**ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

BÁO CÁO ĐỒ ÁN LẬP TRÌNH TRỰC QUAN

**ĐỀ TÀI:**

**LẬP TRÌNH GAME CỜ CARO CHƠI QUA MẠNG LAN**

***GIÁO VIÊN HƯỚNG DẪN:* *NHÓM THỰC HIỆN: TTK***

**Thầy: Huỳnh Tuấn Anh**

**Sinh viên MSSV**

**Hà Huy Khôi 17520647**

**Hoàng Xuân Tùng 17521233**

**Quách Thế Tường 17521243**

***LỚP: IT008.J12.PMCL***

**Tp HCM, 12/2018**

[**MỤC LỤC** 2](#_Toc312840716)

[**LỜI CẢM ƠN** 3](#_Toc312840717)

[**PHẦN I: MỞ ĐẦU** 3](#_Toc312840718)

[1 Lý do chọn đề tài 4](#_Toc312840719)

[2. Đối tượng nghiên cứu 4](#_Toc312840720)

[3. Giới hạn và phạm vi nghiên cứu 4](#_Toc312840721)

[4. Mục đích nghiên cứu 4](#_Toc312840722)

[5. Nhiệm vụ nghiên cứu 4](#_Toc312840723)

[6. Phương pháp nghiên cứu 4](#_Toc312840724)

[7. Ý nghĩa lý luận và thực tiễn của đề tài 5](#_Toc312840725)

[**PHẦN II: NỘI DUNG** 5](#_Toc312840726)

[**Chương I: Giới thiệu trò chơi caro** 5](#_Toc312840735)

[1.1: Giới thiệu sơ lược về trò chơi caro 5](#_Toc312840735)

[1.2: Giới thiệu sơ lược về đề tài 5](#_Toc312840735)

[**Chương II: Giải thuật của chương trình** 5](#_Toc312840736)

[**Chương III: Thiết kế chương trình và cài đặt mạng LAN** 6](#_Toc312840737)

[**3.1: Thiết kế chương trình** 6](#_Toc312840737)

[3.1.1: Giao diện chính 6](#_Toc312840737)

[3.1.2: Giao diện bàn cờ khi đang chơi 7](#_Toc312840737)

[3.1.3: Giao diện bàn cờ khi 1 người chơi thắng 9](#_Toc312840737)

[3.1.4: Giao diện bàn cờ khi người chơi ấn thoát 10](#_Toc312840737)

[3.1.5: Một số hàm sử dụng trong game 11](#_Toc312840737)

[**3.2: Cài đặt mạng LAN** 16](#_Toc312840737)

[3.2.1: Khái niệm địa chỉ và cổng (Address & Port) 16](#_Toc312840737)

[3.2.2: Lớp IPAddress 17](#_Toc312840737)

[3.2.3: IPendPoint 17](#_Toc312840737)

[3.2.4: Giao thức TCP 17](#_Toc312840737)

[3.2.5: TCPListener 17](#_Toc312840737)

[3.2.6: Phân tích từ nhu cầu thực tiễn và yêu cầu 17](#_Toc312840737)

[3.2.7:Mô hình kết nối Client-Sever chế độ có kết nối (TCP) 18](#_Toc312840737)

[**PHẦN III: KẾT LUẬN** 21](#_Toc312840738)

[1. Kết quả đạt được 21](#_Toc312840739)

[2. Hạn chế của đề tài 21](#_Toc312840740)

[3. Hướng phát triển của đề tài 21](#_Toc312840741)

[4. Đề nghị ý kiến. 21](#_Toc312840742)

[**PHẦN IV: PHÂN RÃ CÔNG VIỆC** 22](#_Toc312840738)

[**PHẦN V: TÀI LIỆU THAM KHẢO** 23](#_Toc312840743)

**LỜI CẢM ƠN**

Sau thời gian thực hiện làm đề tài đến nay nhóm đã hoàn thành đề tài của mình. Để có được kết quả như hôm nay, nhóm thực hiện đề tài đã nhận được rất nhiều sự quan tâm, giúp đỡ và chỉ đạo của thầy **Huỳnh Tuấn Anh**. Thầy là người đã tận tình chỉ bảo, chia sẻ cho nhóm những kiến thức bổ ích hướng dẫn động viên trong suốt thời gian qua, tạo mọi điều kiện tốt nhất cho nhóm trong quá trình thực hiện.

Mặc dù nhóm thực hiện đề tài đã rất cố gắng trong quá trình làm đề tài song đồ án của nhóm không thể tránh khỏi những thiếu sót, hạn chế. Nhóm thực hiện đề tài rất mong nhận được sự đóng góp chân thành từ thầy và các bạn để đề tài của nhóm được hoàn thiện hơn.

*Nhóm thực hiện đề tài xin chân thành cảm ơn!*

TP HCM, tháng 12 năm 2018

**Sinh viên thực hiện**

***Hà Huy Khôi***

***Hoàng Xuân Tùng***

***Quách Thế Tường***

# PHẦN I: MỞ ĐẦU

## 1. Lý do chọn đề tài

Hiện nay cờ caro là một trò chơi rất phổ biến trong giới học đường. Từ nhu cầu giải trí đó, đã có rất nhiều hãng đã sản xuất trò chơi caro trên máy tính, giúp cho việc chơi cờ caro được dễ dàng hơn.Trên cùng mục đích muốn đem trò chơi cờ caro đến với mọi người, làm cho việc chơi cờ trở nên dễ dàng hơn, có thể hỗ trợ cho bạn chơi với bạn bè, nhóm em đã chọn đề tài là: “**Lập trình game cờ caro**”.

