**深 圳 大 学 实 验 报 告**

|  |
| --- |
| **课程名称 人工智能导论**  **项目名称 实验二：对抗搜索游戏**  **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10月18日至2023年11月26日**  **实验报告提交时间 2023年11月25日** |

**教务处制**

# 一、实验目的与要求

**实验目的：**

1．熟悉博弈树及博弈搜索；

2．了解minmax搜索算法和alpha-beta剪枝算法；

3．了解评估函数设计方法，运用博弈搜索解决益智游戏问题；

**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+Task2”，如2020154099张三Task2；

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（三选一）：**

1．改进Reflex智能体；设计Minimax智能体；具体细节参见文件“吃豆人指引.doc”;

2．利用Minimax算法的思想实现井字棋游戏；

3．自选对抗搜索游戏并进行实现。

# 三、实验步骤与过程

**1.Reflex智能体的改进**

**1）改进思路：**根据当前食物分布，判断Pacman进行当前移动是否会使自身离食物更近，以及求出进行当前移动后距离最近食物的距离d，当前移动使得到的d越小则分数越高。在最后返回分数时，特判进行移动后的Pacman和Ghost的距离，如果距离过近则放弃当前移动操作。

**2）具体代码实现：**

I．初始化：获取移动前的食物分布情况，通过遍历得到地图的大小。

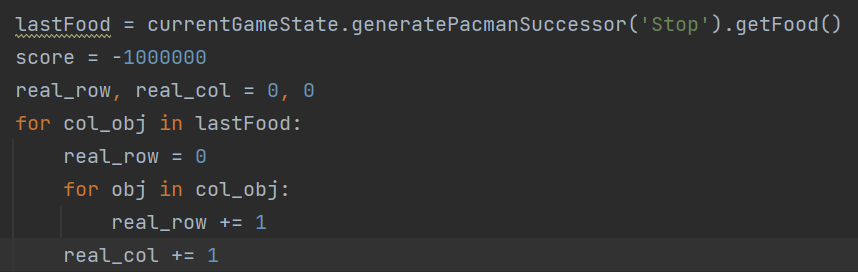


图1

II．利用bfs求出移动后Pacman与Food的最近距离：walls是墙体分布情况；st是判断当前点是否被遍历；dist记录Pacman到当前点最短距离；q是队列，用于更新dist，hh表示队头；ans是分数即返回值；dx和dy是偏移量，通过遍历dx和dy可以实现上下左右搜索。

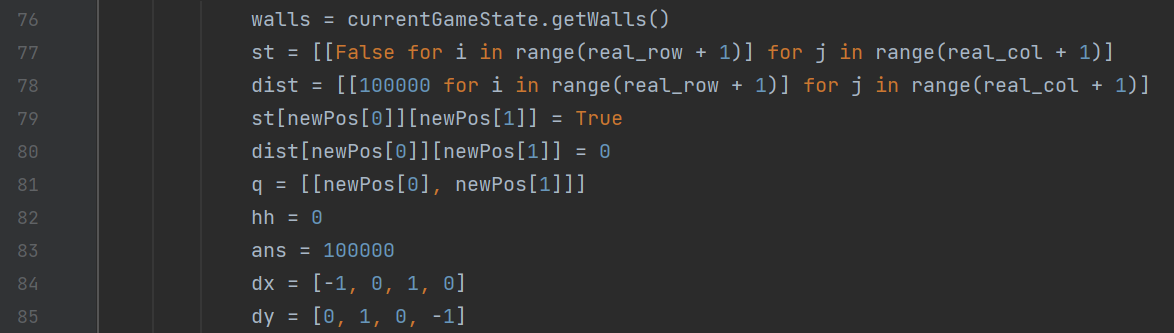


图2

循环：取出队头元素，判断当前点是否为Food，如是则返回，否则继续（bfs的拓扑序保证第一次出队的点的dist是最小值）；然后进行上下左右搜索，剔除非法情况和已经遍历过的点（bfs的拓扑序保证第一次进队的点的dist也是最小值），将符合要求的点的dist更新，st标记为遍历，入队。值得一提的是，返回值为-ans，因为最近的食物的dist最小，取负值后就是最大，从而使分数最大。

