

**《信息管理软件开发技术》**

**课程设计报告书**

**姓　　名**

**学　　号**

**学 　院**

**专　　业**

**指导教师**

**年　 月　　日**

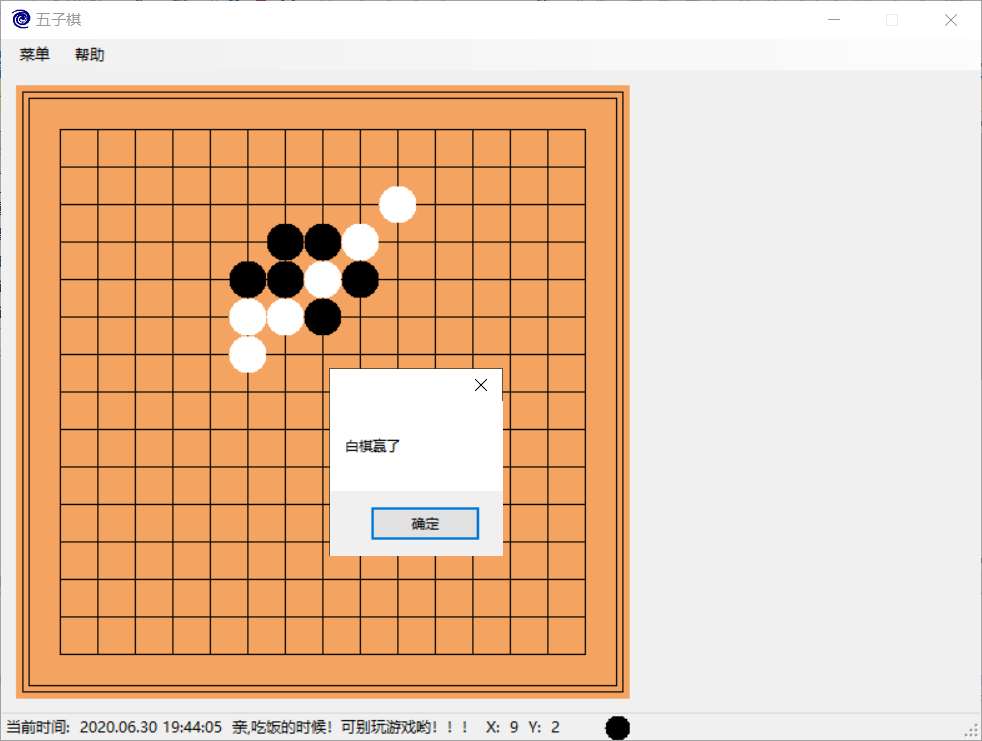
1. 课程设计功能介绍

设计的程序是一个五子棋窗体应用程序。采用C#语言编写。该应用程序包含一个主要窗体。窗体的上方有菜单。主要菜单功能显示如下：

* 1. 重新开始：清空棋盘棋子，重新开始游戏。
  2. 黑棋先下：开始游戏时，黑棋先落子。
  3. 白棋先下：开始游戏时，白棋先落子。
  4. 退一步棋：对于落子错误时，可以退一步，重新落子。
  5. 退出：退出窗体应用程序

程序运行之后，用户可以在棋盘上点击鼠标落子。如果有一方满足五子棋胜利条件，则系统提示用户游戏成功，并告知用户哪一方获得了胜利。

系统的底部有状态栏，主要显示当前时间和一些贴心的小提醒。窗体的主要页面如下所示：



1. 主要代码
   1. 棋盘代码：

/// <summary>

/// 生成初始棋盘位图

/// </summary>

/// <returns>初始棋盘位图</returns>

public Bitmap drawBoard()

{

Bitmap broadPaper = new Bitmap(490, 490);

Graphics myGraphic = Graphics.FromImage(broadPaper);

//黑色画笔

Pen blackPen = new Pen(Color.Black);

//SolidBrush yellowBrush = new SolidBrush(Color.Yellow);

//myGraphic.FillRectangle(yellowBrush, 0, 0, 490, 490);

//画棋盘外框

myGraphic.FillRectangle(new SolidBrush(Color.SandyBrown), 0, 0, 491, 491);

myGraphic.DrawLine(blackPen, 5, 5, 5, 485);

myGraphic.DrawLine(blackPen, 5, 485, 485, 485);

myGraphic.DrawLine(blackPen, 485, 485, 485, 5);

myGraphic.DrawLine(blackPen, 485, 5, 5, 5);

myGraphic.DrawLine(blackPen, 10, 10, 10, 480);

myGraphic.DrawLine(blackPen, 10, 480, 480, 480);

myGraphic.DrawLine(blackPen, 480, 480, 480, 10);

myGraphic.DrawLine(blackPen, 480, 10, 10, 10);

//划格线

int x = 35;

for (int i = 0; i < 15; i++)

{

myGraphic.DrawLine(blackPen, 35, x, 455, x);

myGraphic.DrawLine(blackPen, x, 35, x, 455);

x += 30;

}

return broadPaper;

}

* 1. 棋子代码：

//下一步棋子

public Bitmap PictureNext()

{

Bitmap chessnext = new Bitmap(20, 20);