Qua quá trình tìm hiểu chúng em đã hoàn thành bản báo cáo với nội dung sau:

Phần I: Mở đầu

Phần II: Nội dung

Lập trình game caro

* Chương 1: Giới thiệu trò chơi caro
* Chương 2: Giải thuật của chương trình
* Chương 3: Thiết kế chương trình và các hàm chính của chương trình

Phần III: Kết luận

Phần IV: Phân rã công việc

Phần V: Tài liệu tham khảo

## 2. Đối tượng nghiên cứu

* Chương trình chơi game caro
* Công cụ xây dựng phần mềm: Visual Studio 2017

## 3. Giới hạn và phạm vi nghiên cứu

Lập trình game caro hai người chơi qua mạng LAN.

Kiến thức về mạng máy tính.

## 4. Mục đích nghiên cứu

* Xây dựng được chương trình chơi game caro sử dụng ngôn ngữ C# thông qua phần mềm Visual Studio 2017.

## 5. Nhiệm vụ nghiên cứu

* Tìm hiều các kiến thức về Winform C# để áp dụng vào xây dựng chương trình game caro.

## 6. Phương pháp nghiên cứu

* Tìm hiểu hết tổng hợp kiến thức chung của Lập trình Winform C#, sau đó đi sâu vào những kiến thức cụ thể để xây dựng giao diện ứng dụng cờ caro.

## 7. Ý nghĩa lý luận và thực tiễn của đề tài

* Qua việc tìm hiểu đã giúp em biết thêm kiến thức một số kỹ năng lập trình mới**.**
* Lập trình Winform C# cho ta một cái nhìn tổng quan về thiết kế giao diện đẹp và bóng bảy mà không đòi hỏi phải sử dụng quá nhiều kiến thức cùng lúc mới tạo được.

# PHẦN II: NỘI DUNG

### **Chương I: Giới thiệu trò chơi caro**

#### **Giới thiệu sơ lược về trò chơi caro**

Cờ caro là một loại trò chơi khá phổ biến vì luật chơi đơn giản và không đòi hỏi nhiều về công cụ chơi, bạn chỉ cần một tờ giấy có kẻ ô và bút là đủ.Chính vì vậy trò chơi này lại càng phổ biến trong giới học sinh sinh viên.

Cờ caro là trò chơi đối kháng 2 người. Gồm 2 người chơi, đối thủ này sẽ tìm cách dành chiến thắng trước đối thủ kia trong một số hữu hạn nước đi, mỗi nước đi đuợc tạo ra dựa từ 1 trạng thái bất kỳ của trận đấu. Nếu sau 1 số giới hạn nước đi, nếu chưa ai dành chiến thắng thì xem như hoà. Ngoài ra, thông tin về trận đấu là hoàn toàn biết đuợc đối với cả 2 đối thủ.

Trò chơi có luật chơi hết sức đơn giản như sau: mỗi đối thủ trong mỗi lượt đi của mình sẽ chọn 1 ô trống còn lại trên bàn cờ (kẻ sẵn các ô lưới ) sao cho tạo thành n con liên tiếp để chiến thắng ... Nếu n = 3 thì nó có 1 tên khác là Tic Tac Toe, nhưng ở đây trò chơi kết thúc khi có một người đạt được 5 con liền nhau cùng  trên một  hàng ngang, hàng dọc hoặc đường chéo. Nếu khi đã chơi kín bàn cờ nhưng không có ai đạt được 5 con liền nhau thì trò chơi kết thúc và hoà nhau.

Trò chơi cờ caro có nguồn gốc từ rất lâu, và là trò chơi mang tính quốc tế. Ở các nước khác nhau trò chơi này mang những cái tên khác nhau nhưng đều gắn với luật chơi đơn giản của nó. Ví dụ như: ở Nhật trò chơi có tên Gomoku, Gobang (Go trong tiếng Nhật có nghĩa là 5, Gomoku có nghĩa là 5 điểm), ở Anh trò chơi có tên là Connect 5.

#### **Giới thiệu sơ lược về đề tài**

Xây dưng một chương trình phục vụ chơi cờ caro với các chức năng cơ bản như sau:

* Chức năng Multi Player: Chơi đa người dùng, chơi online: Chương trình hỗ trợ cho 2 người ở trên cùng mạng LAN để có thể chơi.

### **Chương II: Giải thuật của chương trình**

Đây là một chương trình trò chơi mang tính trí tuệ nên phần giải thuật của chương trình là phần quan trọng nhất. Đánh giá một chương trình trò chơi cờ hay hay không là thông qua giải thuật của chương trình.Trong phần này, ta sẽ trình bày chi tiết về giải thuật của chương trình chơi cờ caro.

* Giải thuật có 1 phần đó là: Giải thuật xác định thắng thua cho một ván cờ.

Giải thuật được dựa trên chương trình caro cổ điển với luật chơi là xác định một người thắng khi người đó có 5 con liên tiếp nằm trên cùng một hàng ngang, hàng dọc hoặc đường chéo.

### **Chương III: Thiết kế chương trình và cài đặt mạng LAN**

#### **3.1. Thiết kế chương trình**

##### ***3.1.1. Giao diện chính***

##### 

* Giao diện của bàn cờ bao gồm:

+ Bàn cờ có kích thước 16 x 16 (là một mảng các button)