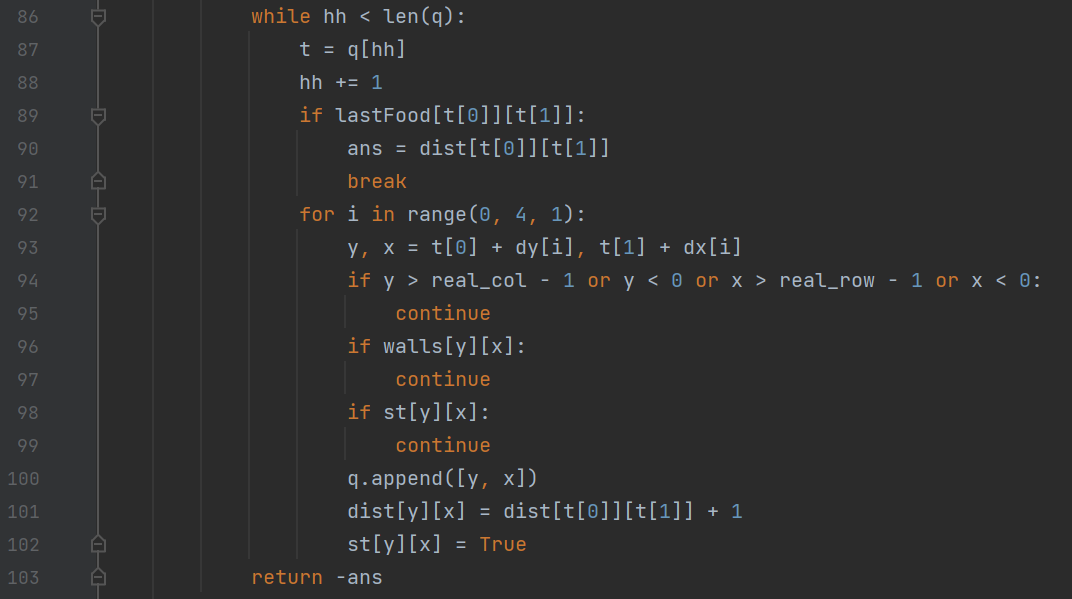


图3

III．判断移动后Pacman和Ghost的距离：使用API获取所有Ghost的位置后，遍历所有Ghost，判断该Ghost和Pacman的距离是否过近（小于或等于1），如是，则本次移动不是最佳，修改返回值为最低，否则正常返回。