Graphics g = Graphics.FromImage(chessnext);

SolidBrush b = new SolidBrush(color);

g.FillEllipse(b,0,0,20,20);

return chessnext;

}

//下一步棋的颜色

public void ChangNextColor()

{

if (color == Color.Black)

{

color = Color.White;

}

else

{

color = Color.Black;

}

}

//获取位置

public int getXY(int xy)

{

xy -= 35;

int shang = xy / 30;

int yushu = xy - shang \* 30;

if (yushu < 10)

{

return shang;

}

else if (yushu > 20)

{

return ++shang;

}

return -1;

}

//放棋子

public Bitmap pieceDown(Bitmap chessBoard, int x, int y)

{

Graphics myGraphic = Graphics.FromImage(chessBoard);

//计算坐标

int xCoordinate = x \* 30 + 20;

int yCoordinate = y \* 30 + 20;

//画棋子

if (color == Color.Black)

{

myGraphic.FillEllipse(new SolidBrush(Color.White), xCoordinate, yCoordinate, 30, 30);

}

else

{

myGraphic.FillEllipse(new SolidBrush(Color.Black), xCoordinate, yCoordinate, 30, 30);

}

return chessBoard;

}

* 1. 五子棋规则代码：

public int Check(Point p)

{

if (color == Color.White)

{

//计算各个方向相同的棋子

#region 计算竖线的相邻棋子数

//计算上方有的棋子数

Count = 0;

for (int i = p.Y - 1; i > -1; i--)

{

if (Blist.Contains(new Point(p.X, i))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

//计算下方有的棋子数

for (int i = p.Y + 1; i < 15; i++)

{

if (Blist.Contains(new Point(p.X, i)))

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

#endregion

#region 计算横线相邻的棋子数

Count = 0;

//左边部分

for (int i = p.X - 1; i > -1; i--)

{

if (Blist.Contains(new Point(i, p.Y))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

//右边棋子

for (int i = p.X + 1; i < 15; i++)

{

if (Blist.Contains(new Point(i, p.Y))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

#endregion

#region 计算45度角的棋子数

Count = 0;

//左上边部分

int px, py;

px = p.X;

py = p.Y;

for (int i = p.X - 1; i > -1; i--)

{

py--;

if (Blist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

//右下边棋子

px = p.X;

py = p.Y;

for (int i = p.X + 1; i < 15; i++)

{

py++;

if (Blist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

#endregion

#region 计算135度角的棋子数

Count = 0;

//左下边部分

px = p.X;

py = p.Y;

for (int i = p.X - 1; i > -1; i--)

{

py++;

if (Blist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

//右上边棋子

px = p.X;

py = p.Y;

for (int i = p.X + 1; i < 15; i++)

{

py--;

if (Blist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 1;

}

else

{

break;

}

}

#endregion

}

else

{

//计算各个方向相同的棋子

#region 计算竖线的相邻棋子数

//计算上方有的棋子数

Count = 0;

for (int i = p.Y - 1; i > -1; i--)

{

if (Wlist.Contains(new Point(p.X, i))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

//计算下方有的棋子数

for (int i = p.Y + 1; i < 15; i++)

{

if (Wlist.Contains(new Point(p.X, i)))

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

#endregion

#region 计算横线相邻的棋子数

Count = 0;

//左边部分

for (int i = p.X - 1; i > -1; i--)

{

if (Wlist.Contains(new Point(i, p.Y))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

//右边棋子

for (int i = p.X + 1; i < 15; i++)

{

if (Wlist.Contains(new Point(i, p.Y))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

#endregion

#region 计算45度角的棋子数

Count = 0;

//左上边部分

int px, py;

px = p.X;

py = p.Y;

for (int i = p.X - 1; i > -1; i--)

{

py--;

if (Wlist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

//右下边棋子

px = p.X;

py = p.Y;

for (int i = p.X + 1; i < 15; i++)

{

py++;

if (Wlist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

#endregion

#region 计算135度角的棋子数

Count = 0;

//左下边部分

px = p.X;

py = p.Y;

for (int i = p.X - 1; i > -1; i--)

{

py++;

if (Wlist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

//右上边棋子

px = p.X;

py = p.Y;

for (int i = p.X + 1; i < 15; i++)

{

py--;

if (Wlist.Contains(new Point(i, py))) //黑棋的判断

{

Count++;

if (Count > 3)

return 2;

}

else

{

break;

}

}

#endregion

}

return 0;

}

1. 课程设计总结

通过本次课程设计，了解到一个程序的流程。主要包括软件产品的需求分析、可行性研究、详细设计、界面设计以及代码编写。也加深了对C#这门语言艺术魅力的深入学习研究，尤其是本次的五子棋应用程序，加深了对gdi绘图的认知。相信随着以后的接触，自己也能开发出更好，更符合用户体验操作的软件产品。

当然本次课程设计，也存在一些困难。比如如何构建棋子以及部分清楚棋盘棋子等。经过自己多方的资料查询和自己知识的储备，最终完成了本次的产品。在这里也特别感谢本次课程中对我给予帮助的老师、同学以及学长们，感谢他们让我学到了很多全新的知识，也让自己的能力得到了提升。