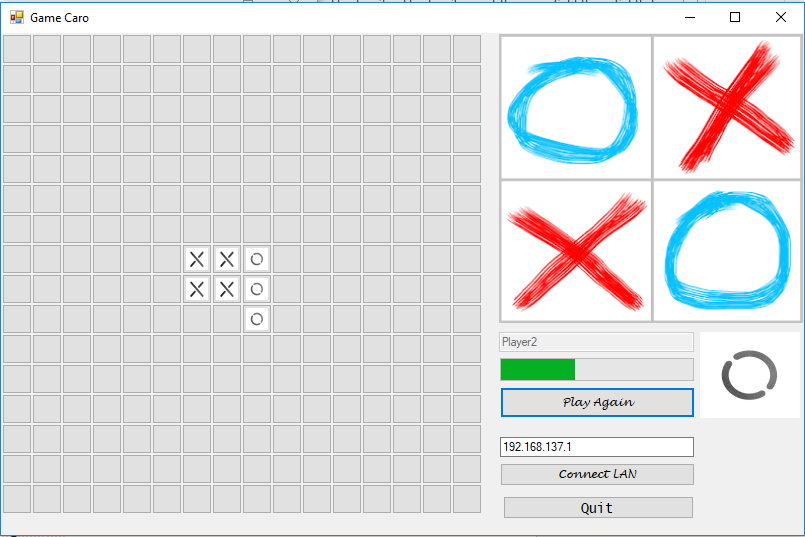
+ Các button: Chơi lại (play again), Connect LAN (kết nối mạng LAN), Quit(Thoát)

+ Các textbox: Player (để khi chuyển lượt đánh thì sẽ hiện tên người chơi đang đánh), IP(hiện lên địa chỉ IP)

+ Picture box: Hiện kí hiệu mà người chơi đang đánh sẽ đánh ra.

+ Thanh progress bar: hiện thời gian đếm ngược giới hạn để người chơi đánh

##### ***3.1.2. Giao diện bàn cờ khi đang chơi***

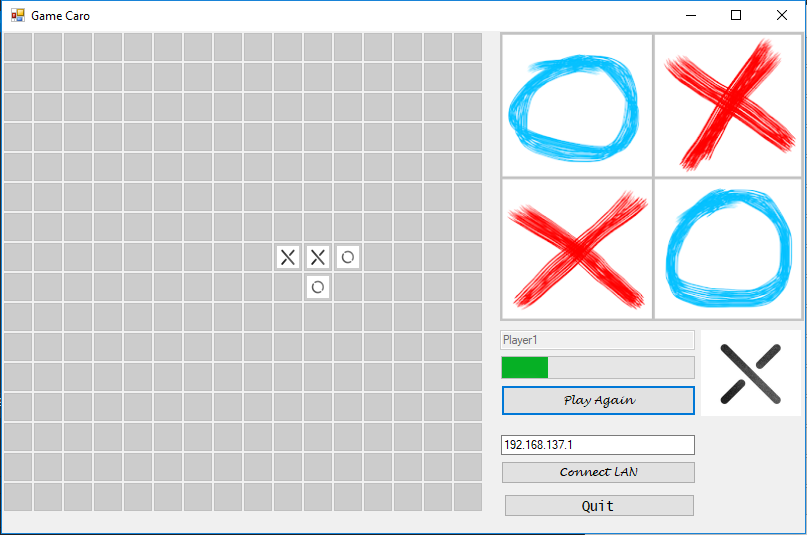


* Khi người chơi đang đánh:

+ Thanh progress bar sẽ chạy để đếm thời gian mà người chơi có thể đánh

+ Bàn cờ sẽ hiện lên để người chơi có thể đánh

+ Kí hiệu trong picture box chính là kí hiệu mà người chơi đánh ra



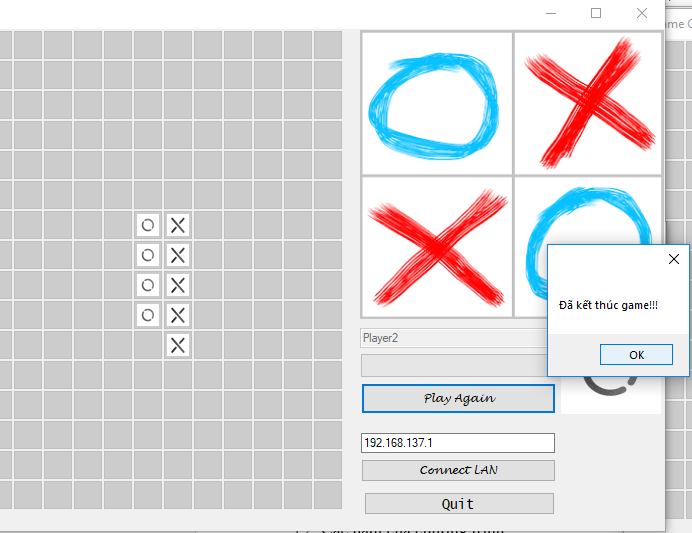
* Khi người chơi đợi người chơi khác đánh:

+ Thanh tên sẽ đổi thành tên người chơi phía bên kia

+ Bàn cờ sẽ bị vô hiệu hóa để người chơi đang đợi không được phép đánh vào

+ Kí hiệu trong picture box sẽ đổi thành kí hiệu của người chơi bên kia

##### ***3.1.3. Giao diện bàn cờ khi 1 người chơi thắng***



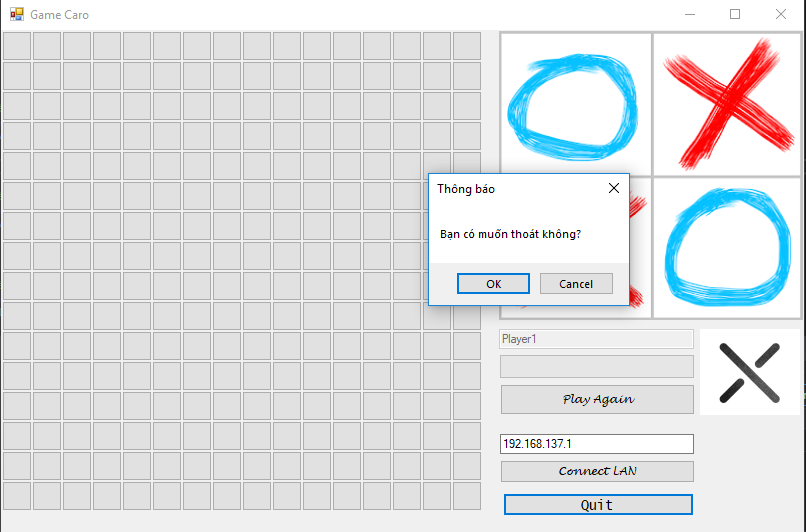
- Khi kết thúc game, hiện lên thông báo “Đã kết thúc game!!!”.

