Tic_Tac_Toe 作业说明

1. 任务

本编程作业要求实现基于MiniMax Search的tic_tac_toe问题解决方案。

2.程序框架

样例程序包含两个文件：**main_tic_tac_toe.py**以及**tic_tac_toe.py**

tic_tac_toe.py:

- 实现了GameJudge类，用于判断当前状态的输赢情况，**这部分代码不需要修改**；
- 实现了MiniMax_Search函数的基本流程，但是**关键步骤的子函数代码 (eg. min_value, max_value, utility函数) 要求自己编写**；

main_tic_tac_toe.py：

- 测试文件，从tic_tac_toe.py中导入GameJudge和Minimax_Search，实现用户和电脑之间的博弈过程。

完成tic_tac_toe.py文件之后，可以运行main_tic_tac_toe.py测试自己写的Minimax Search算法得到的落子是否合理。一般情况下，如果Minimax Search算法实现合理，电脑和用户总是能下成平局。

3. Minimax Search算法实现说明

本作业要求实现Minimax Search算法，并要求有限搜索深度 (depth=3)。实现过程中，为了保证程序细节的一致性，我们事先做出以下约定：

1. 电脑玩家 (用1表示) 使用circle，你 (用-1表示) 使用cross；
2. 电脑玩家是MAX user，你是MIN user，也就是说电脑的落子要使utility最大，而你落子要使utility最小；
3. Minimax Search的搜索深度限定为3 (depth=3足够得到合理的落子)，也就是你落子的时候会往前多算两步；
4. 你先落子；

Minimax Search算法的基本流程为，对于电脑玩家：

1. 找出当前状态下所有可以落子的地方；
2. 采用depth limited minimax search方法估计每一个落子的地方的utility；
3. 在utility最大的地方落子；

4. 正确运行样例

```
-------------------------------------------------------
[X][ ][X]
[X][O][ ]
[O][ ][ ]
Last move was conducted by you
Game going on
-------------------------------------------------------
[X][O][X]
[X][O][ ]
[O][ ][ ]
Last move was conducted by computer
Game going on
Input the row and column index of your move
1, 0 means draw a cross on the row 1, col 0
2,1
-------------------------------------------------------
[X][O][X]
[X][O][ ]
[O][X][ ]
Last move was conducted by you
Game going on
-------------------------------------------------------
[X][O][X]
[X][O][O]
[O][X][ ]
Last move was conducted by computer
Game going on
Input the row and column index of your move
1, 0 means draw a cross on the row 1, col 0
2,2
-------------------------------------------------------
[X][O][X]
[X][O][O]
[O][X][X]
Last move was conducted by you
Draw
```