五子棋评分标准

各部分评分

1. 运行效率（10）
   - 是否能在规定时间内给出结果，即一步6 second内返回结果
2. 对战排名（20）
3. 代码风格（20）
4. 策略实现（50）
   - 实现了几种策略
   - 是否能自主判断策略选择

代码实现补充说明

- 请下载obe上新发布的包并使用该包中的文件用qt运行
- 所有你需要做的，是将原 `xx.cpp` 中的部分代码复制并粘贴到新的 `xx.cpp` 之中，具体操作如下
  1. 将 `solve_xx` 的命名改为 `solve`
  2. 在新 `xx.cpp` 41行起有如下部分 `namespace{.....}` ，将你在初版 `xx.cpp` 的实现复制粘贴到 `namespace{.....}` 花括号的中间，即你写的**所有**代码，包括 `solve` 都在该区域内完成即可
  3. 最后提交，只提交 `xx.cpp` 一份代码，并将其重命名为 `xxxxxxxxxx.cpp` 即 `学号.cpp`
- 以下代码为新版 `xx.cpp`

```
#include<boardwindow.h>

/*
 * You need to complete this function.
 *
 * The first parameter you've got describes a board,
 *
 * You can use "board_size" as a const number. Actually it has been defined as 19.
 *
 * If turn=1, black
 * If turn=0, white
 *
 * you can use "at(x,y)" to know whether there is a stone at (x,y) and what color it is.
 * If it returns -1, it means there is no stone.
 * If it returns -2, it means the position is invalid.
 *
 *
 * After you decide where to put your stone, please use "make_move" function.
 * Use it like this "Game->make_move(x,y)" if you decide to put your stone at (x,y).
 *
 * The coordinates begin from 0.
 * Make sure the coordinate you give is valid, or the application will crash!!!
 *
 * Here is a example for you with a random algorithm.
*/

// Include headers you need here.
#include <cstdlib>
#include <ctime>
#include <windows.h>
#include <sys/stat.h>
#include <io.h>
#include <stdio.h>
//#include <QtTest/QTest>

using namespace std;

/*
 * Your code goes here.
 */
namespace {

void solve(const Board& board, BoardWindow *Game,int turn)
{
    //Write your algorithm here.
    printf("%d ", turn);
    srand(time(0));
    int x=rand()%board_size;
    int y=rand()%board_size;
    while(~board.at(x,y))
    {
        x=rand()%board_size;
        y=rand()%board_size;
    }


    //printf("%d,%d\n", x,y);
    //Remember to recall "make_move" function.
    //Sleep(10);
    Game->make_move(x,y);
}

}

void
#if !defined(USER_COLOR)
play
#elif USER_COLOR == BLACK
play_black
#elif USER_COLOR == WHITE
play_white
#else
#error "Invalid complier option"
#endif
(const Board& board, BoardWindow *Game, int turn)
{
    return solve(board, Game, turn);
}
```