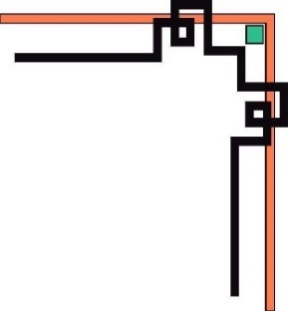
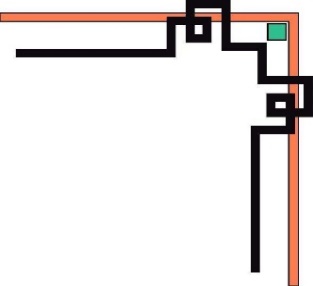
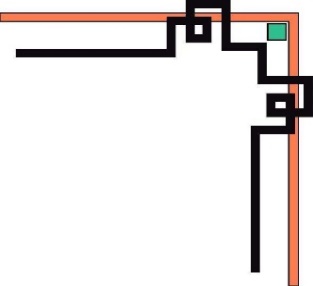
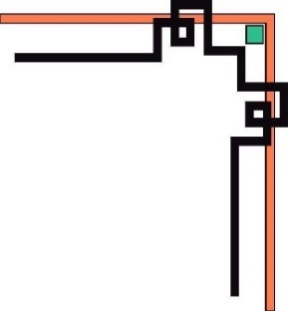
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH**



**ĐỒ ÁN KẾT THÚC HỌC PHẦN**

**PHÁT TRIỂN GAME CARO**

**SVTH : TRẦN MINH QUANG**

**MSSV : 20110704**

**SVTH : VÕ ĐINH QUỐC THUẬT**

**MSSV : 20110733**

**SVTH : ĐINH VĂN TOÀN**

**MSSV : 20110738**

**Khoá: K20**

**Ngành: CÔNG NGHỆ THÔNG TIN**

**Môn: Lập trình hướng đối tượng**

**GVHD: TS. LÊ VĨNH THỊNH**

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

---\*\*\*---

Tp. Hồ Chí Minh, ngày 30 tháng 7 năm 2020

**NHIỆM VỤ ĐỒ ÁN**

Họ và tên sinh viên: TRẦN MINH QUANG MSSV: 20110704

Họ và tên sinh viên: VÕ ĐINH QUỐC THUẬT MSSV: 20110733

Họ và tên sinh viên: ĐINH VĂN TOÀN MSSV: 20110738

Ngành: Công nghệ thông tin Lớp: 201104B

Giảng viên hướng dẫn: TS. LÊ VĨNH THỊNH ĐT: +84 938 252 222

Ngày nhận đề tài: 21/09/2021 Ngày nộp đề tài: 14/12/2021

1. Tên đề tài: Phát triển game caro

2. Các số liệu, tài liệu ban đầu:

* Báo cáo đồ án (được thực hiện bởi nhóm)

3. Nội dung thực hiện đề tài:

* Tìm hiểu và tham khảo các source code trên Github.
* Thiết kế giao diện game
* Xây dựng các chức năng: new game, undo, quit, play with computer, two player, nghe nhạc khi chơi game.

4. Sản phẩm

App game caro.

**TÓM TẮT**

1. **Nội dung game Caro**

Game cờ caro hay còn được biết đến với tên gọi là chơi XO. Đây được xem là một trò chơi dân gian và xuất hiện từ lâu đời tại Việt Nam. Biết bao nhiêu thế hệ người dân Việt Nam đều đã chơi loại cờ này. Đặc biệt là ở độ tuổi học sinh. Hiện tại, về cơ bản thì với game cờ caro, người chơi sẽ chỉ cần làm sao ghi được 5 nước liên tiếp là có thể giành chiến thắng. Nghe có vẻ đơn giản thế nhưng, game cờ caro lại là những màn đấu trí vô cùng căng thẳng. Ngày nay, game cờ caro cũng có khá nhiều biến thể khác nhau. Điều này nhằm tạo sự phù hợp với người chơi ở từng quốc gia, khu vực hay vùng miền khác nhau. Và ở kỷ nguyên 4.0 thì sẽ không thể thiếu được game cờ caro trên các nền tảng máy tính cũng như điện thoại.

1. **Chức năng trong game**

Chúng em cung cấp đầy đủ những chức năng cần thiết cho một game caro bao gồm: tạo một ván game mới, chức năng đi lại (undo), thoát game, nghe nhạc khi đang chơi, được đặt tên người chơi.

MỤC LỤC

[NHIỆM VỤ ĐỒ ÁN](#_Toc44455430)

[TÓM TẮT](#_Toc44455434)

[MỤC LỤC](#_Toc44455436)

[CHƯƠNG 1: GIỚI THIỆU](#_Toc44455440)

[1.1. Giới thiệu đề tài](#_Toc44455441)

[1.2. Mục tiêu đề tài](#_Toc44455442)

[1.3. Công cụ thực hiện đề tài](#_Toc44455443)

[1.5. Phương pháp nghiên cứu](#_Toc44455445)

CHƯƠNG 2: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

2.1 Định nghĩa

2.2 Các đặc tính

CHƯƠNG 3: CẤU TRÚC GAME CARO

3.1 Luồng chương trình

3.2 Class diagram

3.3 Các class trong việc xây dựng code

3.3.1 Program

3.3.2 ChessBoardManager

3.3.2.1 Properties

3.3.2.2 Initialize

3.3.2.3 Methods

3.3.2.4 Play with computer

3.3.3 Player

3.3.4 PlayInfo

3.3.5 MusicFile

3.3.5 Const

[CHƯƠNG 4: KẾT LUẬN](#_Toc44455501)

**CHƯƠNG 1: GIỚI THIỆU**

**1.1 Giới thiệu đề tài**

Đã trải qua bao thập kỷ và cho đến nay Caro vẫn được ưa chuộng trên toàn thế giới, với cách chơi đơn giản, nhưng càng chơi thì sự biến hóa của các nước cờ là vô cùng kì diệu. Sự hấp dẫn game Caro đã lôi cuốn các bạn trẻ Việt Nam rất mạnh, nhất là đối với các học sinh, sinh viên. Sau giờ nghỉ giải lao đều có thể chơi với 1 mảnh giấy nhỏ kẻ ô vuông và 2 cây bút. Và đây chính là 1 game giải trí lành mạnh, phát triển trí tuệ cho các bạn.

Giờ đây vơi sự phát triển của công nghệ, Game Caro đã được đưa vào PC với rất nhiều các phiên bản cùng nhiều cấp độ cho người chơi với máy, hay hai người chơi vơi nhau. Cùng với đó Caro cũng chiếm 1 phần trong làng game giải trí online, chủ yếu để 2 người chơi thi đấu và thư giãn khi đánh với máy. Thật tiện ích khi game caro được đưa vào các điện thoại di động có thể chơi được mọi lúc.

**1.2 Mục tiêu đề tài**

Ứng dụng game caro được xây dựng nhằm mục đích làm quen với ngôn ngữ, các lệnh cơ bản và ứng dụng được phương pháp lập trình hướng đối tượng vào một ứng dụng thực tế.

Ứng dụng còn có mục đích giải trí nhẹ nhàng cho những ai đang căng thẳng.

**1.3 Công cụ thực hiện đề tài**

Sử dụng framework Windows Forms (gọi tắt winforms).

**1.4 Phương pháp nghiên cứu**

Lý‎ thuyết thực tiễn: tìm hiểu tài liệu từ các website, tài liệu có liên quan đến nghiệp vụ đang tìm hiểu.

Tham khảo chuyên gia: Tham khảo và tiếp thu ý kiến của thầy hướng dẫn, của các đàn anh đi trước và bạn bè có kinh nghiệm liên quan đến các vấn đề của đồ án.

Phát triển mô hình: phát triển dự án theo mô hình phát triển phần mềm từ đặc tả, thiết kế đến hoàn thành sản phẩm.

**CHƯƠNG 2: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**2.1 Định nghĩa**

OOP (viết tắt của Object Oriented Programming) – lập trình hướng đối tượng là một phương pháp lập trình dựa trên khái niệm về lớp và đối tượng. OOP tập trung vào các đối tượng thao tác hơn là logic để thao tác chúng, giúp code dễ quản lý, tái sử dụng được và dễ bảo trì.

Mục tiêu của OOP là tối ưu việc quản lý source code, giúp tăng khả năng tái sử dụng và quan trọng hơn hết là giúp tóm gọn các thủ tục đã biết trước tính chất thông qua việc sử dụng các đối tượng.

Đối tượng trong OOP bao gồm 2 thành phần chính:

+ Thuộc tính (Attribute): là những thông tin, đặc điểm của đối tượng

+ Phương thức (Method): là những hành vi mà đối tượng có thể thực hiện

Lớp là sự trừu tượng hóa của đối tượng. Những đối tượng có những đặc tính tương tự nhau sẽ được tập hợp thành một lớp. Lớp cũng sẽ bao gồm 2 thông tin là thuộc tính và phương thức. Một đối tượng sẽ được xem là một thực thể của lớp.

**2.2 Các đặc tính**

Đóng gói: là 1 đặc tính che dấu tiến trình thực thi và chi tiết của 1 đối tượng cụ thể, các dữ liệu được đóng gói và không thể sử dụng bởi các hàm hay phương thức bên ngoài, ngoại trừ sử dụng các phương thứ đặc biệt và các thành phần bên trong của nó. Tính chất này giúp tăng tính bảo mật cho đối tượng và tránh tình trạng dữ liệu bị hư hỏng ngoài ý muốn.

Đa hình: các phương thức, hành động có cùng 1 tên gọi nhưng thực hiện các chức năng khác nhau trong các ngữ cảnh khác nhau.

Kế thừa: cho phép xây dựng một lớp mới (lớp Con), kế thừa và tái sử dụng các thuộc tính, phương thức dựa trên lớp cũ (lớp Cha) đã có trước đó. Các lớp Con kế thừa toàn bộ thành phần của lớp Cha và không cần phải định nghĩa lại. Lớp Con có thể mở rộng các thành phần kế thừa hoặc bổ sung những thành phần mới.

Trừu tượng: giúp loại bỏ những thứ phức tạp, không cần thiết của đối tượng và chỉ tập trung vào những gì cốt lõi, quan trọng.

