顿距离和欧氏距离作为启发式函数，在“input2.txt”迷宫上测试两个启发式函数。十次运行测量得出曼哈顿距离平均用时 0.000228s，

欧氏距离平均用时 0.000246s。

由于两种启发式函数都是一致性估计，都可以使 A*算法最优。两者唯一的区别就在启发式函数的计算量上，可以发现欧氏距离的计算量要大于曼哈顿距离，因为欧氏距离中有平方根计算，而平方根计算的代价十分昂贵。

部分实现中可能使用距离的平方从而避免欧氏距离中昂贵的平方根运算，但这样做存在严重的问题！这明显地导致衡量单位的问题。当 A*计算 f(n) = g(n) + h(n)，距离的平方将比 g 的代价大很多，并且你可能会因为启发式函数评估值过高而出错。对于更长的距离，这样做会使 A*退化成 BFS。

因此曼哈顿距离性能最优，与实验结果相符。

3. 复杂度分析

a) 时间复杂度

设 A*搜索扩展节点树的深度为 d（从起点到终点的最短路长度），本次实验中搜索树节点的分支最多为 4，我们需要检查$O(4^d)$个节点才能找出最优解。

故时间复杂度为$O(4^d)$，A*时间复杂度为指数级别。

b) 空间复杂度

设 A*搜索扩展节点树的深度为 d（从起点到终点的最短路长度），本次实验中搜索树节点的分支最多为 4，因为我们需要检查$O(4^d)$个节点才能找出最优解，于是总共需要将$O(4^d)$个节点加入优先队列，另外需要维护的 G 值列表与$O(4^d)$相比为常数。

