《2048 游戏》大作业要求说明

提交说明

1、源代码。该任务需要在给定框架下，在 Python3 中开发，写出玩家 AI 文件，机智地玩 2048 游戏。可以提交其他文件，但必须通过 PlayerAI_3.py 包含这些文件。
2、实验报告。注意：报告应当描述大作业设计过程。搜索算法部分的报告应当简洁清晰。

任务说明

首先大家可以在网上搜索 2048 游戏，熟悉游戏规则。该游戏中，电脑 AI 负责在空白位置随机放置 2 或者 4，而玩家 AI 负责移动棋子，移动方向包括上下左右四个方向。

为了让同学们专注于算法的细节，本程序提供了一个框架代码以帮助同学，并允许自行测试算法。注意，你只能更改其中 PlayAI_3.py 文件，实现 getMove()算法，而其余部分是只读的：

- **只读： GameManager_3.py** 。加载电脑 AI 和玩家 AI 的驱动程序开始游戏。请参阅下面有关如何执行此程序。
- **只读： Grid_3.py** 。该模块定义了 Grid 对象以及一些操作：move（），getAvailableCells（），insertTile（）和 clone（），可以在代码中使用它们，但它们绝不是最有效的方法。如果想争取更好的表现，可以忽略这些提供的基础操作方法，并在独立的文件中编写你自己的方法。
- **只读： BaseAI_3.py** 。这是 AI 组件的基类。所有 AI 都从该模块继承，并实现 getMove（）函数，该函数将 Grid 对象作为参数并返回一个移动（对于不同的 AI，有不同的"移动"，对于电脑 AI 采用 ComputerAI_3.py 的 getMove()操作，对于玩家 AI 采用 PlayerAI.py 的 getMove()操作）。
- **只读： ComputerAI_3.py** 。继承自 BaseAI_3。该 getMove（）函数返回一个计算机的'移动'，是表示要放置 2 或 4 的地方一个元组（X，Y）。
- **可写： PlayerAI_3.py**。继承自 BaseAI_3。需要实现的 getMove（）函数，返回一个表示玩家动作的数字。特别是，0 代表"向上"，1 代表"向下"，2 代表"向左"，3 代表"向右"。你可以在提交中包含其他文件，但必须通过此文件包含这些文件。给出的玩家 AI 示例是一个随机策略，请修改该策略，以实现好的玩家 AI。
- **只读： BaseDisplayer.py 和 Displayer.py**。这些文件用于显示结果。

为了测试你的代码，输入以下命令执行游戏管理器：

```
$ python3 GameManager.py
```

游戏将显示在终端屏幕上。玩家 AI 允许每个动作**持续 0.2 秒**，即算法要求每一步执行时间低于 0.2 秒。这个过程一直持续到游戏结束。在游戏结束时，会输出最终 2048 游戏中玩家 AI 合成的最大数。

基本要求

1. 有很多策略可以用来设计玩家 AI，但是此次作业要求采用 **minimax 算法**，实施 **alpha-beta 修剪**，使用**启发式功能**。
2. 每一步必须在 0.2s 时间内提供移动策略
3. PlayerAI_3.py 文件名请勿更改。

注：

1. 本作业采用一致的框架代码，助教会首先覆盖目录中所有的只读文件，确保所有学生采用的游戏机制和电脑对手是相同的。
2. 玩家 AI 表现越好，评分越高。助教会运行 python GameManager.py 十次，以 10 次运行中 5 次最好结果的平均值作为大家玩家 AI 的最终得分。
3. 若大家的游戏策略表现没有比随机上下左右移动策略更好，则本次作业可能会得到零分。网上策略大多只能合成到 2048，若能多次合成到 8192 及以上，本次作业将取得高分。
4. 游戏玩法和规则，参考链接 http://gabrielecirulli.github.io/2048/