- Bàn cờ đóng lại không cho người chơi đánh lên được nữa.

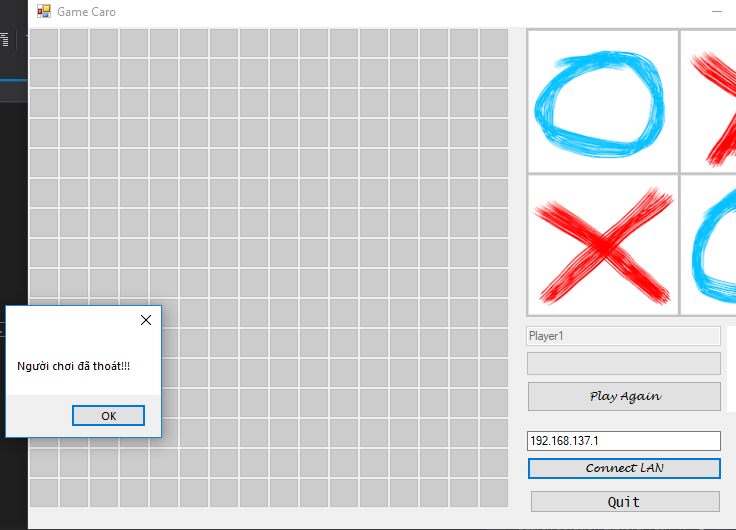
- Thanh progress bar dừng lại.

- Ấn nút “Play Again” để chơi lại.

#### **3.1.4. Giao diện khi người chơi ấn thoát**



* Hiện lên thông báo: “Bạn có muốn thoát không?”.
* Nếu ấn Cancel -> không ảnh hưởng gì cả.
* Nếu ấn Ok -> Chương trình đóng lại, người chơi phía bên kia hiện lên thông báo: “Người chơi đã thoát”.



***3.1.5.Một số hàm sử dụng trong game***

***3.1.5.1.Hàm vẽ bàn cờ***

\* Vì bàn cờ là một mảng các button ta sẽ sử dùng vòng lặp for để vẽ, đồng thời thiết lập các thuộc tính cho nó như: chiều dài, chiều cao, vị trí, sự kiện click cho nó.

public void DrawChessBoard()

{

//Cho người chơi hiện tại = 0

CurrentPlayer = 0;

//Hàm hiện thông tin hiện tại của người đang đánh

ChangePlayer();

//Cho phép người chơi đánh

Play.Enabled = true;

//Tạo một list button

matrix = new List<List<Button>>();

//Button cũ (button trước đó)

Button oldButton = new Button() { Width = 0, Location = new Point(0, 0) };

//Vẽ mảng button (các ô cờ)

for (int i = 0; i < ChessBoard.Chess\_board\_height; i++)

{

//Tạo một list các button

Matrix.Add(new List<Button>());

for (int j = 0; j < ChessBoard.Chess\_board\_width; j++)

{

//Khởi tạo button

Button btn = new Button()

{

//Các thông số cho các button

Width = ChessBoard.Chess\_Width,

Height = ChessBoard.Chess\_Height,

Location = new Point(oldButton.Location.X + oldButton.Width, oldButton.Location.Y),

BackgroundImageLayout = ImageLayout.Stretch,//giúp kí hiệu nằm trong không gian của ô cờ

Tag = i.ToString()

};

btn.Click += btn\_Click;//Sự kiện click cho button

Play.Controls.Add(btn);//ADD các button vào panel

matrix[i].Add(btn);//ADD các button vào matrix

oldButton = btn;//Gán oldbutton = button

}

oldButton.Location = new Point(0, oldButton.Location.Y + ChessBoard.Chess\_Height);//Set lại vị trí mới của oldbutton khi hết hàng(xuống dòng)

//Set lại kích thước của oldbutton

oldButton.Width = 0;

oldButton.Height = 0;

}

}

***3.1.5.2.Hàm sự kiện Click của các button (ô cờ)***

\* Sau khi thiết lập sự kiện Click của mảng button đã được tạo ở phía trên, bây giờ ta sẽ thiết lập khi Click vào button sẽ có những gì:

private void btn\_Click(object sender, EventArgs e)

{

Button btn = sender as Button;

//Nếu button đã được người chơi đánh thì sẽ không đánh chèn vô được nữa

if (btn.BackgroundImage != null)

return;

//Hàm hiện kí hiệu hiện tại của người đánh

Mark(btn);

currentPlayer = CurrentPlayer == 1 ? 0 : 1; //Nếu người chơi = 1 thì chuyển sang 0, nếu 0 thì chuyển sang 1

ChangePlayer();//Hàm hiện thông tin hiện tại của người đang đánh

if (playermarked != null)

playermarked(this, new buttonclickevent(GetChessPoint(btn)));

//Nếu kết thúc game

if(isEndGame(btn))

{

EndGame();

}

}

***3.1.5.3.Hàm xử lý thắng thua (thắng đường dọc, thắng đường ngang, thắng đường chéo chính, thắng đường chéo phụ)***

\* Vì để chiến thắng trong game caro thì cần 5 điểm nằm trên 1 đường thẳng, dựa trên điều đó, ta có thể làm ra thuật toán xử lý thắng thua

\* Hàm thắng dọc

//Hàm thắng dọc

private bool isEndVertical(Button btn)

{

Point point = GetChessPoint(btn);

int count top = 0;

for (int i = point.Y; i >= 0; i--)