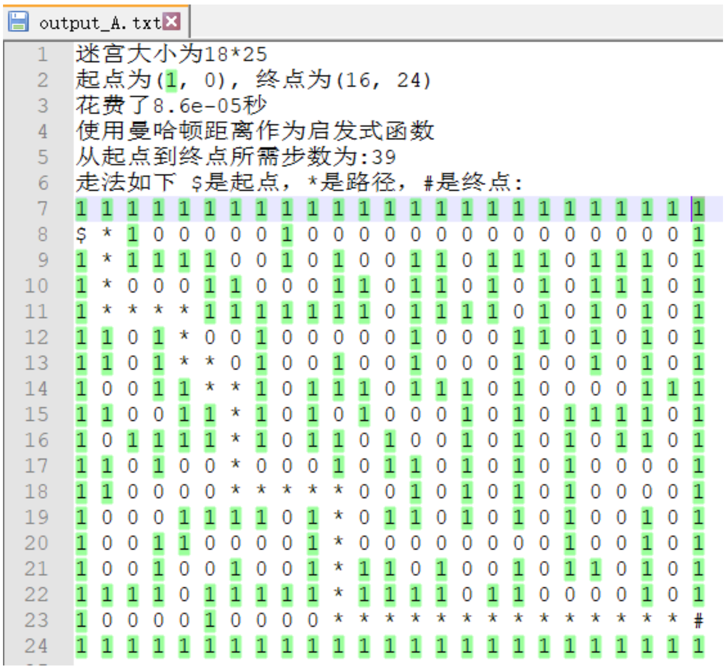
故空间复杂度为$O(4^d)$，A*空间复杂度为指数级别。

4. 测试结果

表一 A*算法测试结果

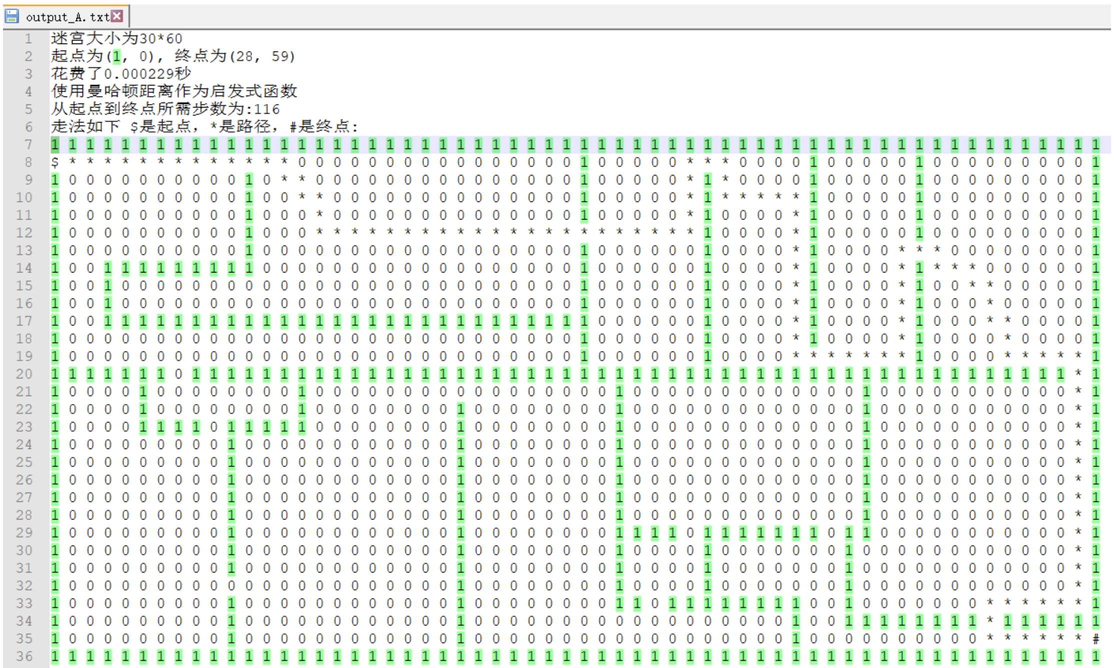
输入文件	从起点到终点步数	平均用时，单位：秒
Input.txt	39	7.9e-05
Input2.txt	116	2.28e-04

其中“input.txt”的路线为



```
output_A.txt
迷宫大小为18*25
起点为(1, 0), 终点为(16, 24)
花费了8.6e-05秒
使用曼哈顿距离作为启发式函数
从起点到终点所需步数为:39
走法如下 $是起点, *是路径, #是终点:
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 * 1 1 1 1 0 0 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1 0 1
1 * 0 0 0 1 1 0 0 0 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1
1 * * * * 1 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 0 1 0 1
1 1 0 1 * 0 0 1 0 0 0 0 0 1 0 0 0 1 1 0 1 0 1 0 1
1 1 0 1 * * 0 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 1
1 0 0 1 1 * * 1 0 1 1 1 0 1 1 1 0 1 0 0 0 0 1 1 1
1 1 0 0 1 1 * 1 0 1 0 1 0 0 0 1 0 1 0 1 1 1 1 0 1
1 0 1 1 1 1 * 1 0 1 1 0 1 0 0 1 0 1 0 1 0 1 1 0 1
1 1 0 1 0 0 * 0 0 0 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1
1 1 0 0 0 0 * * * * * 0 0 1 0 1 0 1 0 1 0 0 0 0 1
1 0 0 0 1 1 1 1 0 1 * 0 1 1 0 1 0 1 0 1 0 0 1 0 1
1 0 0 1 1 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1 0 0 1 0 1
1 0 0 1 0 0 1 0 0 1 * 1 1 0 1 0 0 1 0 1 1 0 1 0 1
1 1 1 1 0 1 1 1 1 1 * 1 1 1 1 0 1 1 0 0 0 0 1 0 1
1 0 0 0 0 1 0 0 0 0 * * * * * * * * * * * * * * #
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

“input2.txt”的路线为



```
output_A.txt
迷宫大小为30*60
起点为(1, 0), 终点为(28, 59)
花费了0.000229秒
使用曼哈顿距离作为启发式函数
从起点到终点所需步数为:116
走法如下 $是起点, *是路径, #是终点:
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ * * * * * * * * * * * * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * * * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 * * * * * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 * * * * * * * * * * * * * * * * * * * * * * 1 0 0 0 0 * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * * * 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 * * * 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 * * 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 * 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 * * 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * 1 0 0 0 0 * 1 0 0 0 0 * 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 * * * * * * * 1 0 0 0 0 * * * * * 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 * * * * * * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 * 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 * * * * * * #
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