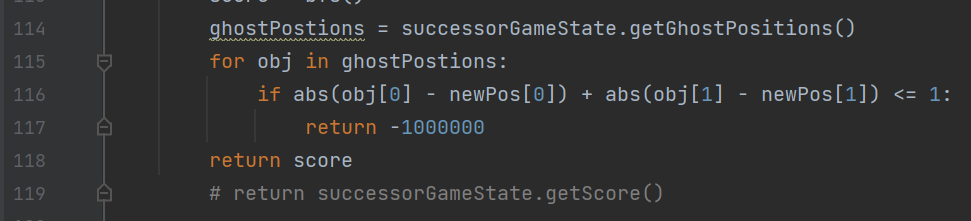


图4

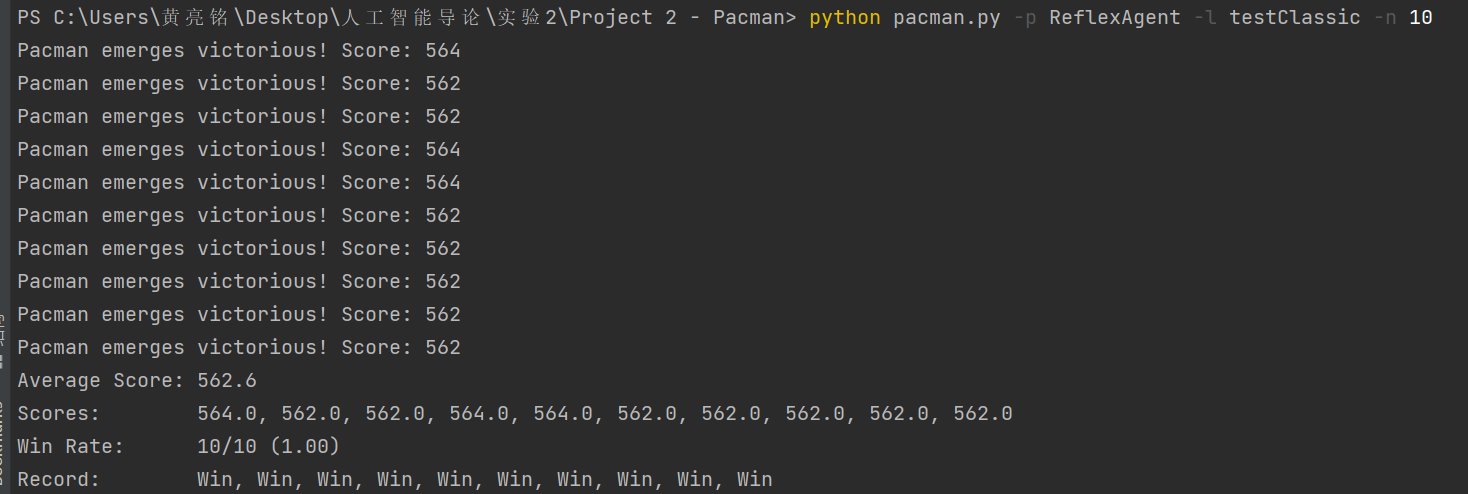
3**）运行结果：**

testClassic布局：运行10次，均能获得胜利。

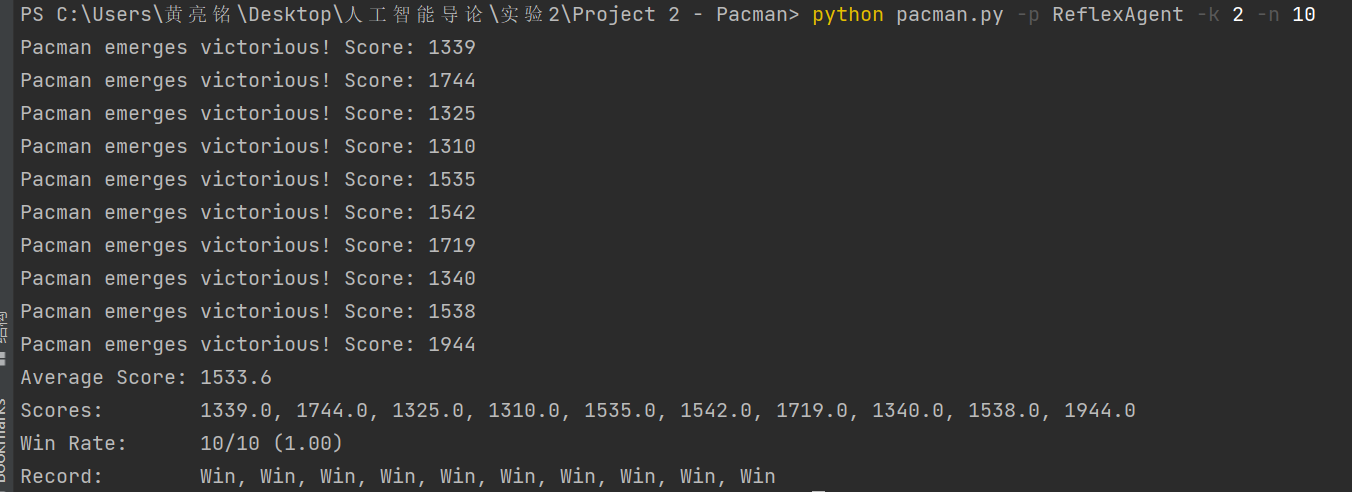
mediumClassic布局：运行10次，均能获得胜利。

mediumClassic布局+“-g DirectionalGhost”：运行10次，9次获得胜利。

openClassic布局：运行10次，均能获得胜利。



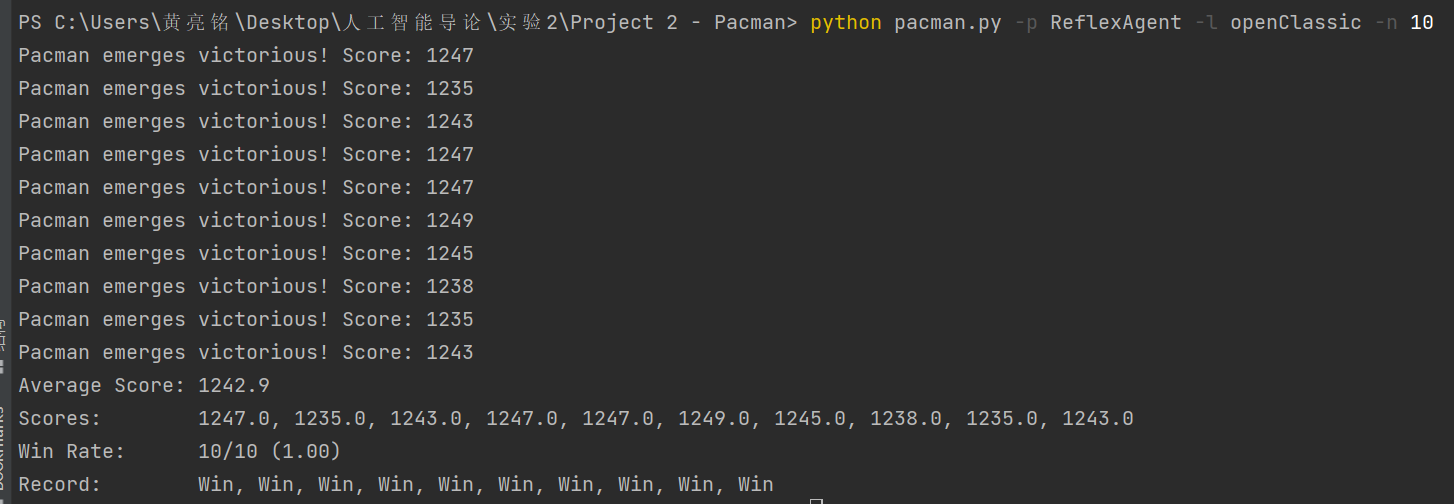
testClassic布局运行结果（图5）



mediumClassic布局运行结果（图6）



mediumClassic布局+“-g DirectionalGhost” 运行结果（图7）



openClassic布局运行结果（图8）

**2.设计minmax智能体：**

**1）设计思路：**①顶层设计在getAction函数中，获取合法动作并执行后进入minmax函数中计算得分，minmax会返回一个分数，通过这个分数，在顶层判断最佳动作，并随机返回最佳动作。②在minmax函数中，先判断当前是什么层以及是否失败、成功或者到达最深深度，随后根据层的属性执行对应的代码，最后返回分数。

**2）具体代码实现：**

I．在getAction函数中，获取合法动作并执行后进入minmax函数中计算得分，minmax会返回一个分数，通过这个分数，在顶层判断最佳动作，并随机返回最佳动作。