**CHƯƠNG 3: CẤU TRÚC GAME CARO**

**3.1 Luồng chương trình**

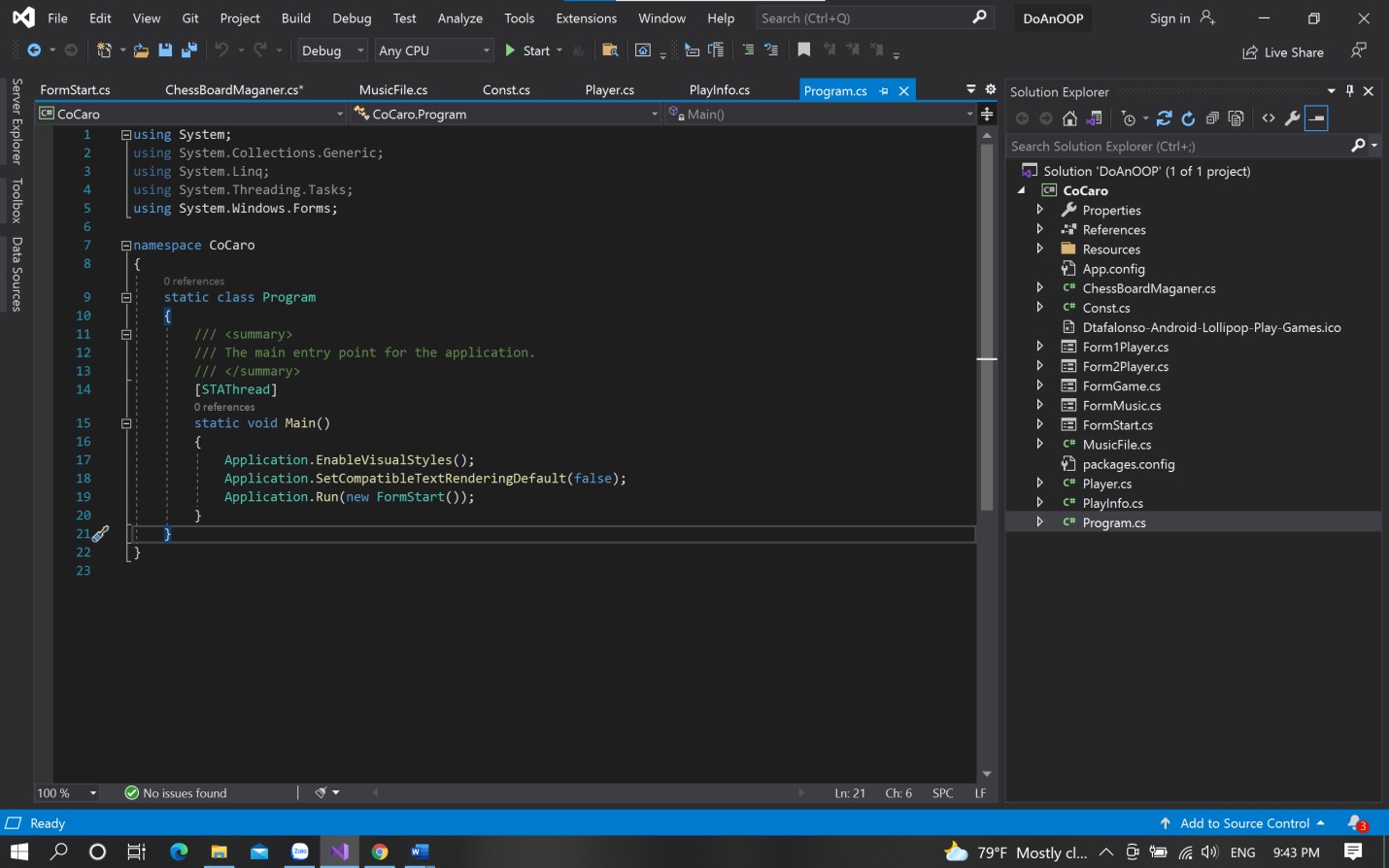
|  |  |  |
| --- | --- | --- |
| TT | Màn hình, cửa sổ | Mục đích |
| 1 | Hình 1. Giao diện FormStart | Tạo form gồm 2 button để người chơi có thể chọn 1 trong 2 chế độ: Chơi với máy hoặc Chơi 2 người. |
| 2 | Hình 2. Giao diện Form1Player | Xuất hiện khi người chơi chọn chế độ Chơi với máy. Gồm 1 TextBox để nhập tên và 1 ComboBox để chọn chế độ chơi tương thích với bản thân, 1 Button để chuyển đến FormGame. |
| 3 | Hình 3. Giao diện Form2Player | Xuất hiện khi người chơi chọn chế độ Chơi 2 người, gồm 1 TextBox để nhập tên người chơi 1 và 1 TextBox nhập tên người chơi 2, 1 Button để chuyển đến FormGame. |
| 4 | Hình 4. Giao diện FormGame | Gồm Panel chứa bảng game để chơi, MenuItem chứa các chức năng của game như New game, Undo, Quit và 1 Button để khởi chạy FormMusic. |
| 5 | Hình 5. Giao diện FormMusic | Chức năng của form là phát nhạc gồm 1 Button để chọn file cần phát và 1 ListBox liệt kê danh sách phát. |

**3.2 Class diagram:**



**3.3 Các class trong việc xây dựng code**

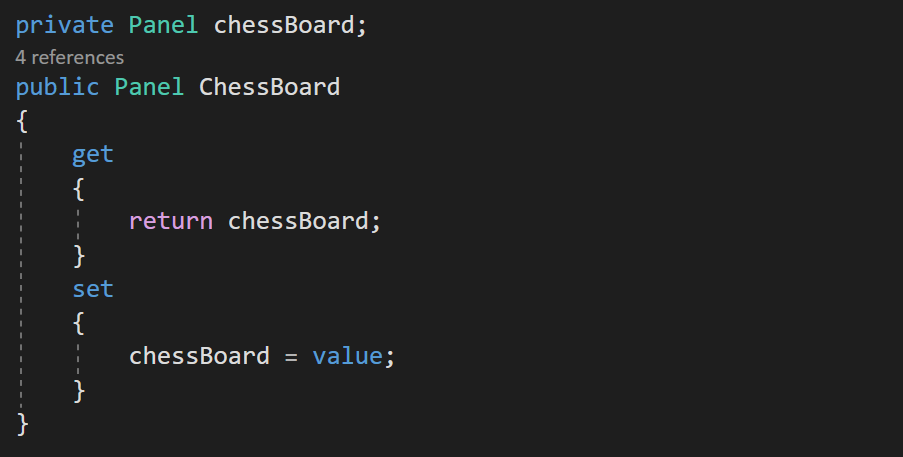
**3.3.1 Program**



Class Program là một hàm Main bắt đầu trò chơi của chúng ta, khi khởi chạy sẽ gọi cửa sổ màn hình bắt đầu trò chơi.

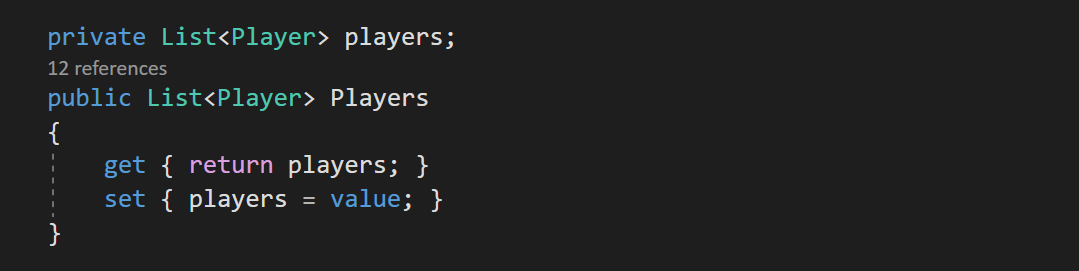
**3.3.2 ChessBoardManager**

***3.3.2.1 Properties***



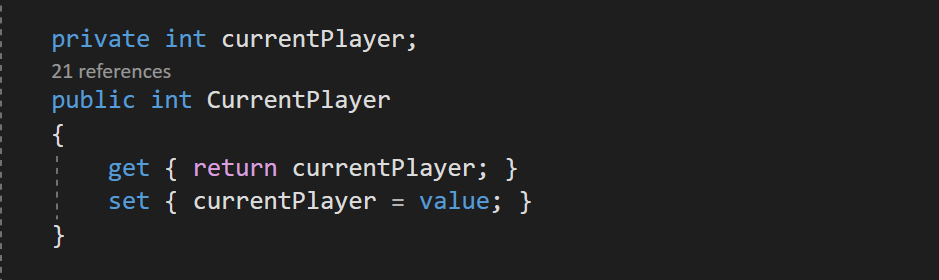
Khởi tạo biến private chessBoard để miêu tả bàn cờ và property ChessBoard để gọi giá trị cũng như gán giá trị cho chessBoard

//



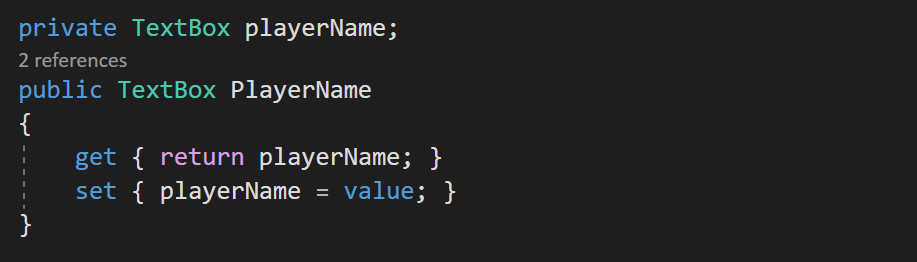
Khởi tạo biến private player để miêu tả người chơi và property Player để gọi giá trị cũng như gán giá trị cho player

//



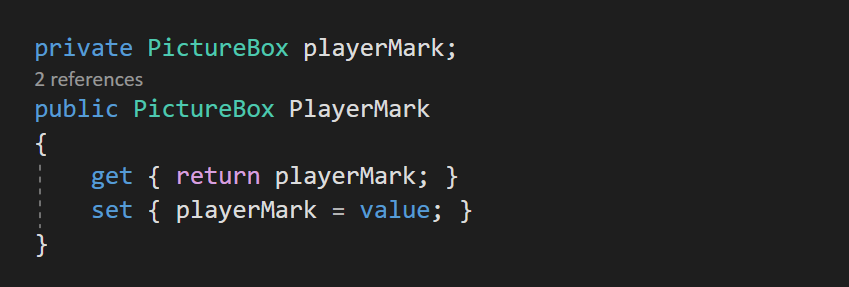
Khởi tạo biến private currentPlayer để miêu tả người chơi hiện tại và property CurrentPlayer để gọi giá trị cũng như gán giá trị cho currentPlayer

//



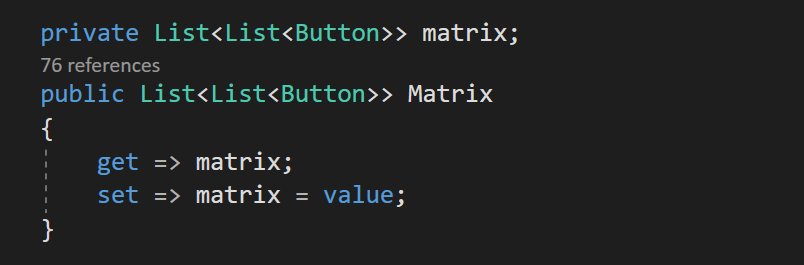
Khởi tạo biến private playerName để miêu tả người chơi tên gì và property PlayerName để gọi giá trị cũng như gán giá trị cho playerName

//



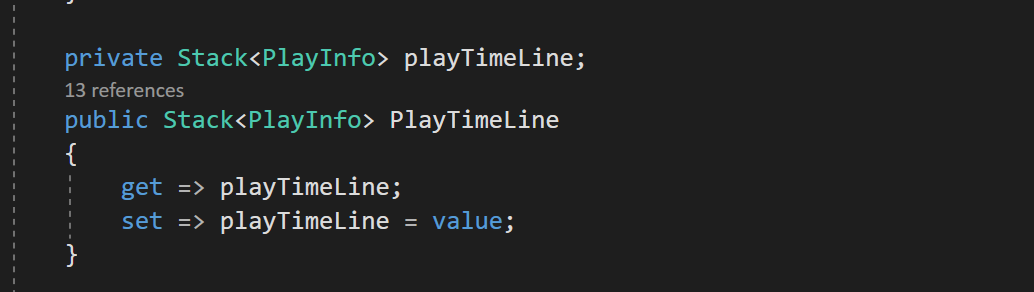
Khởi tạo biến private playerMark để miêu tả người chơi hiện tại là O hay X và property PlayerMark để gọi giá trị cũng như gán giá trị cho playerMark

//



Khởi tạo biến private matrix để mô tả ma trận các ô trong bàn cờ dưới dạng nút và khởi tạo property cho matrix khi cần gọi giá trị cũng như gán giá trị cho matrix

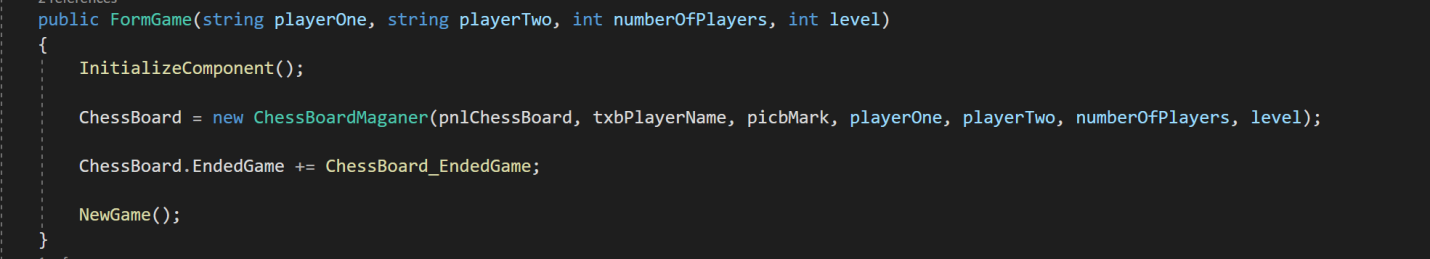
//

Khởi tạo biến private playTimeLine sau đó tạo properties PlayTimeLine để gán giá trị cho biến playTimeLine cũng như trả về giá trị của playTimeLine khi cần sử dụng.