{

if (matrix[i][point.X].BackgroundImage == btn.BackgroundImage)

{

countTop++;

}

else

break;

}

int countbottom = 0;

for (int i = point.Y + 1; i < ChessBoard.Chess\_board\_height; i++)

{

if (matrix[i][point.X].BackgroundImage == btn.BackgroundImage)

{

countbottom++;

}

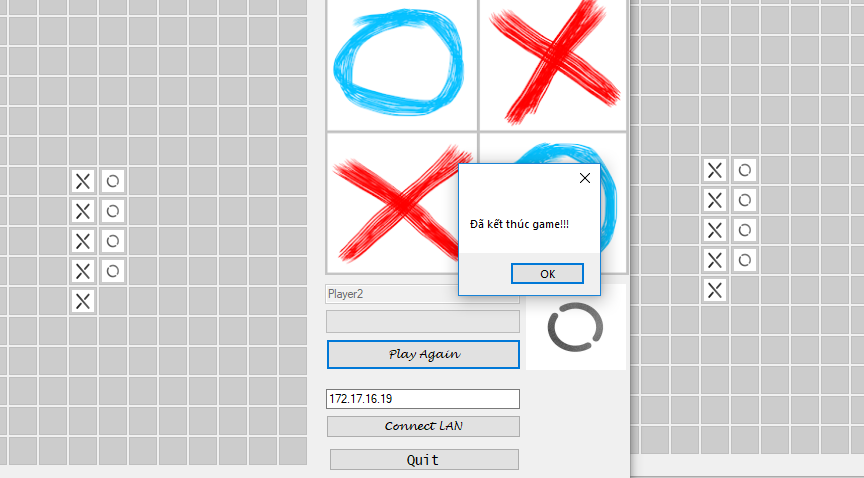
else

break;

}

return countTop + countbottom == 5;

}



\* Hàm thắng ngang

//Hàm thắng ngang

private bool isEndHorizontal(Button btn)

{

//Xét vị trí của button

Point point = GetChessPoint(btn);

int countLeft = 0;

for (int i = point.X; i >= 0 ; i--)

{

if (matrix[point.Y][i].BackgroundImage == btn.BackgroundImage)

{

countLeft++;

}

else

break;

}

int countRight = 0;

for (int i = point.X + 1; i < ChessBoard.Chess\_board\_width ; i++)

{

if (matrix[point.Y][i].BackgroundImage == btn.BackgroundImage)

{

countRight++;

}

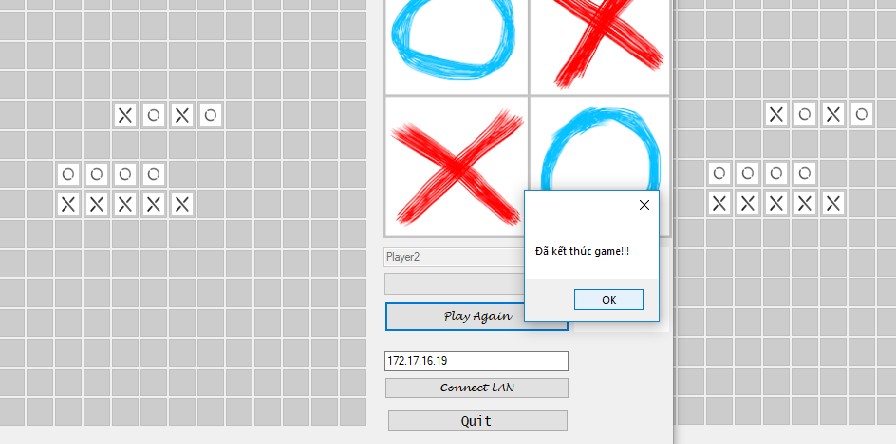
else

break;

}

return countLeft + countRight == 5;

}



\* Hàm thắng đường chéo chính

//Hàm thắng đường chéo chính

private bool isEndDiagonal(Button btn)

{

Point point = GetChessPoint(btn);

int countTop = 0;

for (int i = 0; i <= point.X; i++)

{

if (point.Y - i < 0 || point.X - i < 0)

break;

if (matrix[point.Y - i][point.X - i].BackgroundImage == btn.BackgroundImage)

{

countTop++;

}

else

break;

}

int countbottom = 0;

for (int i = 1; i <= ChessBoard.Chess\_board\_width - point.X; i++)

{

if (point.X + i >= ChessBoard.Chess\_board\_width || point.Y + i >= ChessBoard.Chess\_board\_height)

break;

if (matrix[point.Y + i][point.X + i].BackgroundImage == btn.BackgroundImage)

{

countbottom++;

}

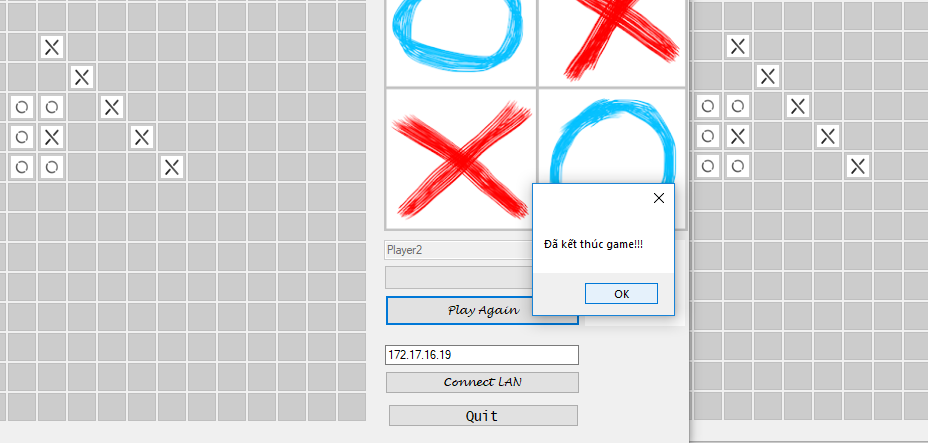
else

break;