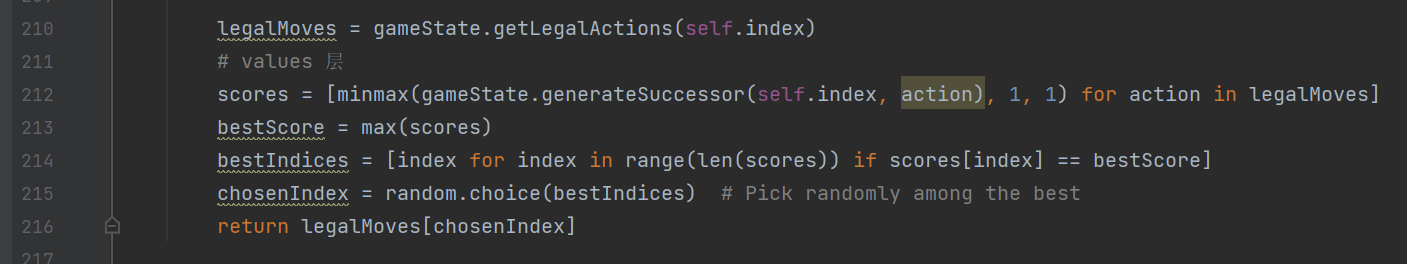


图9

II．在minmax函数中，先获取当前智能体的下标加1的值，如果超出最大智能体下标则将下标设置为0，否则加1。然后判断是否胜利、失败和到达最深深度。

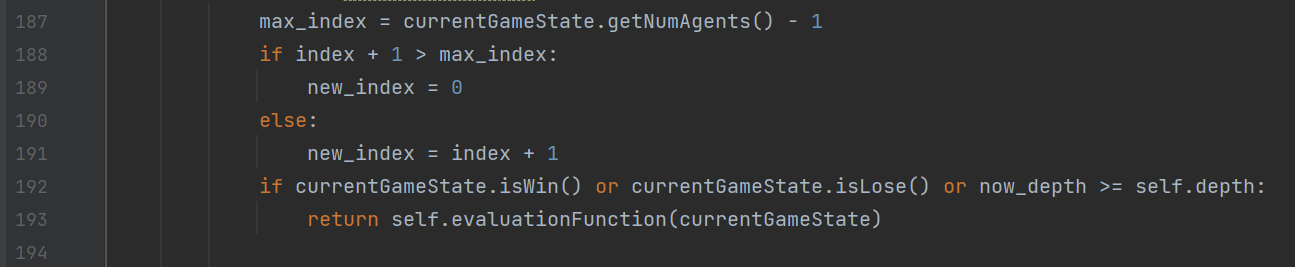


图10

III．获取并遍历合法动作，根据下一层的属性进入选择分支，最后根据当前层的属性返回分数最大值或者最小值。

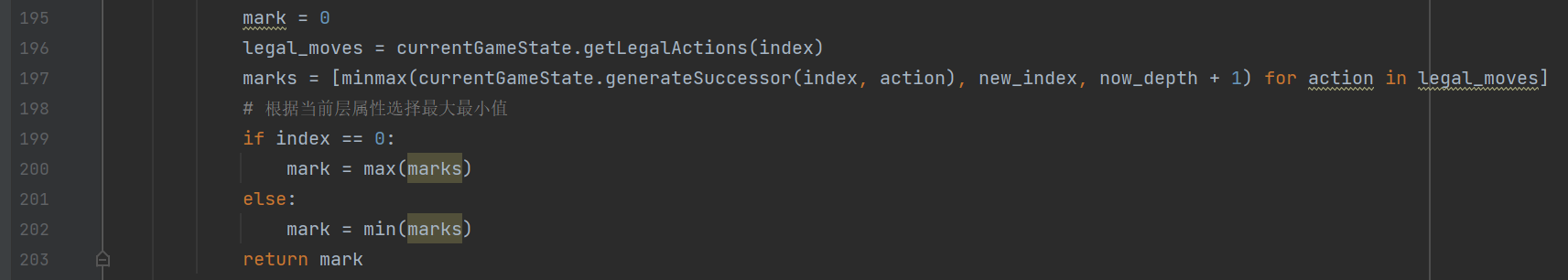
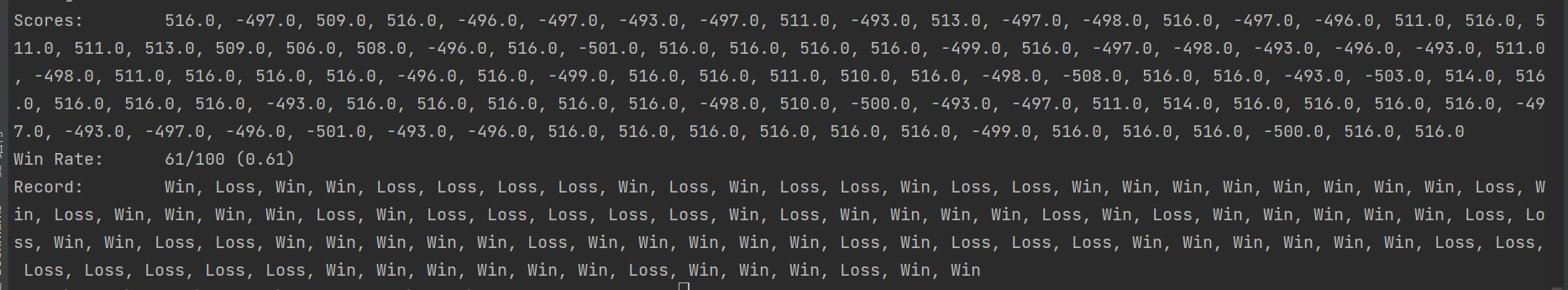