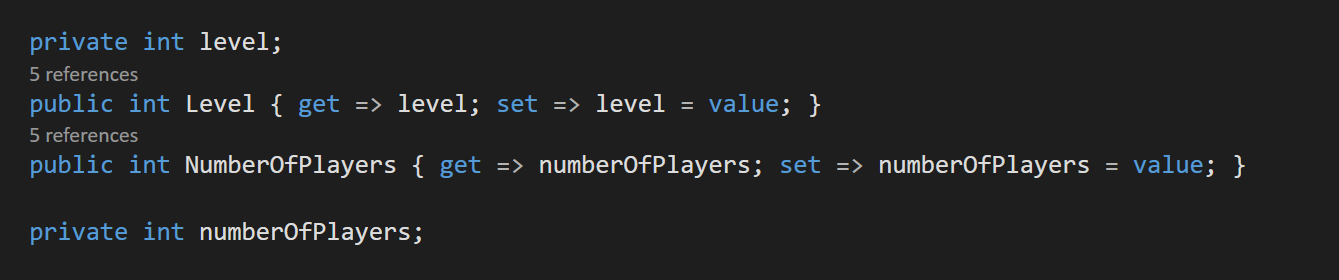
//



Khởi tạo biến private endedGame và properties dạng add and remove để thêm và bớt event nhằm mục đích kết thúc trò chơi. EndedGame được gọi tại Formgame

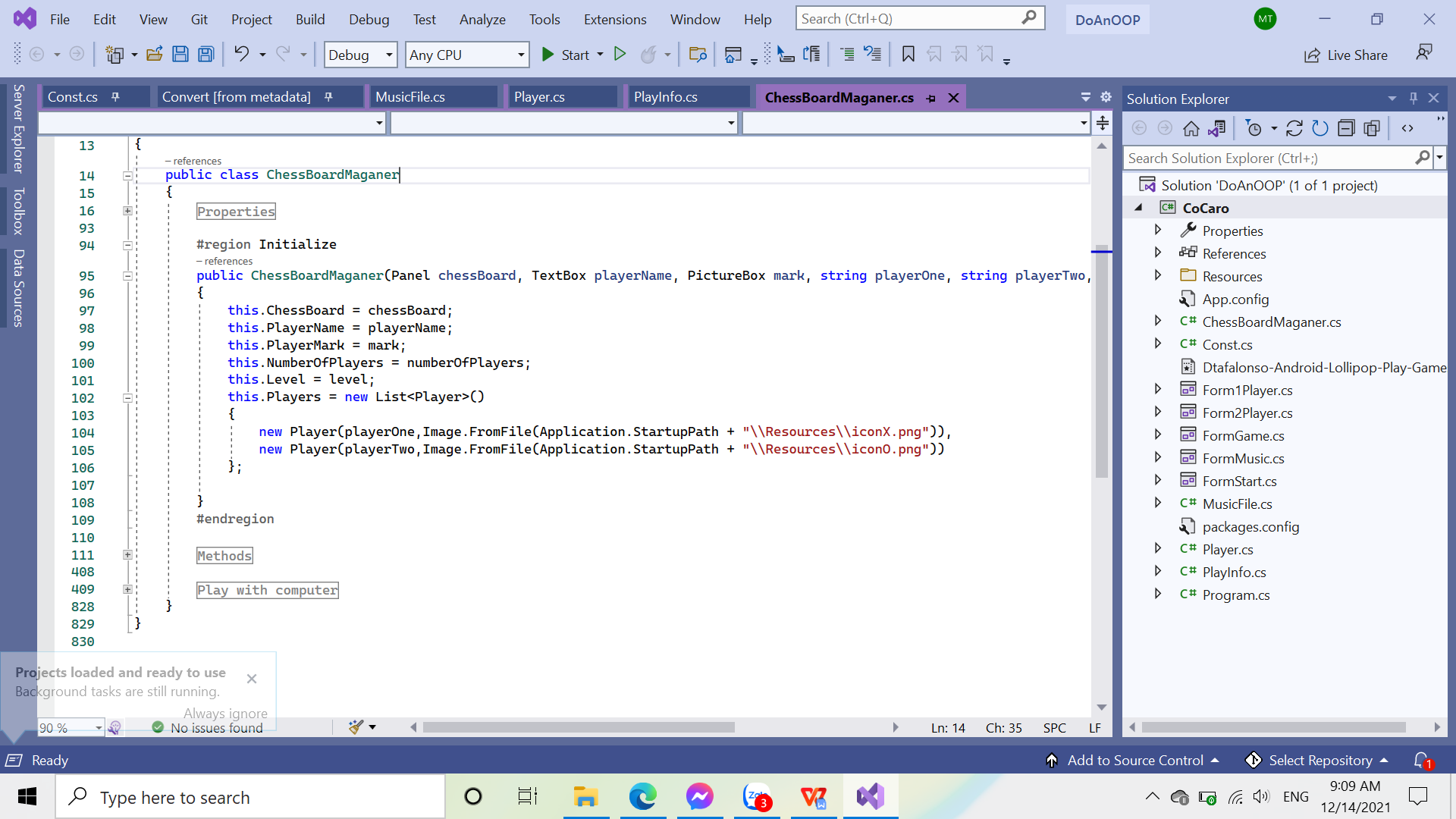


//



Tạo hai biến private level và numberOfPlayers sau đó tạo properties nhằm lấy giá trị cug như gán giá trị cho hai biến trên khi được gọi.

***3.3.2.2 Initialize***

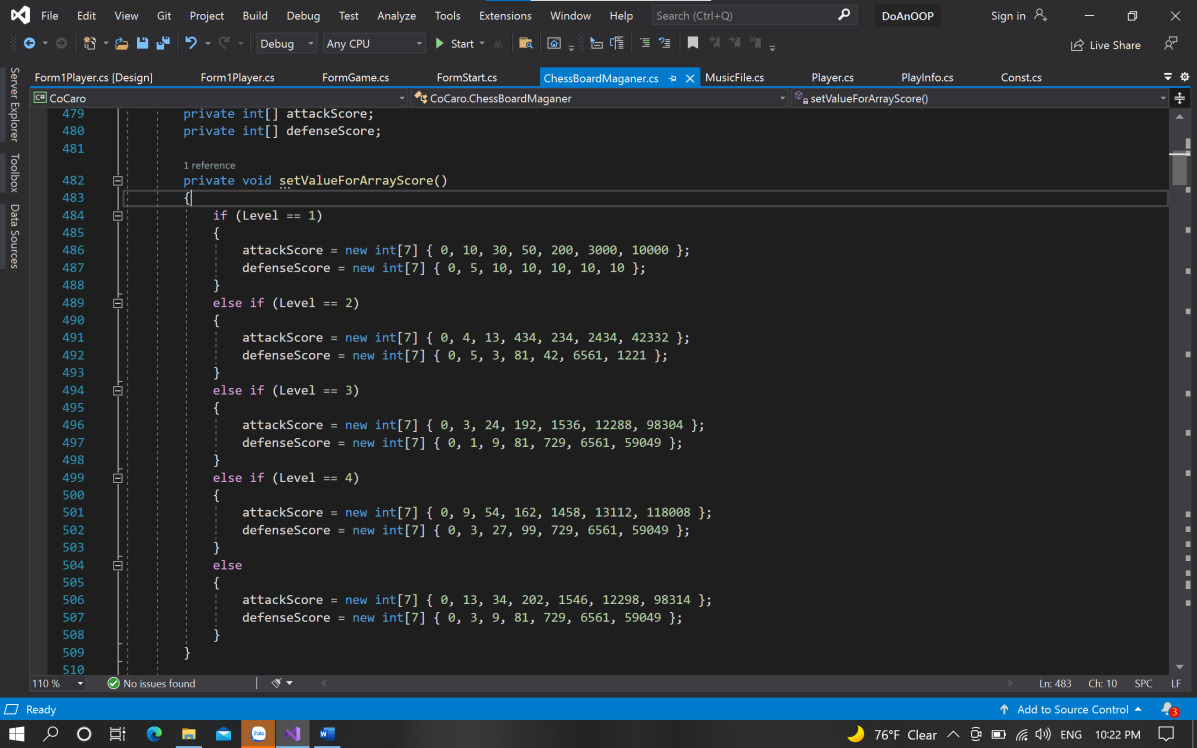
Thiết lập các giá trị ban đầu cho các biến 

***3.3.2.3 Methods***

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Phương thức | Mục đích | Tên file, số thứ tự dòng khai báo |
| 1 | drawChessBoard() | Tạo bàn cờ game Caro. | ChessBoardMaganer.cs (112-168) |
| 2 | btn\_Click(object sender, EventArgs e) | Chứa các cài đặt giúp người chơi có thể đánh O hoặc X vào bàn cờ game Caro. | ChessBoardMaganer.cs  (169-199) |
| 3 | endGame(int currentPlayer) | Gọi các hàm kiểm kết quả và Hiện thông báo kết quả trò chơi và người thắng cuộc. | ChessBoardMaganer.cs  (200-209) |
| 4 | EndGame() | Hiện thông báo kết thúc game. | ChessBoardMaganer.cs  (210-215) |
| 5 | Undo() | Tạo ra chức năng hoàn tác cho game. Xóa cả 2 nước đi gần nhất của cả 2 khi ở chế độ chơi với máy, xóa nước đi của người chơi hiện tại khi ở chế độ chơi 2 người. | ChessBoardMaganer.cs  (216-249) |
| 6 | isEndGame(Button btn) | Hàm dùng để kiểm tra điều kiện kết thúc của trò chơi. | ChessBoardMaganer.cs  (250-258) |
| 7 | GetChessPoint(Button btn) | Thực hiện lấy và xử lí dữ liệu từ ô vừa đánh. | ChessBoardMaganer.cs  (259-266) |
| 8 | isEndHorizontal(Button btn) | Kiểm tra điều kiện chiến thắng trên hàng ngang - khi trên một hàng có 5 kí hiệu cùng loại liên tiếp. | ChessBoardMaganer.cs  (267-294) |
| 9 | isEndVertical(Button btn) | Kiểm tra điều kiện chiến thắng trên cột dọc - khi trên một cột có 5 kí hiệu cùng loại liên tiếp. | ChessBoardMaganer.cs  (295-322) |
| 10 | isEndMain(Button btn) | Kiểm tra điều kiện chiến thắng trên đường chéo chính - khi trên một đường chéo chính có 5 kí hiệu cùng loại liên tiếp. | ChessBoardMaganer.cs  (323-356) |
| 11 | isEndSub(Button btn) | Kiểm tra điều kiện chiến thắng trên đường chéo phụ - khi trên một đường chéo phụ có 5 kí hiệu cùng loại liên tiếp. | ChessBoardMaganer.cs  (357-390) |
| 12 | isDrawGame() | Hàm chứa điều kiện để kiểm tra nếu như kết quả trò chơi là Hòa. | ChessBoardMaganer.cs  (391-400) |
| 13 | ChangePlayer() | Thay đổi hiển thị tên và mark của người chơi lượt hiện tại trên FormGame . | ChessBoardMaganer.cs  (401-406) |