}

return countTop + countbottom == 5;

}



\* Hàm thắng đường chéo phụ

//Hàm thắng đường chéo phụ

private bool isEndSub(Button btn)

{

Point point = GetChessPoint(btn);

int countTop = 0;

for (int i = 0; i <= point.X; i++)

{

if (point.Y - i < 0 || point.X + i >= ChessBoard.Chess\_board\_width)

break;

if (matrix[point.Y - i][point.X + i].BackgroundImage == btn.BackgroundImage)

{

countTop++;

}

else

break;

}

int countbottom = 0;

for (int i = 1; i <= ChessBoard.Chess\_board\_width - point.X; i++)

{

if (point.X - i < 0 || point.Y + i >= ChessBoard.Chess\_board\_height)

break;

if (matrix[point.Y + i][point.X - i].BackgroundImage == btn.BackgroundImage)

{

countbottom++;

}

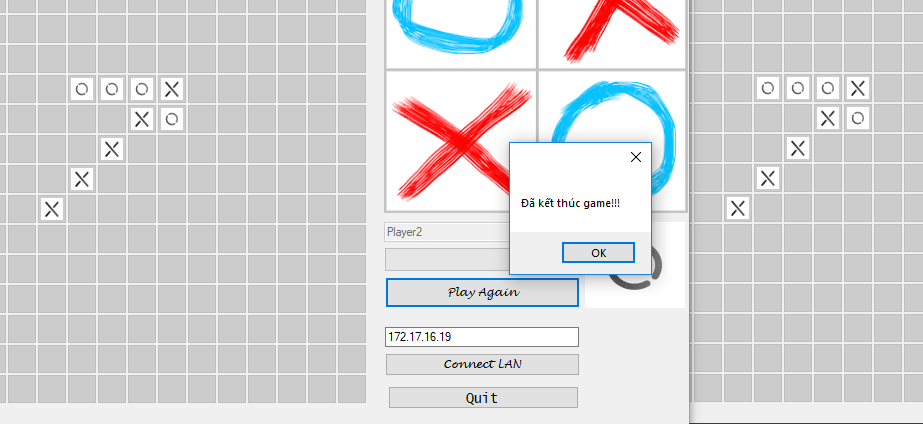
else

break;

}

return countTop + countbottom == 5;

}



**3.2.Cài đặt mạng LAN**

**3.2.1. Khái niệm địa chỉ và cổng (Address & Port)**

***Nguyên lí:*** Trong một máy có rất nhiều ứng dụng muốn trao đối với các ứng dụng khác thông qua mạng (ví dụ trên có 2 ứng dụng trong máy A muốn trao đổi với với 2 ứng dụng trên máy B) và mỗi máy tính chỉ có duy nhất một đường truyền dữ liệu (để gửi và nhận).

***Vấn đề :*** Rất có thể xảy ra "nhầm lẫn" khi dữ liệu từ máy A gửi đến máy B thì không biết là dữ liệu đó gửi cho ứng dụng nào trên máy B?

***Giải quyết:*** Mỗi ứng dụng trên máy B sẽ được gán một số hiệu (mà ta vẫn quen gọi là cổng : Port), số hiệu cổng này từ 1..65535. Khi ứng dụng trên máy A muốn gửi cho ứng dụng nào trên máy B thì chỉ việc điền thêm số hiệu cổng (vào trường RemotePort) vào gói tin cần gửi. Trên máy B, các ứng dụng chỉ việc kiểm tra giá trị cổng trên mỗi gói tin xem có trùng với số hiệu cổng của mình (đã được gán – chính là giá trị Localport) hay không? Nếu bằng thì xử lý, còn trái lại thì không làm gì (vì không phải là của mình).

***Như vậy***: Khi cần trao đổi dữ liệu cho nhau thì hai ứng dụng cần phải biết thông tin tối thiểu là địa chỉ (Address) và số hiệu cổng (Port) của ứng dụng

**3.2.2. Lớp IPAddress**

Trên Internet mỗi một trạm (có thể là máy tính, máy in, thiết bị …) đều có một định danh duy nhất, định danh đó thường được gọi là một địa chỉ (Address). Địa chỉ trên Internet là một tập hợp gồm 4 con số có giá trị từ 0-255 và cách nhau bởi dấu chấm.

Để thể hiện địa chỉ này, người ta có thể viết dưới các dạng sau:

+ Tên : Ví dụ May01, Server, ….

+ Địa chỉ IP nhưng đặt trong một xâu: "192.168.1.1", "127.0.0.1"

**3.2.3. IPendPoint**

Trong mạng, để hai trạm có thể trao đổi thông tin được với nhau thì chúng cần phải biết được địa chỉ (IP) của nhau và số hiệu cổng mà hai bên dùng để trao đổi thông tin. Lớp IPAddress mới chỉ cung cấp cho ta một vế là địa chỉ IP (IPAddress), như vậy vẫn còn thiếu vế thứ hai là số hiệu cổng (Port number). Như vậy, lớp IPEndpoint chính là lớp chứa đựng cả IPAddress và Port number.

Đối tượng IPEndpoint sẽ được dùng sau này để truyền trực tiếp cho các đối tượng UDP, TCP…

**3.2.4. Giao thức TCP**

Để đảm bảo độ tin cậy trong các ứng dụng mạng, người ta còn dùng một giao thức khác, gọi là giao thức có kết nối : TCP (Transport Control Protocol). Trên Internet chủ yếu là dùng loại giao thức này, ví dụ như Telnet, HTTP, SMTP, POP3… Để lập trình theo giao thức TCP, MS.NET cung cấp hai lớp có tên là TCPClient và TCPListener.