图11

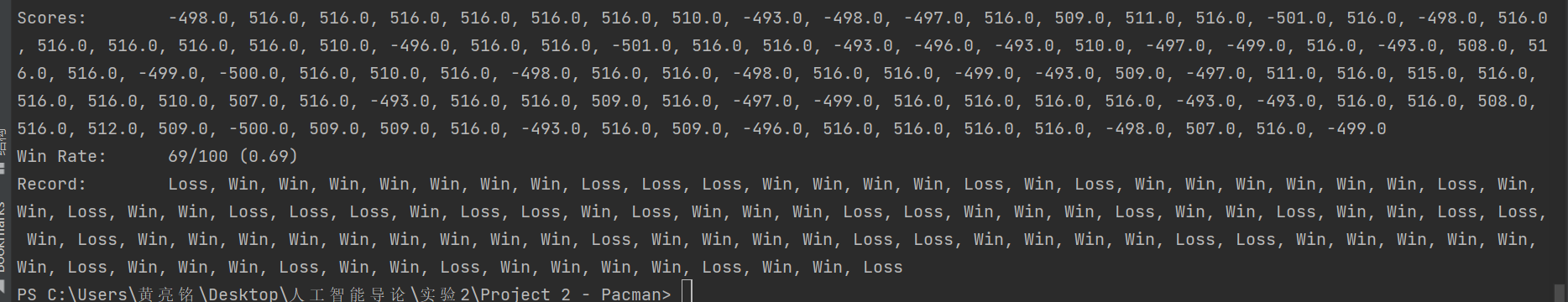
**3）运行结果：**

最坏情况**：**在minmaxClassic布局中运行100次，胜利61次，胜率为61%。



minmaxClassic布局最坏运行结果（图12）

最佳情况：在minmaxClassic布局中运行100次，胜利69次，胜率为69%。



minmaxClassic布局最佳运行结果（图13）

平均情况：多次在minmaxClassic布局中运行100次，平均胜率为65%。

**3.实现alpha-beta剪枝：**

1）设计思路，在minmax智能体的基础上，在minmax函数中添加两个参数alpha和beta，通过判断当前层是否符合alpha<=score<=beta来决定是否继续搜索，从而达到剪枝的效果，加快运行速度。

2）具体代码实现：

I．在getAction中，增加了INF表示无穷，alpha和beta初始化为-INF和INF，其他与minmax智能体中的保持一致。

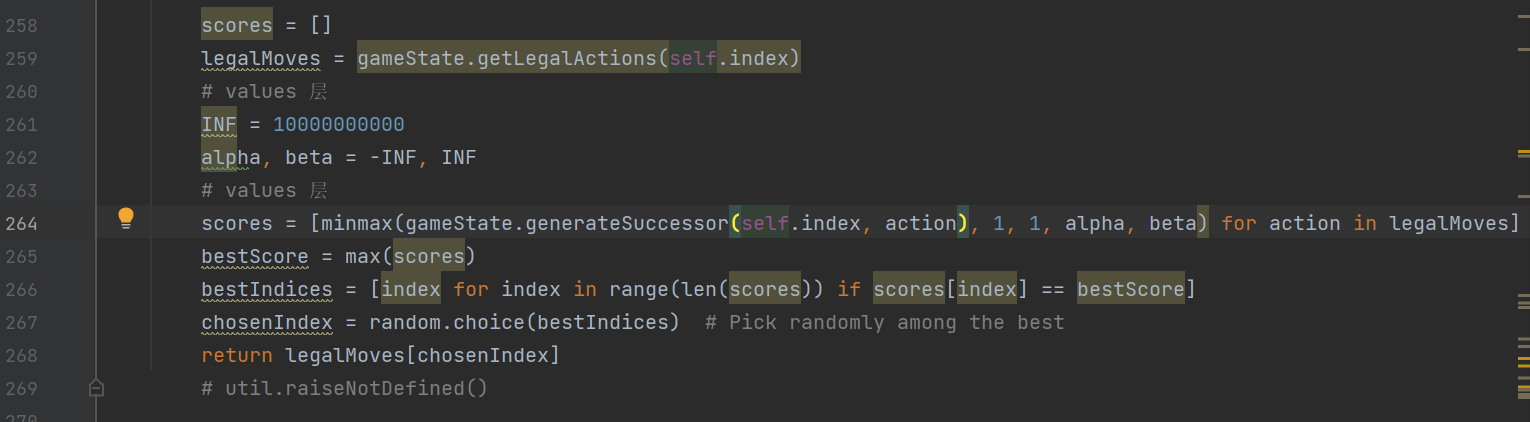


图14

II．在minmax函数中，先获取当前智能体的下标加1的值，如果超出最大智能体下标则将下标设置为0，否则加1。然后判断是否胜利、失败和到达最深深度。（与minmax智能体保持一致）

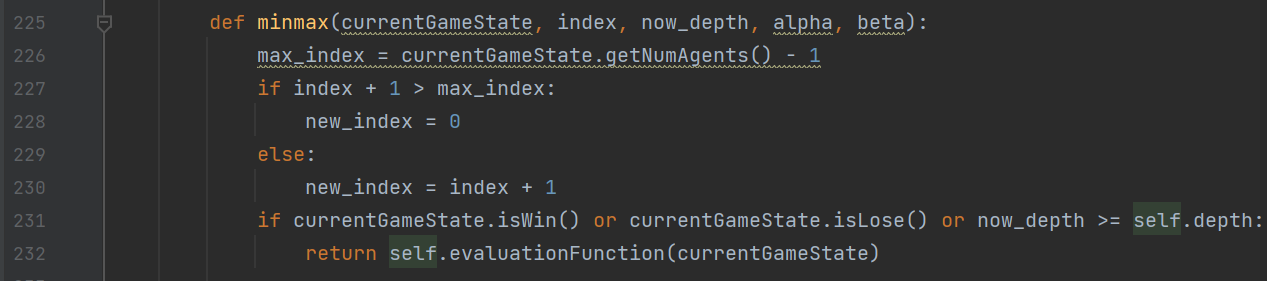


图15

III．根据当前层的属性和返回的分数缩小区间[alpha, beta]，最大层更新最大下界alpha，最小层更新最小上界beta。如果出现beta>alpha的情况，当前层不成立，无需继续搜索，将分数直接返回给上一层，即剪枝。