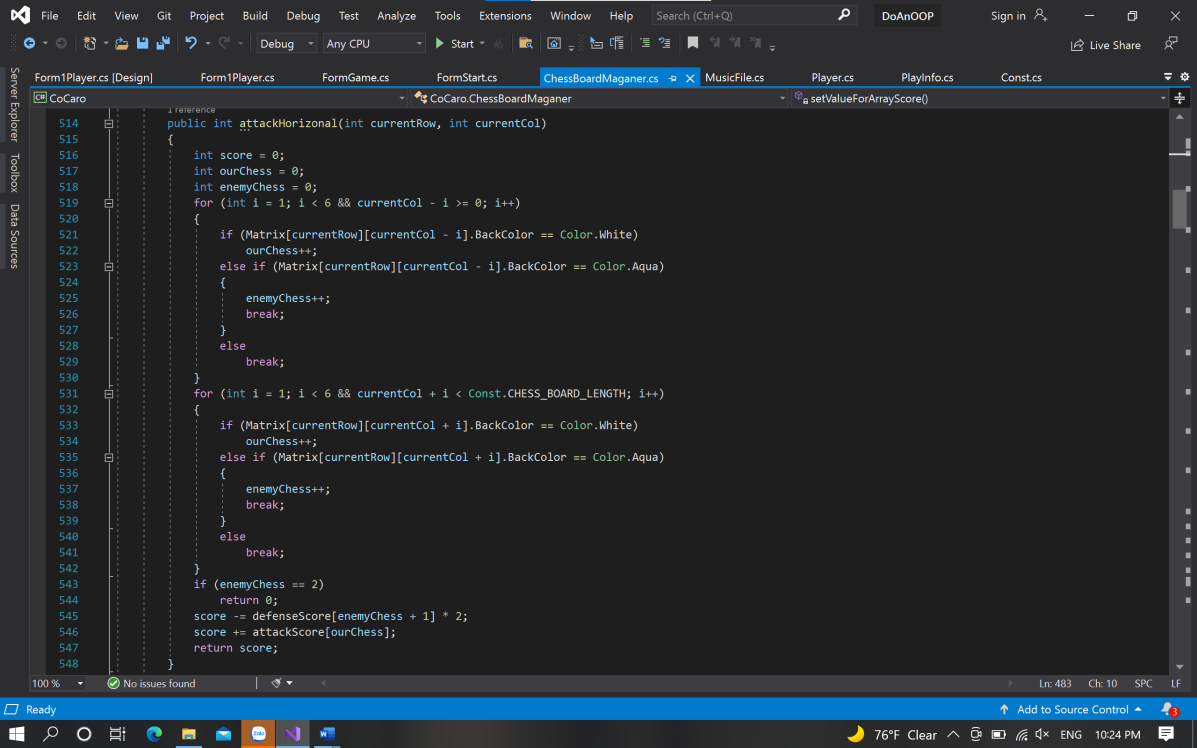
***3.3.2.4 Play with computer***

1. Set value for array score



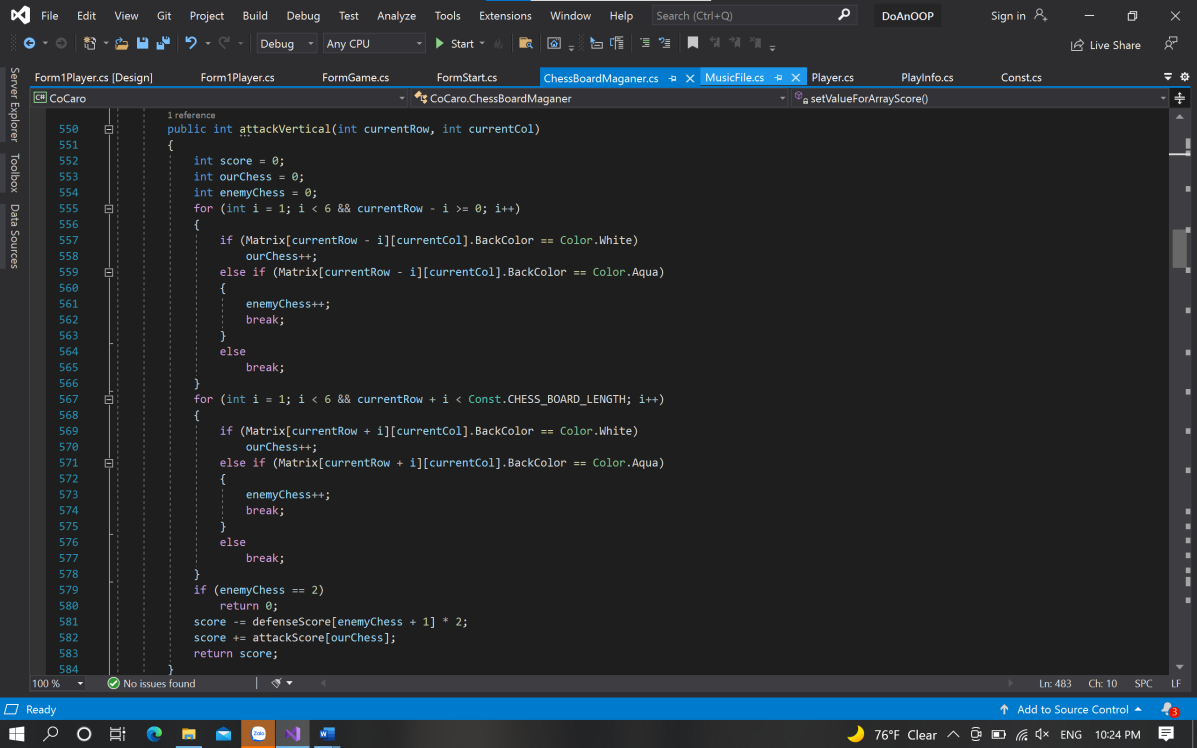
Với phần code trên chúng ta sẽ có thể lựa chọn độ khó của trò chơi theo 5 level khác nhau trong phần chơi cùng máy với các mức độ tấn công, phòng thủ khó dễ khác nhau của máy.

1. Attack horizonal



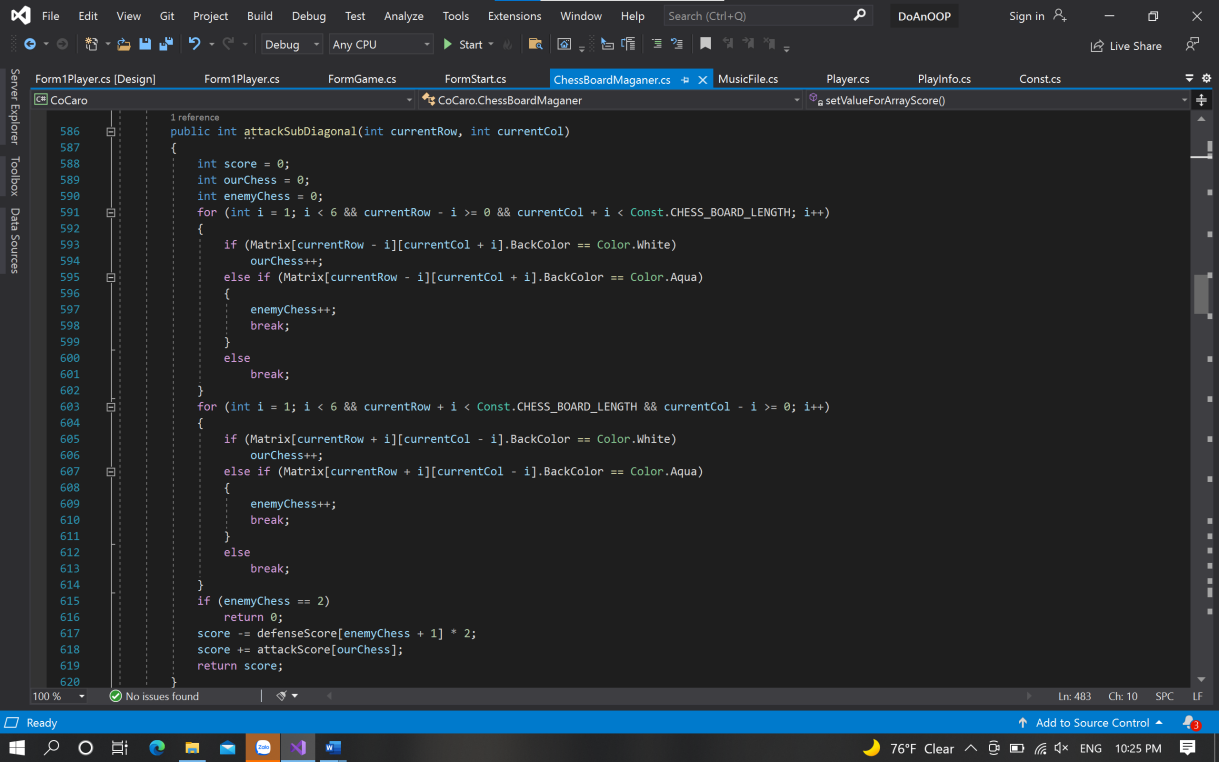
Với function tấn công hàng ngang trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên hàng của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách trừ đi giá trị defenseScore tại index thứ (tổng ô của máy cộng cho 1) nhân cho 2 và sau đó sẽ cộng với giá tị attackScore tại index thứ (tổng ô của người chơi đã chọn). Và sau đó trả về điểm số vừa tính được.