**3.2.5. TCPListener**

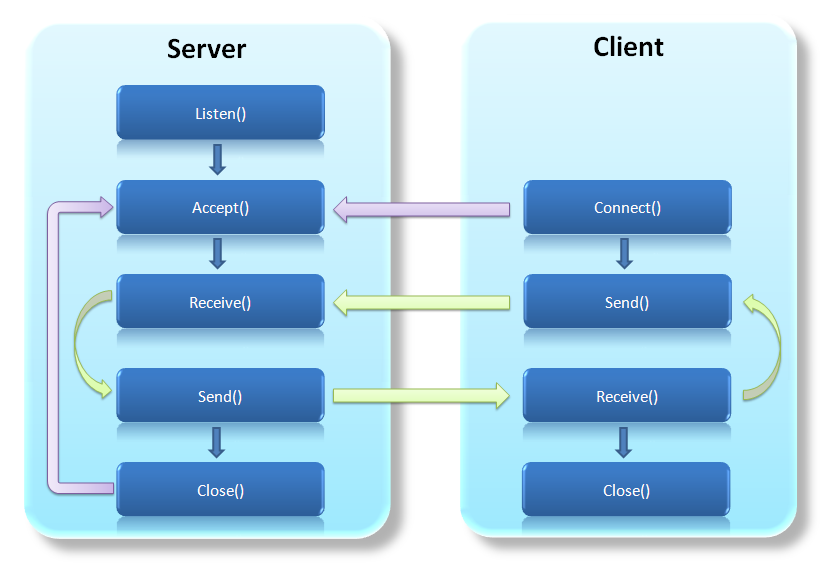
TCPListerner là một lớp cho phép người lập trình có thể xây dựng các ứng dụng Server (Ví dụ như SMTP Server, FTP Server, DNS Server, POP3 Server hay server tự định nghĩa ….). Ứng dụng server khác với ứng dụng Client ở chỗ nó luôn luôn thực hiện lắng nghe và chấp nhận các kết nối đến từ Client.

**3.2.6.Phân tích từ nhu cầu thực tiễn và yêu cầu**

Hiện nay, mạng Lan phát triển rất mạnh cả trong trường học, các cơ quan tổ chức và ở cả các hộ gia đình. Chính điều đó kéo theo nhu cầu liên lạc trao đổi thông tin trong mạng Lan cũng phát triển theo

**3.2.7. Mô hình kết nối client – server theo chế độ có nối kết( TCP)**

Dưới đây là mô hình kết nối giữa client – server



***3.2.7.1.Đối với Client:***

- Tạo một đối tượng client.

- Với lớp IPendPoint ta có thể có địa chỉ IP và port để kết nối đến server.

- Khi có địa chỉ IP và Port ta có thể kết nối đến server.

Socket Client;

//Hàm kết nối với server

public bool ConnectServer()

{

IPEndPoint iep = new IPEndPoint(IPAddress.Parse(IP), Port);

Client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

//Cố gắng để kết nối

try

{

Client.Connect(iep);

return true;

}

catch

{

return false;

}

}

***3.2.7.2.Đối với Server:***

*-* Tạo một đối tượng Listener để bắt đầu “lắng nghe” trên một cổng cục bộ

*-* Đợi và chấp nhận kết nối từ client với phương phức Accept

- Thực hiện giao tiếp với client.

Socket server;

//Hàm tạo server

public void CreatServer()

{

IPEndPoint iep = new IPEndPoint(IPAddress.Parse(IP), Port);

server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

server.Bind(iep);

server.Listen(10);

//Tạo một luồng riêng để khởi tạo server

Thread acceptClient = new Thread(() =>

{

Client = server.Accept();

});

//Để tránh chương trình bị cắt ngang mà kết nối không tắt, gây tốn tài nguyên

acceptClient.IsBackground = true;

acceptClient.Start();

}

\*Ở đây: Phương thức Accept(), cho phép client kết nối tới server được bỏ trong luồng thread, bởi vì muốn việc chấp nhận kết nối không ảnh hưởng gì tới hàm main() (Ở đây là game caro) đồng thời tránh thất thoát.

**3.2.8.Nhận gửi dữ liệu giữa client – server.**

***3.2.8.1.Gửi dữ liệu***

*+* Phân rã data thành một chuỗi byte[], đồng thời kiểm tra xem dữ liệu đã được gửi hay chưa?

public bool Send(object data)

{

//Phân ra dữ liệu

byte[] sendData = SerializeData(data);

//Kiểm tra dữ liệu đã được gửi hay chưa? Bằng hàm SendData

return SendData(Client, sendData);

}

private bool SendData(Socket target, byte[] data)

{

return target.Send(data) == 1 ? true : false;

}

***3.2.8.2.Nhận dữ liệu***

*+* Chuyển chuỗi byte[] thành đối tượng ban đầu đồng thời kiểm tra xem dữ liệu đã được nhận hay chưa?

public object Reveive()

{

byte[] receiveData = new byte[BUFFER];

//Xem xem nhận dữ liệu có thành công hay không?

bool isOK = ReceiveData(Client, receiveData);

return DeserializeData(receiveData);

}

private bool ReceiveData(Socket target, byte[] data)

{

return target.Receive(data) == 1 ? true : false;

}

***3.2.8.3.Gửi dữ liệu bằng cách phân mảnh dữ liệu thành mảng byte[] (Serialize).***

*1.Khái niệm.*

Là tiến trình chuyển đổi một đối tượng thành chuỗi tuần tự các byte để có thể lưu trữ hoặc trao đổi.

*2.Các bước thực hiện.*

+ Tạo đối tượng Stream lưu kết quả chuyển đổi.

+ Tạo đối tượng BinaryFormatter.