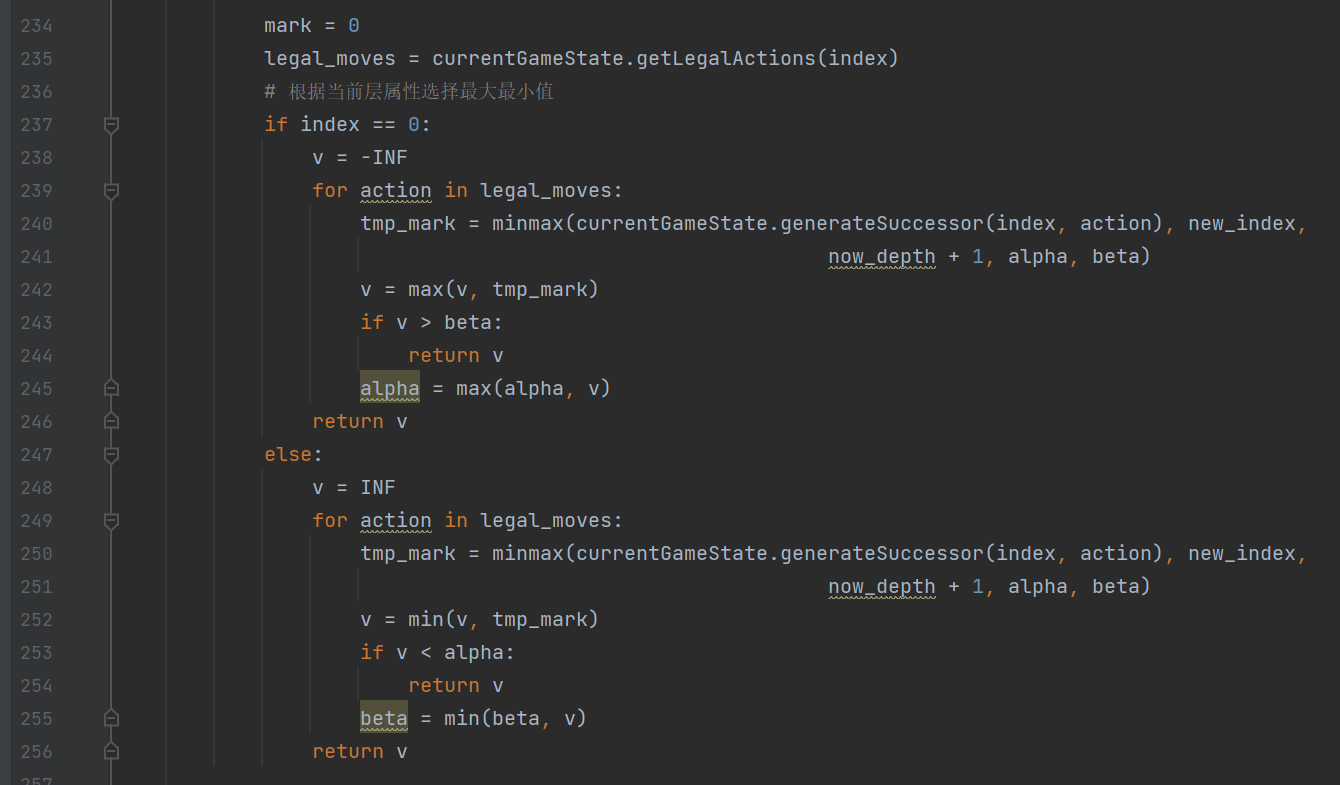
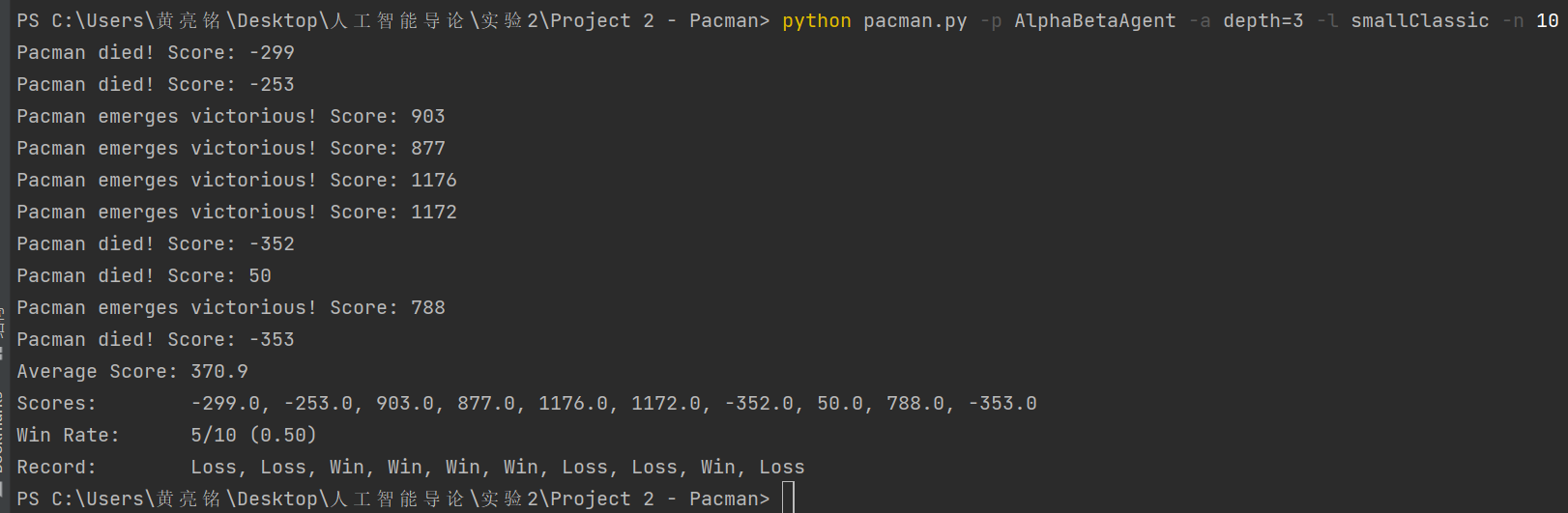