1. Attack vertical



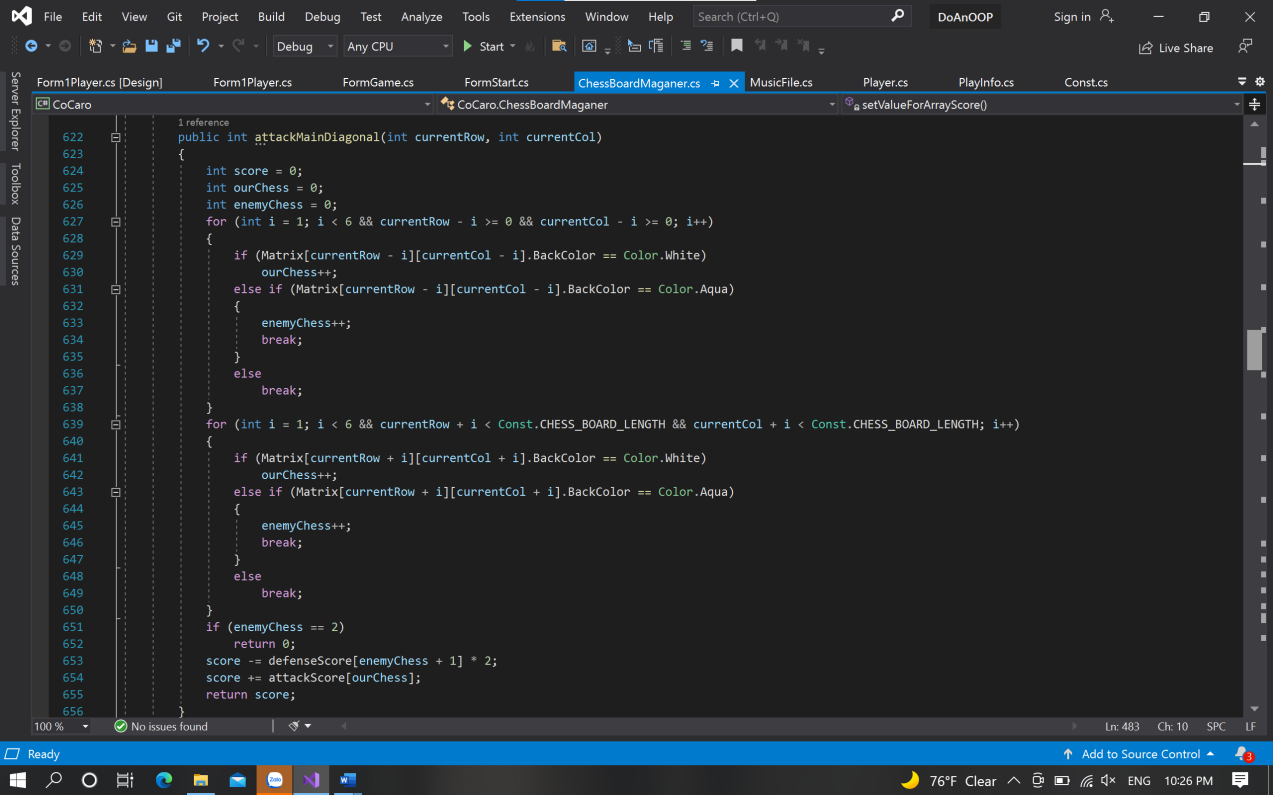
Với function tấn công hàng dọc trên chúng ta sẽ đếm tổng số ô đã được đánh của bản than người chơi và máy trên cột của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách trừ đi giá trị defenseScore tại index thứ (tổng ô của máy cộng cho 1) nhân cho 2 và sau đó sẽ cộng với giá tị attackScore tại index thứ (tổng ô của người chơi đã chọn). Và sau đó trả về điểm số vừa tính được.

1. Attack subDiagonal



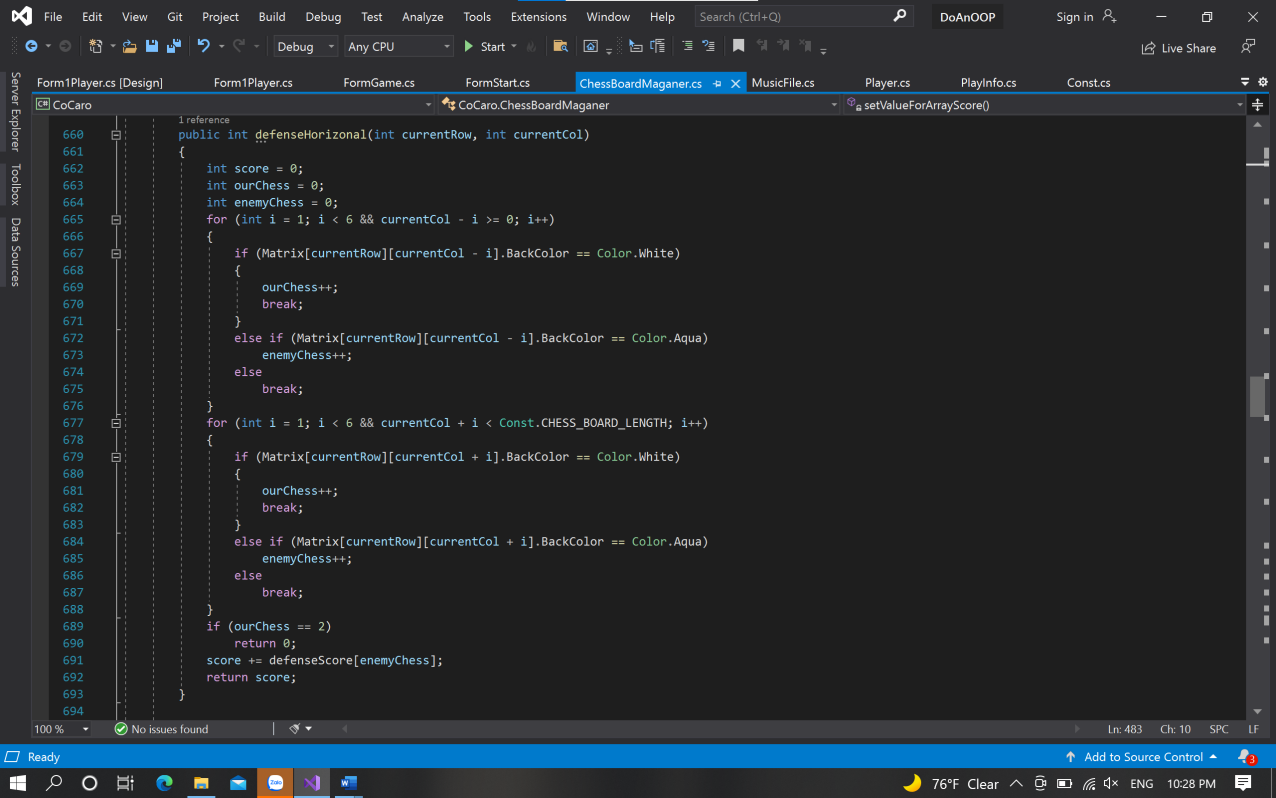
Với function tấn công đường chéo chính trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên đường chéo chính của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách trừ đi giá trị defenseScore tại index thứ (tổng ô của máy cộng cho 1) nhân cho 2 và sau đó sẽ cộng với giá tị attackScore tại index thứ (tổng ô của người chơi đã chọn). Và sau đó trả về điểm số vừa tính được.

1. Attack mainDiagonal



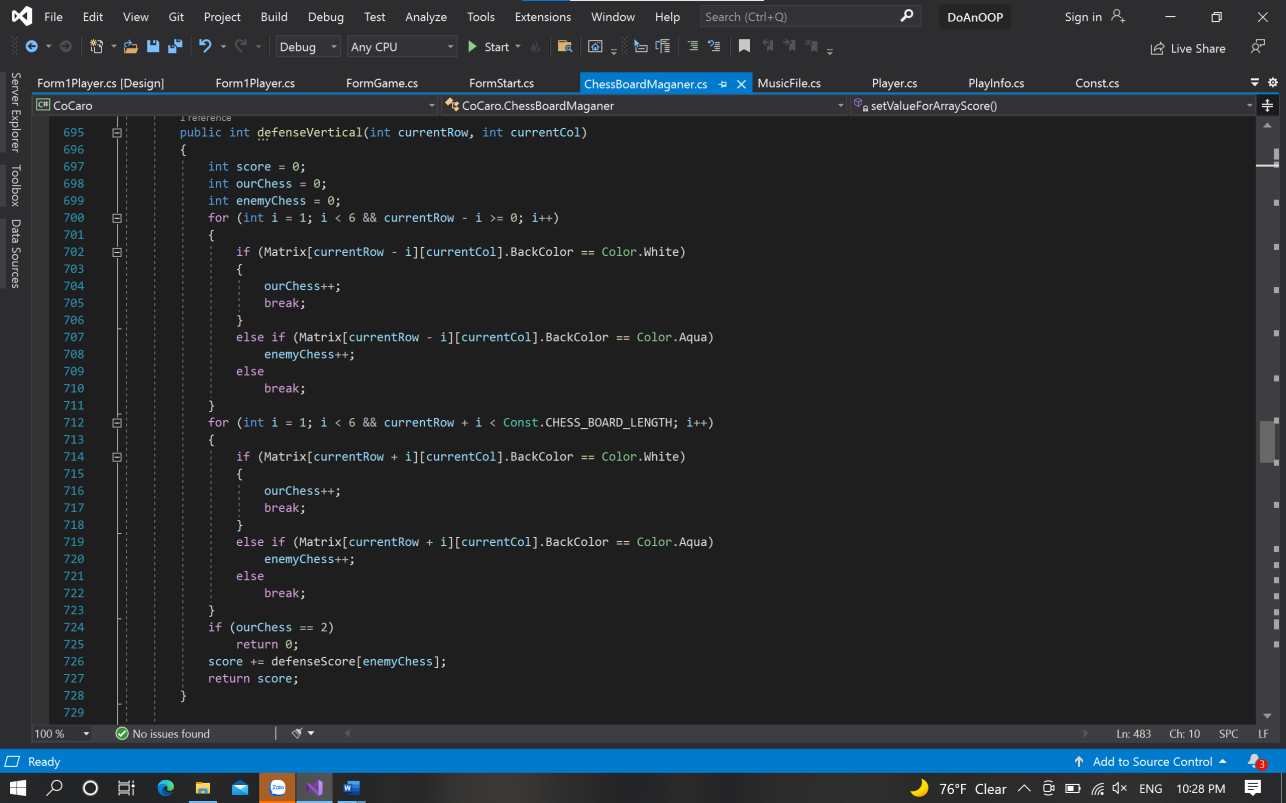
Với function tấn công đường chéo phụ trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên đường chéo phụ của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách trừ đi giá trị defenseScore tại index thứ (tổng ô của máy cộng cho 1) nhân cho 2 và sau đó sẽ cộng với giá tị attackScore tại index thứ (tổng ô của người chơi đã chọn). Và sau đó trả về điểm số vừa tính được.

1. Defense horizonal



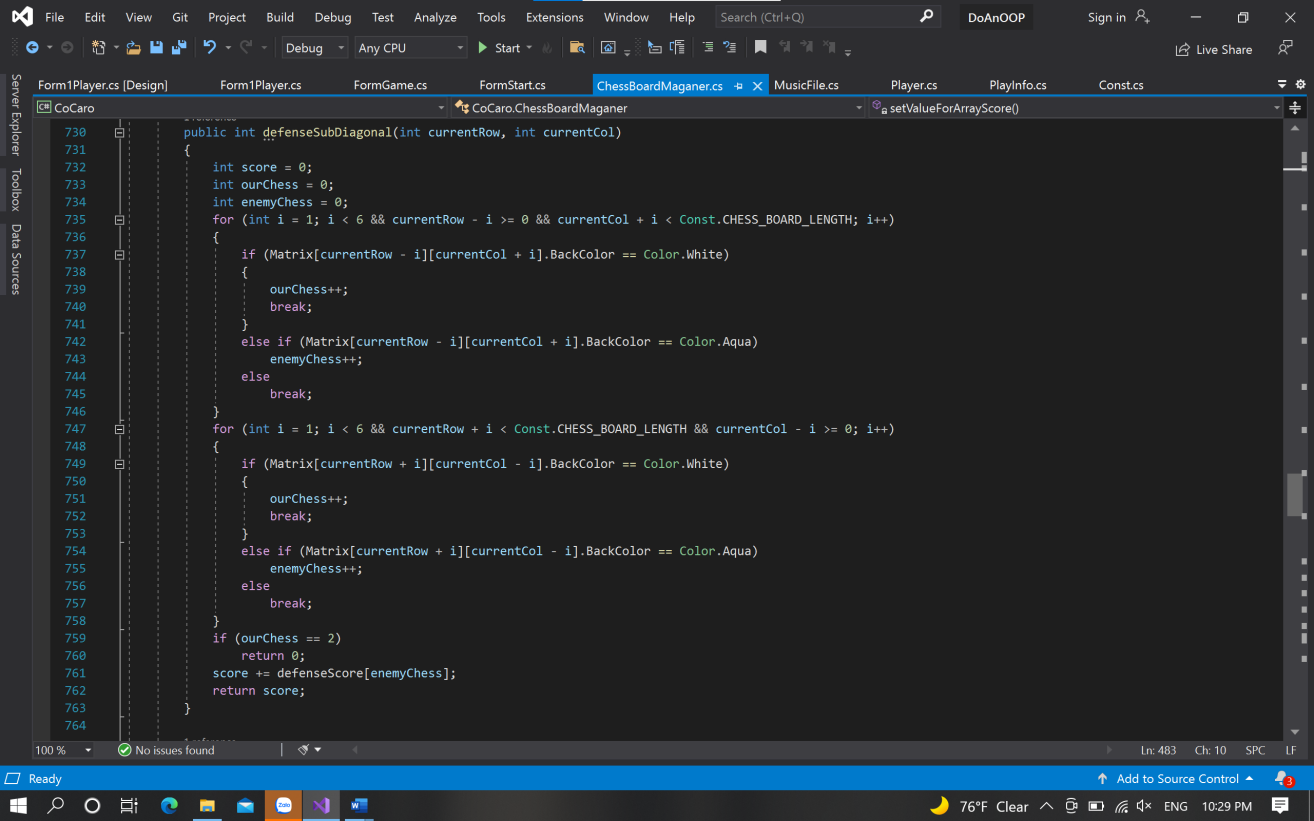
Với function phòng thủ hàng ngang trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên hàng ngang của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách cộng đi giá trị defenseScore tại index thứ (tổng ô của máy cộng). Và sau đó trả về điểm số vừa tính được.