+ Gọi phương thức BinaryFormatter.Serialize để chuyển đổi, lưu kết quả vào Stream.

public byte[] SerializeData(Object o)

{

MemoryStream ms = new MemoryStream();

BinaryFormatter bf1 = new BinaryFormatter();

bf1.Serialize(ms, o);

return ms.ToArray();

}

***3.2.8.4.Nhận dữ liệu bằng cách chuyển chuỗi byte[] thành đối tượng ban đầu (Deserialize).***

*1.Khái niệm*

Là tiến trình chuyển chuỗi tuần tự các byte thu được trong quá trình serialize thành đối tượng ban đầu.

*2.Các bước thực hiện*

+ Tạo Stream đọc kết quả của quá trình Serialize

+ Tạo đối tượng BinaryFormatter

+ Tạo đối tượng lưu dữ liệu sau khi chuyển đổi

+ Gọi phương thức BinaryFormatter.Deserialize để chuyển đổi lại và ép kiểu phù hợp với kiểu của đối tượng ban đầu.

public object DeserializeData(byte[] theByteArray)

{

MemoryStream ms = new MemoryStream(theByteArray);

BinaryFormatter bf1 = new BinaryFormatter();

ms.Position = 0;

return bf1.Deserialize(ms);

}

# PHẦN III: KẾT LUẬN

## 1. Kết quả đạt được

Qua quá trình tìm hiểu nhóm chúng em đã nắm được các kiến thức cơ bản và nâng cao của lập trình Winform và đã áp dụng vào xây dựng chương trình chơi cờ caro với giao diện sinh động, đơn giản mà hấp dẫn.

Sau thời gian thực hiện đề tài, chương trình đã hoàn thành và đạt được một số kết quả sau:

* Game chơi cờ caro có giao diện dễ nhìn, dễ sử dụng.
* Trò chơi có thể chơi 2 người qua mạng LAN

## 2. Hạn chế của đề tài

Game chưa xây dựng được chưa có nhiều các chức năng như có thể lưu lại ván cờ để lần sau có thể chơi tiếp, chức năng chơi với máy, chức năng chat với người chơi còn lại.

## 3. Hướng phát triển của đề tài

Với sự cố gắng, lỗ lực nhóm chúng em đã cố gắng hoàn thành yêu cầu đề tài. Do thời gian và năng lực có hạn nên game mới chỉ đáp ứng được chơi cờ caro bình thường mà chưa có các chức năng tiện ích khác.

Hướng phát triển Game trở thành một chương trình chơi cờ với một số chức năng thú vị như chat với người chơi còn lại, thả icon khi chơi...

## 4. Đề nghị ý kiến.

Trong thời gian thực hiện đề tài này nhóm đề tài không thể tránh khỏi những thiếu sót, chúng em rất mong nhận được ý kiến đóng góp từ phía quý Thầy và các bạn để nhóm hoàn thiện hơn.

**PHẦN IV: PHÂN RÃ CÔNG VIỆC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Công việc** | **Duration (days)** | **Assign To** | **% Finished** |
| 1 | Tìm hiểu sơ bộ & đăng ký đồ án |  | Tường | 100% |
| 2 | Tìm hiểu kiến thức liên quan |  | Tường | 100% |
| **I. VIẾT BÁO CÁO** | | | | |
| 3 | **LỜI CẢM ƠN** |  | Tùng,Tường,Khôi |  |
| 4 | **PHẦN I: MỞ ĐẦU** |  | Tùng | 100% |
| 5 | **PHẦN II: NỘI DUNG** |  |  |  |
| 6 | **Chương I: Giới thiệu trò chơi caro** |  |  |  |
| 7 | 1.1 Giới thiệu sơ lược về trò chơi caro |  | Tùng,Khôi | 100% |
| 8 | 1.2 Giới thiệu sơ lược về đề tài |  | Tùng,Khôi | 100% |
|  |  |  |  |  |
| 10 | **Chương 2: Giải thuật của chương trình** |  | Khôi | 100% |
| 11 | **Chương 3: Thiết kế chương trình và cài đặt mạng LAN** |  |  |  |
| 12 | 3.1 Thiết kế chương trình |  | Khôi, Tùng | 100% |
| 13 | 3.2 Cài đặt mạng LAN |  | Khôi, Tùng | 100% |
| 14 | **PHẦN 3: KẾT LUẬN** |  |  | 100% |
| 15 | 1/ Kết quả đạt được |  | Tường, Khôi |  |
| 16 | 2/ Hạn chế của đè tài |  | Tường | 100% |
| 17 | 3/ Hướng phát triển của đề tài |  | Tùng, Khôi | 100% |
| 18 | 4/ Đề nghị ý kiến |  | Tường | 100% |
| 19 | **Kiểm thử** |  | Tường | 100% |
| 20 | **Tài kiệu tham khảo** |  | Tùng, Khôi | 100% |
| **II. LẬP TRÌNH** | | | | |
|  | 1. Lên ý tưởng và chốt đồ án |  | Khôi, Tường | 80% |
|  | 2. Giao diện |  | Tùng | 80% |
|  | 3. Code |  | Khôi | 80% |
|  | 4. Tạo file setup cho game |  | Tùng | 80% |
|  | 5. Kiểm thử game qua các lần thay đổi |  | Tường | 90% |
| **III. KIỂM THỬ** | | | | |
|  | 1. Unit Test |  | Khôi | 90% |
|  | 2. Integration Test |  | Khôi | 90% |
|  | 3. System Test |  | Tường | 90% |
|  | 4. Acceptance Test |  | Tùng | 90% |

# PHẦN V: TÀI LIỆU THAM KHẢO

* 1. **Các giải pháp lập trình C# của 2** tác giả Nguyễn Ngọc Bình Phương và Thái Thanh Phong.
  2. Lập trình môi trường Windows của 3 tác giả Phạm Thi Vương - Lê Minh Trí - Nguyễn Hoàng Vũ.

.