**BÁO CÁO**

**BÀI 5**

**ĐỀ TÀI:** TRÒ CHƠI CARO

Sinh viên: Lê Văn Tân

Mssv: 58131382

NỘI DUNG:

1. Giới thiệu.
2. Hướng làm và code.
3. Giới thiệu:

Game caro đơn giản, chỉ có chức năng chơi giữa người với người, menu chỉ có chức năng quit.

Luật chơi:

* Khi người chơi có 5 quân cờ giống nhau liên tiếp trên cùng một hàng ( ngang hoặc dọc) sẽ chiến thắng.
* Khi người chơi có 5 quân cờ giống nhau liên tiếp trên cùng một đường chéo chính hoặc chéo phụ sẽ chiến thắng.

1. Hướng làm và code.

* Ngôn ngữ để viết game: C#.
* Thiết kế form cho game.
* Cách đổi người chơi trong game.
* Cách xử lí thắng thua.

1. Thiết kế form cho game:

* Dùng các panel để bố trí vị trí của bàn cờ, vị trí của thông tin người chơi.
* Vẽ bàn cờ : Các ô trên bàn cờ cơ bản giống như các button nên có thể add các button để tạo thành bàn cờ. Các lệnh sẽ được đặt trong một hàm DrawChessBoard() , add các button bằng cách cho 2 vòng for lồng nhau để tạo dãy các button.

1. Cách đổi người chơi trong game:

* Quy luật đổi người chơi: Khi nhấn vào một button thì một textbox sẽ hiện tên và một picturebox sẽ hiện kí hiệu của người chơi tiếp theo.
* Thêm event Click cho button và viết hàm Click. Trong hàm tạo sự kiện Click, ta ép kiểu cho button như là một sender, chính sender này sẽ gửi event Click đi.
* Thay đổi image của button bằng cách gắn đường dẫn của file ảnh.
* Tạo đối tượng người chơi gồm các thuộc tính : NamePlayer là tên người chơi và Mark là kí hiệu đại diện của người chơi.
* Đồng thời tạo ra một biến currentplayer để biết được người chơi nào đang đánh. Mỗi khi click vào button của currentplayer hiện tại nó sẽ thay đổi backgroundimage. Nếu backgroundimage của current hiện tại khác null, nghĩa là sẽ không thực hiện đổi image của button hiện tại nữa.

1. Cách xử lí thắng thua:

* Cách làm: từ vị trí của một button hiện tại ta sẽ kiểm duyệt sang phía trái và phía phải của button đó xem có bao nhiêu button giống nhau liên tiếp đồng thời cộng chúng lại nếu đúng bằng 5 thì dừng trò chơi.
* Lưu tất cả các button trong một ma trận bằng cách tạo một list lồng bên trong một list.
* Đồng thời ta cũng sẽ tạo ma trận trong lúc vẽ bàn cờ.

1. Duyệt theo hàng ngang, dọc:

* Khởi tạo một biến đếm countleft = 0, countright = 0;
* Cho một vòng for chạy từ tọa độ X của button hiện tại đang xét về bên trái, kiểm tra xem backgroundimage của button đó có giống với backgroundimage của button hiện tại không, nếu có thì tăng biến đếm countleft lên 1 đơn vị, tương tự tiếp tục duyệt qua bên phải của button hiện tại sau đó ta cộng countleft và countright lại với nhau nếu đúng bằng 5 thì xuất thông báo và dừng game.
* Tương tự với hàng dọc.

1. Duyệt theo đường chéo:

* Với đường chéo chính thì tọa độ X, Y của button sẽ tăng hoặc giảm cùng nhau, t tạo 2 vòng for cùng chạy, một vòng chạy với giá trị cùng tăng của X và Y của button, một vòng chạy với giá trị cùng giảm của X và Y của button, thực hiện bước kiểm tra giống trên và sau đó cộng 2 biến đếm lại với nhau nếu đúng bằng 5 sẽ xuất thông báo và dừng game.
* Tương tự kiểm tra với đường chéo phụ, tuy nhiên với tọa của button là một là X tăng Y giảm và hai là X giảm Y tăng.

1. Một số đoạn code trong bài.

-Vẽ bàn cờ:

public void DrawChessBoard() // Hàm vẽ bàn cờ

{

Matrix = new List<List<Button>>(); // Danh sách nút tạo nên ma trận

Button oldButton = new Button() { Width = 0, Location = new Point(0, 0) }; // Set vị trí và kích thước của oldButton

for (int i = 0; i < 14; i++)

{

Matrix.Add(new List<Button>()); // Tạo mảng mới để lưu

for (int j = 0; j < 14; j++)

{

Button btn = new Button()

{

Width = Cons.CHESS\_WIDTH,

Height = Cons.CHESS\_HEIGHT,

Location = new Point(oldButton.Location.X + Cons.CHESS\_WIDTH, oldButton.Location.Y),

BackgroundImageLayout = ImageLayout.Stretch, // Chỉnh layout theo kích thước button

Tag = i.ToString() // Biến lưu vị trí của button

};

btn.Click += btn\_Click; // Sự kiện khi click vào nút

chessBoard.Controls.Add(btn);

Matrix[i].Add(btn); // Add button vào ma trận

oldButton = btn;

}

oldButton.Location = new Point(0, oldButton.Location.Y + Cons.CHESS\_HEIGHT);

oldButton.Width = 0;

oldButton.Height = 0;

}

}

* Tạo đối tượng người chơi:

public ChessBoardManager(Panel chessBoard, TextBox playerName, PictureBox mark)

{

this.chessBoard = chessBoard;

this.PlayerName = playerName;

this.playerMark = mark;

this.Players = new List<Player>()

{

new Player("Player\_1", Image.FromFile(Application.StartupPath + "\\Resources\\X.png")), // Tạo đối tượng và kí hiệu của người chơi

new Player("Player\_2", Image.FromFile(Application.StartupPath + "\\Resources\\O.png")) // Tạo đối tượng và kí hiệu của người chơi

};

CurentPlayer = 0;

ChangePlayer();

}

* Sự kiện khi click vào một button:

void btn\_Click(object sender, EventArgs e)

{

Button btn = sender as Button; // Ép kiểu button cho một sender

if (btn.BackgroundImage != null)

return;

Mark(btn);

ChangePlayer();

if(isEndGame(btn))

{

EndGame();}}

* Xử lí thắng thua theo hàng ngang:

private bool isEndNgang(Button btn)

{

Point point = GetChessPoint(btn); // Lấy tọa độ của button hiện tại

int countLeft = 0;

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

{

// Kiểm tra xem backgroundimage của button đó có giống với backgroundimage của button hiện tại không

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

{

countLeft++;

}

else

break;

}

int countRight = 0;

for (int i = point.X + 1; i < Cons.CHESS\_WIDTH; i++)

{

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

{

countRight++;

}

else

break;

}

return countLeft + countRight == 5;

}