1. Defense vertical



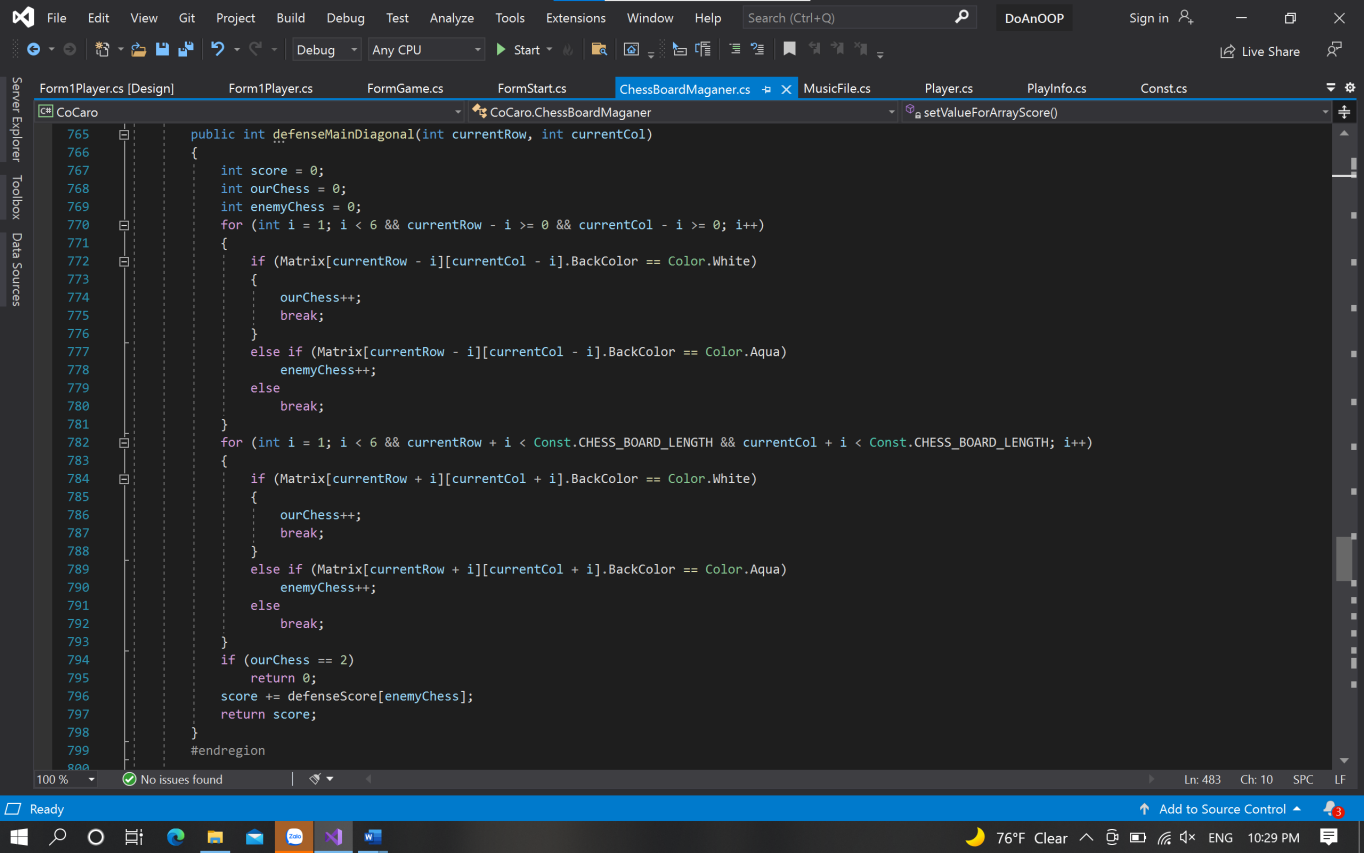
Với function phòng thủ hàng dọc trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên hàng dọc của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách cộng đi giá trị defenseScore tại index thứ (tổng ô của máy cộng). Và sau đó trả về điểm số vừa tính được.

1. Defense subDiagonal



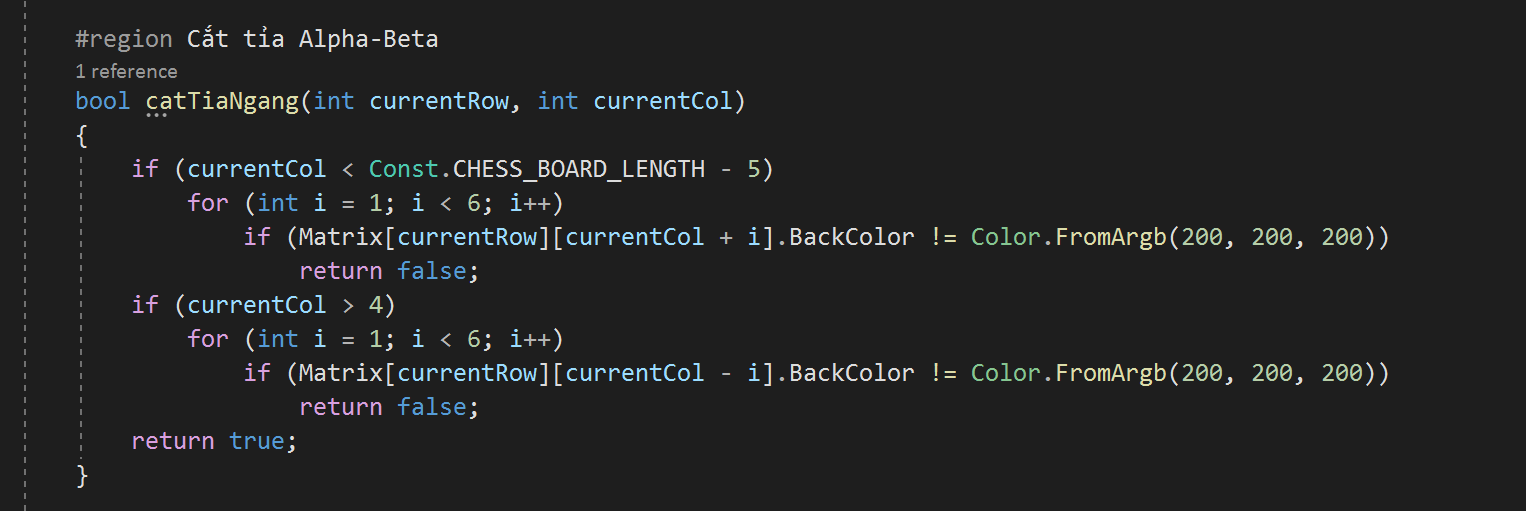
Với function phòng thủ đường chéo phụ trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên đường chéo phụ của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách cộng đi giá trị defenseScore tại index thứ (tổng ô của máy cộng). Và sau đó trả về điểm số vừa tính được.

1. Defense mainDiagonal



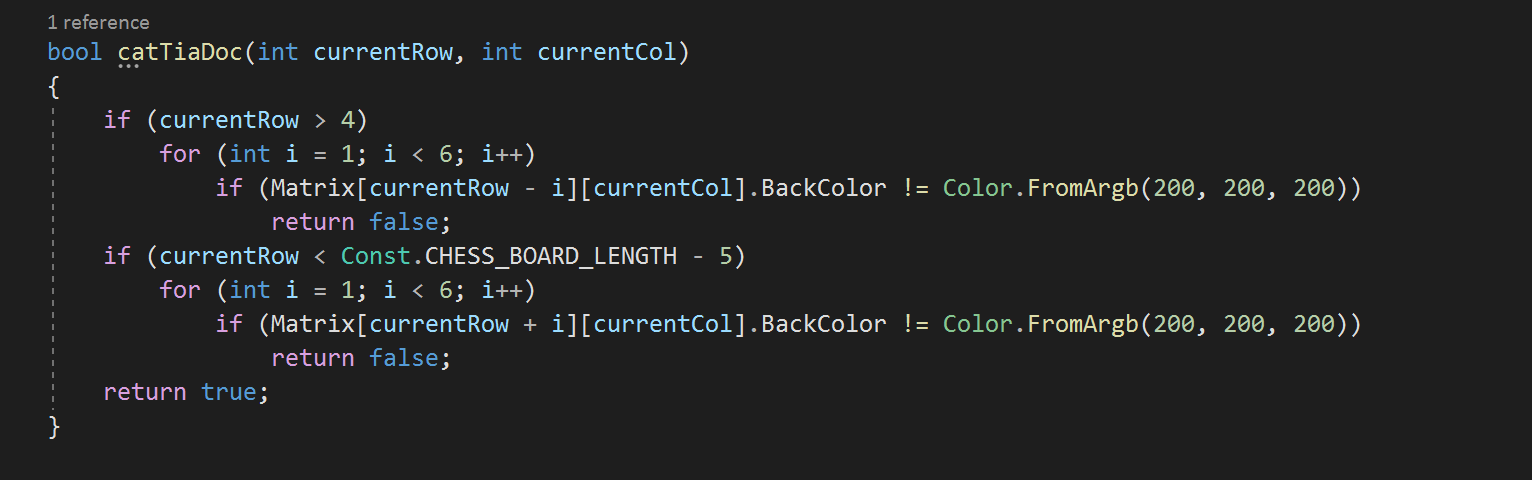
Với function phòng thủ đường chéo chính trên chúng ta sẽ đếm tổng số ô đã được đánh của bản thân người chơi và máy trên đường chéo chính của lượt chơi cuối cùng. Nếu số nước đi của máy chỉ là 2 sẽ trả về mức điểm là 0 còn ngược lại điểm sẽ được tính bằng cách cộng đi giá trị defenseScore tại index thứ (tổng ô của máy cộng). Và sau đó trả về điểm số vừa tính được.