四. IDA*算法

1. 算法分析

迭代加深 A*搜索算法以 DFS 为主体，同时使用 A*算法中的 F 值作为阈值。开始时使用初始节点 F 值作为初始阈值 bound，在每次迭代 DFS 中忽略所有 F 值大于阈值 bound 的节点，如果没有找到解则加大 bound，重复进行上述迭代，直到找到一个解，若阈值足够大时仍找不到则认为无解。

在我的实现中 IDA*做了两处优化：

a) 巧妙设置每次迭代后阈值增加的量。当 DFS 遇到 F 值大于阈值 bound 的节点，立即将其 F 值返回，顶层 DFS 将选择最小的 F 值返回，从而使得每次迭代加深 bound 都会有新的点可以被扩展，同时也不会因为增加 bound 过大而盲目搜索更多的节点。

b) 记录重复搜索的节点，用常数的空间换时间。记录起点到每个节点的最短路径长度，若发现更优的则更新并继续扩展，若发现当前路径较差则放弃扩展。

这两处优化的效果显著，对于“input2.txt”中 30*60 的迷宫，其他同学的 IDA*算法可能需要几个小时能得到结果或者根本无法得出结果，我的 IDA*算法依然能够在 0.05 秒内得出结果。

2. 启发式函数

IDA*算法中依然尝试使用曼哈顿距离和欧氏距离作为启发式函数，在“input2.txt”迷宫上测试两个启发式函数。十次运行测量得出曼哈顿距离平均用时 0.002584s，欧氏距离平均用时 0.015022s。同样这两种方法仅有启发式函数计算量上的区别，可以发现当需要大量的启发式函数计算时，曼哈顿距离计算量较小的优势就体现出来了。在 IDA*搜索中使用曼哈顿距离的总耗时是使用欧氏距离的 1/6。此实验结果与 A*算法实验结果相同。

因此曼哈顿距离性能最优，与实验结果相符。

3. 复杂度分析

a) 时间复杂度

设 IDA*搜索扩展节点树的深度为 d（从起点到终点的最短路长度），本次实验中搜索树节点的分支最多为 4，我们需要检查$O(4^d)$个节点才能找出最优解。

故时间复杂度为$O(4^d)$，IDA*时间复杂度为指数级别。

b) 空间复杂度

设 IDA*搜索扩展节点树的深度为 d（从起点到终点的最短路长度），在 IDA*算法的实现中最多只需要同时记录$O(d)$个节点的信息，这是 IDA*相较于 A*算法的优势。