图16

3）运行结果：

在smallClassic中的效果并不理想，10次一组，多组测试最佳胜率仅有50%，平均胜率35%。只有在Ghost追着Pacman的时候，Pacman才会跑。否则，Pacman吃完部分Food后就会在附近来回移动，而不是去其他地方吃新的Food。



smallClassic布局的最佳运行结果（图17）

# 四、实验结论或体会

1. 在改进Reflex智能体中，通过设计每种合法动作的分数，有效阻止Pacman为了吃Food而被Ghost吃掉这种情况；
2. 在改进Reflex智能体中，设计了偏移量dx、dy实现上下左右搜索，使代码更加简洁，且在大量搜索的情况下，可以减小小部分运行时间；
3. 在设计minmax智能体中，通过智能体的下标判断当前层以及下一层的属性，无需编写min-value和max-value两个函数，减少了代码量；
4. 通过本次实验，我熟悉了博弈树和博弈搜搜，学会运用博弈搜索解决益智游戏问题；
5. 通过本次实验，我熟悉了bfs的实现；
6. 通过本次实验，我了解minmax搜索算法和alpha-beta剪枝算法的实现。

# 五、思考题

设计的AI博弈程序一般的优化方法有哪些？

1. 搜索算法优化

剪枝技术：Alpha-Beta剪枝、Negamax剪枝等可以有效减少搜索树的规模，提高搜索效率。

迭代深化：允许算法在有限时间内进行多次深度有限的搜索，逐步提高搜索深度，直到时间用尽。

-局部搜索：对于大规模博弈，使用局部搜索算法如蒙特卡洛树搜索（Monte Carlo Tree Search，MCTS）等。

2. 评估函数的优化：

特征工程：为评估函数选择和提取合适的特征，以更好地反映当前博弈局势。

机器学习：使用机器学习算法（如神经网络）训练评估函数，使其能够从经验中学习。

3. 学习算法的优化：

强化学习：使用强化学习方法进行训练，例如Q-learning、Deep Q Network（DQN）等。

对抗训练：让不同的AI程序相互对抗，从中学习优化策略。

4. 博弈树的优化：

状态压缩：对于状态空间较大的博弈，使用合适的状态压缩方法，减少内存占用和提高搜索速度。

信息剪枝：基于博弈规则的信息，对决策树进行剪枝，减少搜索空间。

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：高灿  2023 年 12 月 3 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。