1. Cắt tỉa ngang



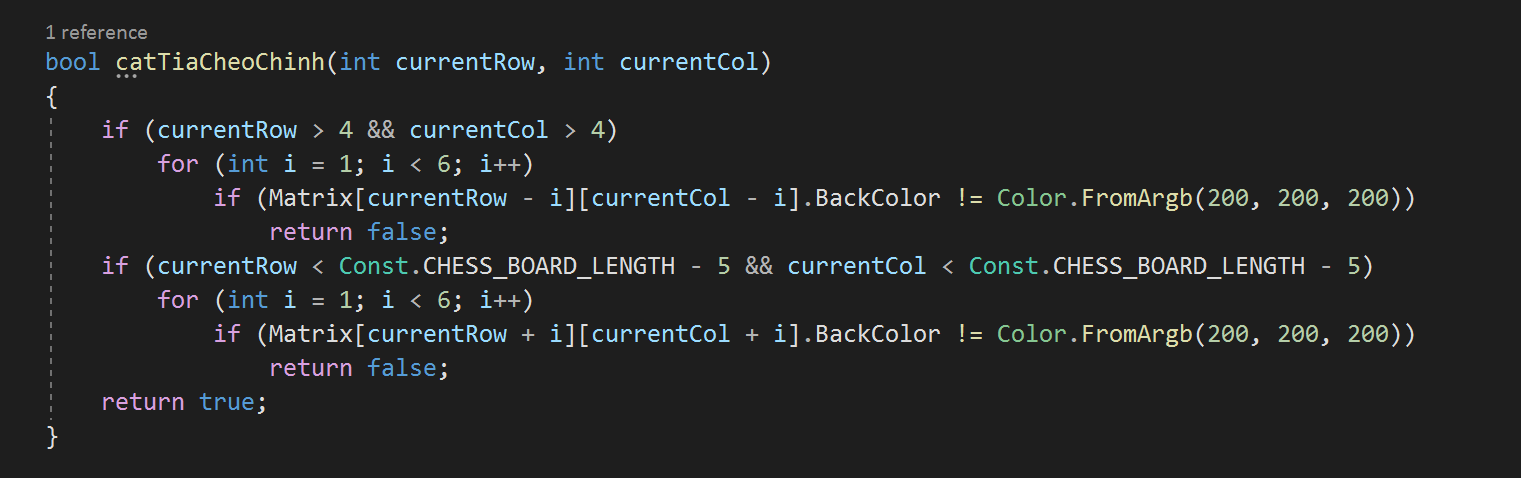
Với function cắt tỉa ngang sẽ kiểm tra ô hiện tại có địa chỉ cột là nhỏ hơn 15 hoặc lớn hơn 4 hay không sau đó kiểm tra các ô trên cùng hàng ngang hiện tại đã được đánh từ trước hay chưa nếu một trong các ô ở phạm vi 6 ô kế tiếp đã được đánh rồi sẽ trả về giá trị false, ngược lại sẽ trả về giá trị true

1. Cắt tỉa dọc



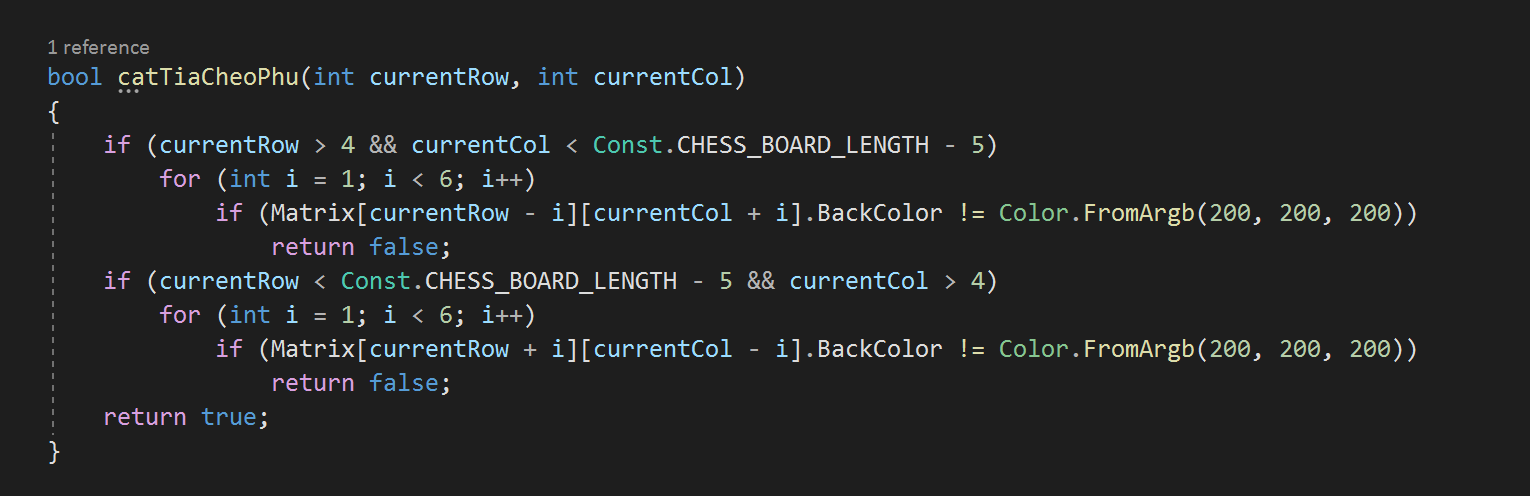
Tương tự như function cắt tỉa ngang, cắt tỉa dọc cũng kiểm tra các ô hiện tại có địa chỉ dòng là lớn hơn 4 hoặc nhỏ hơn 15 hay không sau đó kiểm tra các ô trên cùng dòng hiện tại nếu một trong các ô ở phạm vi 6 ô kế tiếp đã được đánh sẽ trả về false và ngược lại trả về giá trị true.

1. Cắt tỉa chéo chính



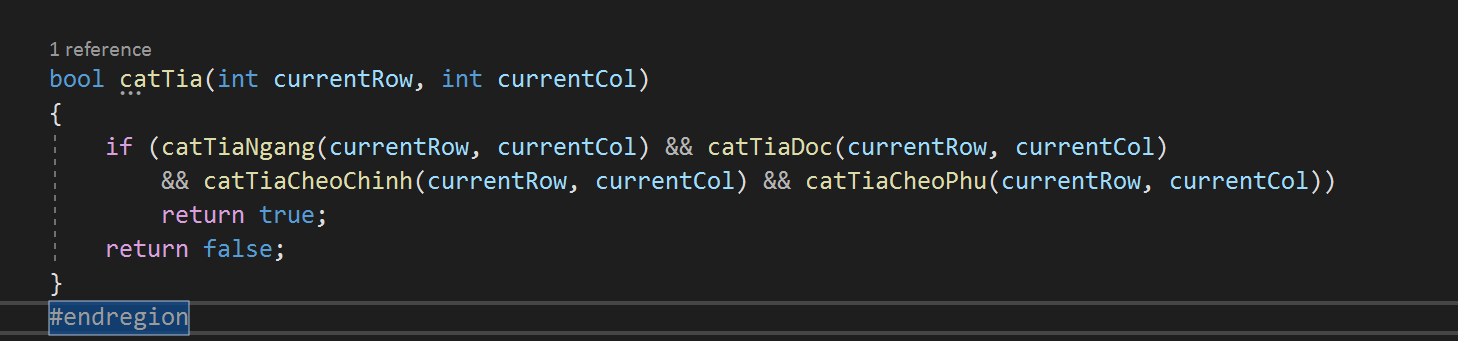
Tương tự như function cắt tỉa ngang, cắt tỉa chéo chính cũng kiểm tra các ô hiện tại có địa chỉ dòng và cột là lớn hơn 4 hoặc nhỏ hơn 15 hay không sau đó kiểm tra các ô trên cùng đường chéo chính hiện tại nếu nếu một trong các ô ở phạm vi 6 ô kế tiếp đã được đánh sẽ trả về false và ngược lại trả về giá trị true

1. Cắt tỉa chéo phụ



Tương tự như function cắt tỉa ngang, cắt tỉa chéo phụ cũng kiểm tra các ô hiện tại có địa chỉ là dòng lớn hơn 4 và cột nhỏ hơn 15 hoặc cột lớn hơn 4 và dòng nhỏ hơn 15 hay không sau đó kiểm tra các ô trên cùng đường chéo chính hiện tại nếu một trong các ô ở phạm vi 6 ô kế tiếp đã được đánh sẽ trả về false và ngược lại trả về giá trị true.

1. Cắt tỉa



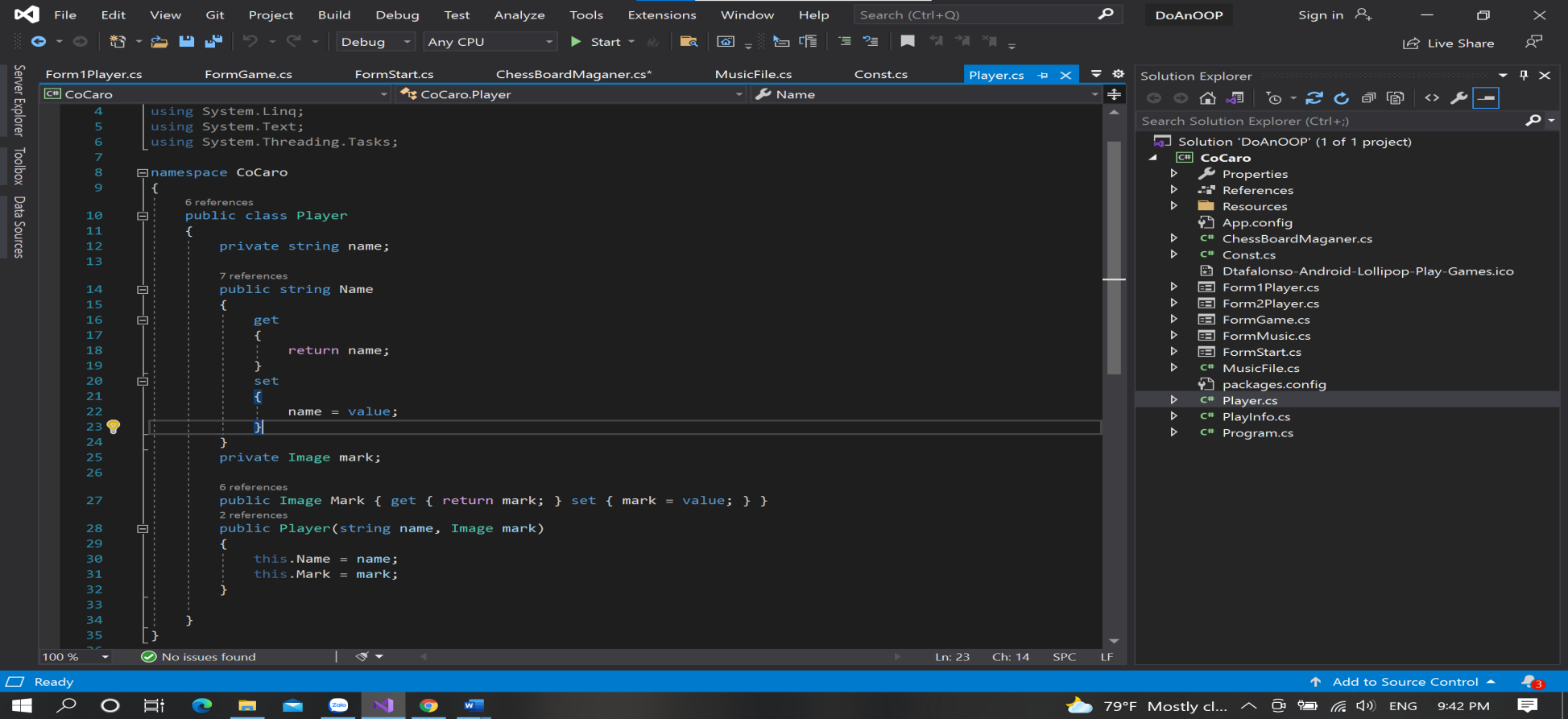
Hàm cắt tỉa sẽ gọi các hàm cắt tỉa ngang, dọc, chéo chính, chéo phụ nếu tất cả đều trả về giá trị true sẽ trả về giá trị true và ngược lại false