故空间复杂度为$O(d)$，IDA*空间复杂度为线性级别。

4. 测试结果

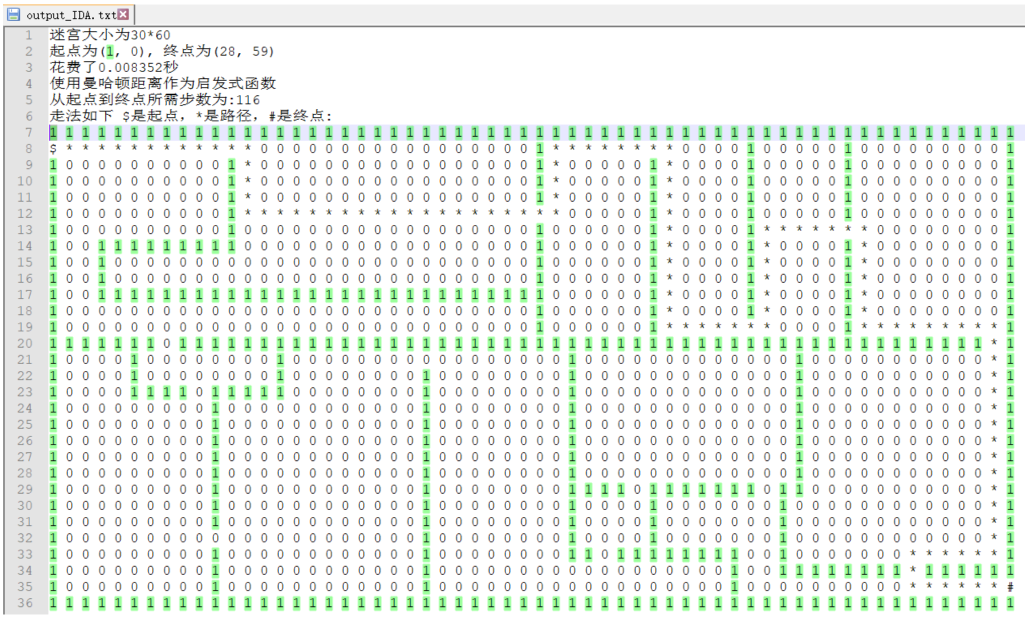
表二 IDA*算法测试结果

输入文件	从起点到终点步数	平均用时，单位：秒
Input.txt	39	3.5e-05
Input2.txt	116	2.584e-03

其中“input.txt”的路线为：

```
迷宫大小为18*25
起点为(1, 0), 终点为(16, 24)
花费了3.5e-05秒
使用曼哈顿距离作为启发式函数
从起点到终点所需步数为:39
走法如下 $是起点, *是路径, #是终点:
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ * 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 * 1 1 1 1 0 0 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1 0 1
1 * 0 0 0 1 1 0 0 0 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1
1 * * * * 1 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 0 1 0 1
1 1 0 1 * 0 0 1 0 0 0 0 0 1 0 0 0 1 1 0 1 0 1 0 1
1 1 0 1 * * 0 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 1
1 0 0 1 1 * * 1 0 1 1 1 0 1 1 1 0 1 0 0 0 0 1 1 1
1 1 0 0 1 1 * 1 0 1 0 1 0 0 0 1 0 1 0 1 1 1 1 0 1
1 0 1 1 1 1 * 1 0 1 1 0 1 0 0 1 0 1 0 1 0 1 1 0 1
1 1 0 1 0 0 * 0 0 0 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1
1 1 0 0 0 0 * * * * * 0 0 1 0 1 0 1 0 1 0 0 0 0 1
1 0 0 0 1 1 1 1 0 1 * 0 1 1 0 1 0 1 0 1 0 0 1 0 1
1 0 0 1 1 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1 0 0 1 0 1
1 0 0 1 0 0 1 0 0 1 * 1 1 0 1 0 0 1 0 1 1 0 1 0 1
1 1 1 1 0 1 1 1 1 1 * 1 1 1 1 0 1 1 0 0 0 0 1 0 1
1 0 0 0 0 1 0 0 0 0 * * * * * * * * * * * * * * #
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

“input2.txt”的路线为：



```
迷宫大小为30*60
起点为(1, 0), 终点为(28, 59)
花费了0.008352秒
使用曼哈顿距离作为启发式函数
从起点到终点所需步数为:116
走法如下 $是起点, *是路径, #是终点:
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ * * * * * * * * * * * * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 * * * * * * * * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 * 0 0 0 0 0 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 * * * * * * * * * * * * * * * * * * * * 0 0 0 0 0 1 * 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * * * * * * * 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 1 * 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 * * * * * * * 0 0 0 0 1 * * * * * * * * * 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 * * * * * * 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 * 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 * * * * * * #
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

五. 实验小结

本次实验可以得出结论：

a) 对于启发式函数，曼哈顿距离比欧氏距离性能更优。
b) 对于搜索算法，A*搜索比 IDA*搜索在时间上更优，但 IDA*搜索在空间上更优，IDA*实现也更加简单。

六. 参考链接

[1]. https://en.wikipedia.org/wiki/A*_search_algorithm
[2]. https://en.wikipedia.org/wiki/Iterative_deepening_depth-first_search