**程序设计基础实训-综合实验报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 小组成员的  学号和姓名 | 13130188022-徐普  13130188039-郭贝贝 | 实验时间 | 2014年8月20日～  2014年9月23日 |
| 实验环境 | Visual Studio 2013 - C#  .Net 4.5.51641 |
| 实验题目 | 题目三 五子棋游戏 | | |
| 需求描述 | 完成有图形界面的，可以人人对弈或人机对战的五子棋游戏。要求提示友好，增添悔棋功能 | | |
| 概要设计  （模块划分，模块间的关系） | 主要分三个部分：   1. 图形界面设计和棋盘绘制； 2. 人人对弈和悔棋； 3. 电脑对于落子权值的计算。   其中，程序在落子时只判断当前方，人或电脑落子都调用同一接口实现；悔棋时甚至可以做到人机换手继续游戏。 | | |
| 详细设计  （自定义数据类型，全程量，函数定义或函数说明） | 【自定义数据类型】  struct PointEx {  public int x;  public int y;  public int state; //落子方  } // 对弈历史中每个元素的结构  struct EvalResultLine {  public int block; // 0 活 1 半活 2 死  public int count; // 联珠数  } // 权值计算中一条线的联珠数和它是否被（半）堵死  struct EvalResult {  public int[] block;  public int count; // 重复数  } // 权值计算中某一连子数的重复数目和每个重复是否被（半）堵死  EvalResult[] ret = new EvalResult[6];  // 比如 ret[3].count=3 代表在“-|\/”这四个方向有三次是三联珠，其中 .block[2]=? 代表了第二个三连珠是否被（半）堵死  【全局变量】  int[,] board; // 棋盘数组，每个元素中0=无 1=黑 2=白  int intStatus; // 游戏状态，0=未开始 1=黑 2=白 3=黑胜 4=白胜  Stack<PointEx> regret = new Stack<PointEx>(); // 历史堆栈（用于悔棋）  Random random = new Random(); // 随机数生成器  Image img; // 图（用于输出棋盘）  Graphics g; // 画布（用于绘图）  【函数定义】  private void DrawBoard(PictureBox pic, bool reset = true)  // 绘制棋盘，第一个参数是棋盘容器，第二个是是否同时重置棋盘  private int PlayBoard(int X, int Y)  // 某方落子，会自动跳转到对方；成功落子返回1  private void AIplay()  // 电脑落子，包括计算权重，取最高权重点中的随机点落子（无法落子则人赢）  private int EvalChess(int x, int y, int side)  // 计算当前棋盘在(x,y)点给side这一方落子后的权重，权重表内置在此函数中  private EvalResult[] EvalLine(int[,] board, int x, int y)  // 计算临时棋盘board在(x,y)点落子后该点能成的联珠数等信息，同时完成了是否能成五子结束游戏的判定任务 | | |
| 调试分析 | VS2013拥有完善而友好的调试帮助，这给了我们很大帮助。  程序曾经在六（七八等）联珠时无法正常判断胜利，fixed。  程序曾经可以在边界线上落子导致数组越界，fixed。  程序曾经在AI判断联珠时越过边界，导致落子在棋盘对角，fixed。  根据AI需要增加对于联珠的死活和是否存在双活等现象的判断，增强了AI的棋力。  可执行文件在 five-in-line/bin/Release/five-in-line.exe | | |
| 测试结果 | 最终版本的程序，在人人对弈和人机对战方面均同时可玩，也拥有了单步悔棋的功能。  程序目前没有发现更多bug，正常地游玩没有任何问题。  能力有限，目前AI棋力还是很弱，很难打赢人类玩家，我们将在以后改善它，使它的人机对战功能真正耐玩！ | | |