1. Get coordinate of computer chess



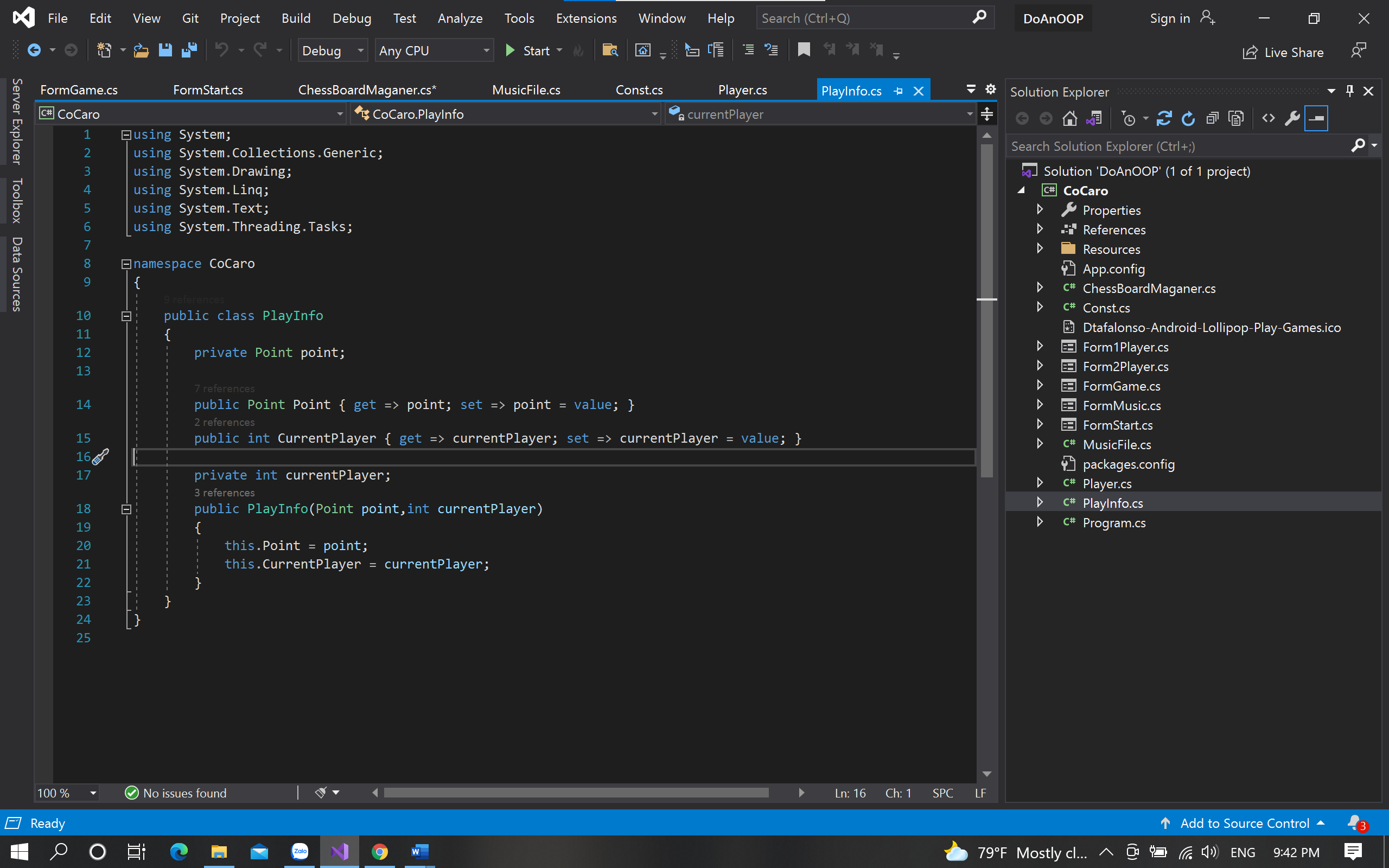
Hàm sẽ kiểm tra người chơi có chọn mode chới với máy hay không và đây có phải là lượt đầu tiên trong trò chơi hay không. Nếu có sẽ thực hiện tạo random một tọa độ x,y khi mà tọa độ x,y trong ma trận trò chơi chưa được chọn, còn mang giá trị null. Nếu ngược lại, sẽ kiểm tra trên toàn bộ các ô của ma trận game caro nếu ô đó chưa được chọn và hàm cắt tỉa trả về giá trị null tại vị trí đó thì sẽ tính điểm tấn công trên các hàng, cột, chéo chính, chéo phụ công lại cũng tương tự với điểm phòng thủ. Điểm tạm thời (tempScore) được tính bằng cách điểm tấn công chia cho điểm phòng thủ nếu như điểm tấn công cao hơn điểm phòng thủ. Từ đây ta sẽ tìm ra được một ô có điểm tạm thời cao nhất gán vào biến điểm cao nhất (maxScore) và vị trí của ô hiện tại được lưu vào tọa độ x,y. Kết thúc hàm trả về cặp tọa độ x,y.

**3.3.3 Player**



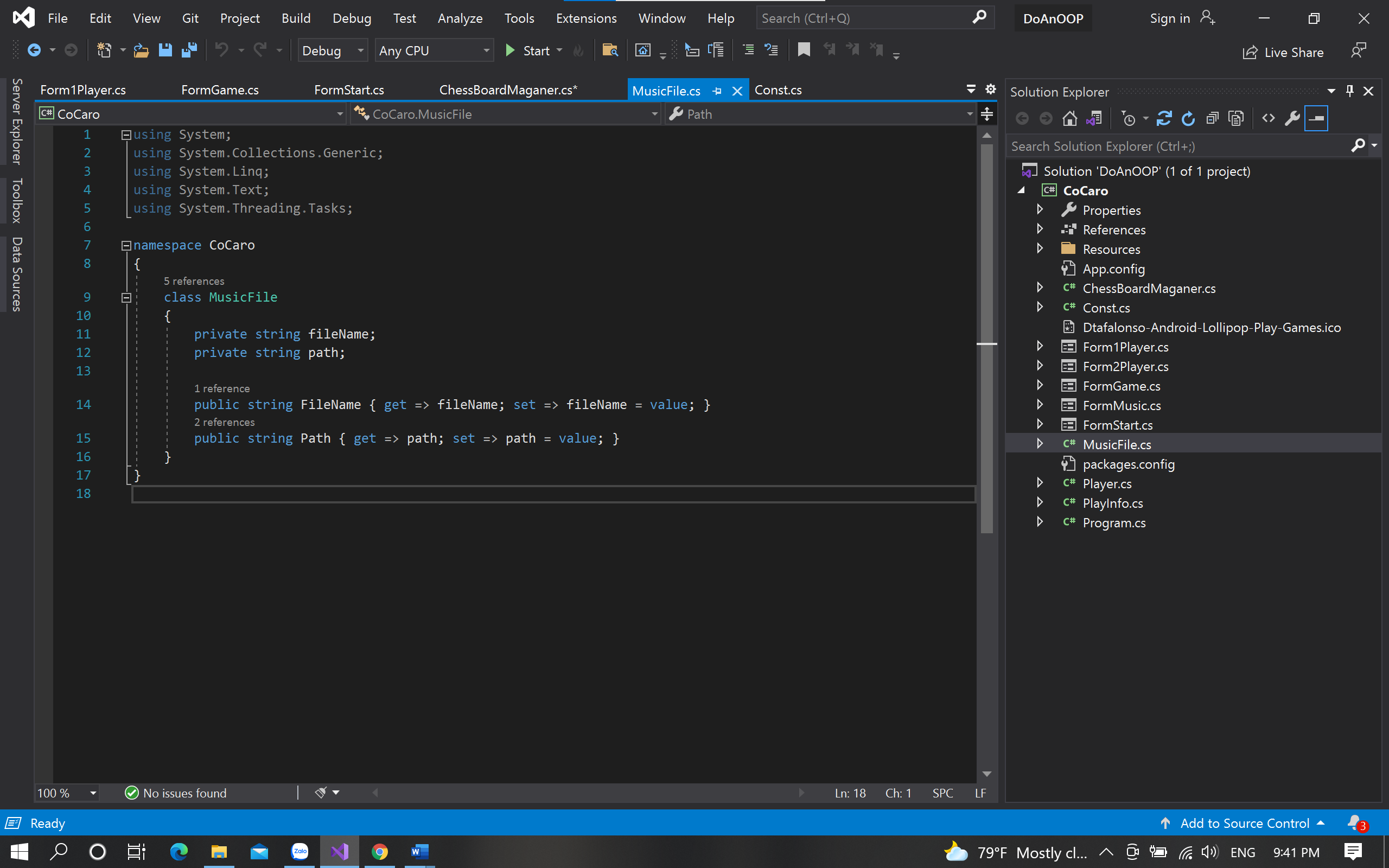
Class được tạo với 2 properties trả lại tên người chơi, đánh dấu các ô đã được một trong hai phe chọn khi chơi game Caro của chúng ta, và môt hàm khởi tạo (constructor) tên người chơi và điểm số

**3.3.4 PlayInfo**



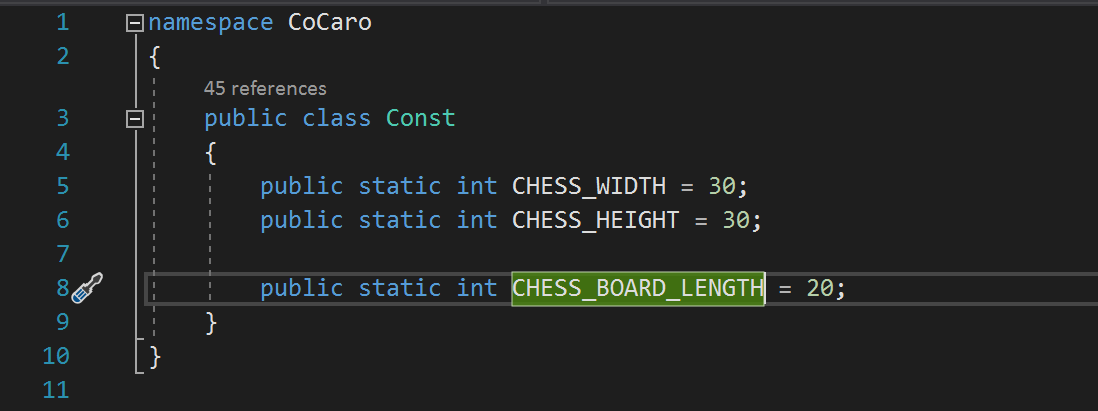
Class được tạo với 2 properties điểm số và người chơi chơi hiện tại, bên cạnh đó là một khởi tạo (constructor) điểm số và người chơi hiện tại

**3.3.5 MusicFile**



Class được tạo với 2 properties nhằm lấy tên và đường dẫn của file nhạc người dùng muốn phát trong cửa sổ phát nhạc của game

**3.3.5 Const**



Class được tạo ra để đặt các giá trị cố định cho các biến, mảng trong trò chơi

**[CHƯƠNG 4: KẾT LUẬN](#_Toc44455501)**