KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH**

**HỌC KỲ I, NĂM HỌC 2023-2024**

**VIẾT CHƯƠNG TRÌNH TRÒ CHƠI TIC-TAC-TOE**

*Giáo viên hướng dẫn:*

TS. Nguyễn Trần Diễm Hạnh

*Sinh viên thực hiện:*

Họ tên: Huỳnh Khải Vinh

MSSV: 110121245

Lớp: DA21TTA

***Trà Vinh, tháng 1 năm 2024.***

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH**

**HỌC KỲ I, NĂM HỌC 2023-2024**

**VIẾT CHƯƠNG TRÌNH TRÒ CHƠI TIC-TAC-TOE**

*Giáo viên hướng dẫn:*TS. Nguyễn Trần Diễm Hạnh

*Sinh viên thực hiện:*

Họ tên: Huỳnh Khải Vinh

MSSV: 110121245

Lớp: DA21TTA

***Trà Vinh, tháng 1 năm 2024***

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành các thầy,cô của Trường Đại học Trà Vinh, đặc biệt là cô Nguyễn Trần Diễm Hạnh đã tận tình hướng dẫn, truyền đạt kiến thức cho em trong suốt quá trình thực hiện đố cơ sở ngành. Với vốn kiến thức được tiếp thu trong quá trình học không chỉ là nền tảng cho quá trình nghiên cứu bài báo cáo mà nó còn là hành trang quý báu để em áp dụng vào thực tế một cách vững chắc và tự tin hơn.

Do còn hạn chế về kiến thức cũng như những kinh nghiệm thực tế cho nên không tránh khỏi được những sai sót trong quá trình tìm hiểu và trình bày rất mong nhận được sự đóng góp ý kiến của thầy để bài báo cáo của em được hoàn chỉnh hơn. Sau cùng, em xin kính chúc thầy thật nhiều sức khoẻ, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau. Xin chân thành cảm ơn!

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN 9](#_Toc155528499)

[CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT 10](#_Toc155528500)

[2.1 Lập trình C# và Windows Forms 10](#_Toc155528501)

[2.1.1 Giới thiệu về ngôn ngữ C# 10](#_Toc155528502)

[2.1.2 Lập trình trên ngôn ngữ C# 10](#_Toc155528503)

[2.1.3 Windows Forms 10](#_Toc155528504)

[2.2 Trò chơi tic-tac-toe 11](#_Toc155528505)

[2.2.1 Lịch sử và nguồn gốc của trò chơi tic-tac-toe 11](#_Toc155528506)

[2.2.2 Luật chơi 11](#_Toc155528507)

[2.2.3 Chiến thuật 11](#_Toc155528508)

[2.3 Thuật toán minimax 12](#_Toc155528509)

[2.3.1 Lý thuyết 12](#_Toc155528510)

[2.3.2 Ví dụ trên cây trò chơi 12](#_Toc155528511)

[2.3.3 Áp dụng vào trò chơi tic-tac-toe 15](#_Toc155528512)

[CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU 17](#_Toc155528517)

[3.1 Yêu cầu của chương trình 17](#_Toc155528518)

[3.2 Thiết kế giao diện 17](#_Toc155528519)

[3.3 Cài đặt thuật toán minimax 18](#_Toc155528521)

[3.3.1 Mã giả của thuật toán minimax 18](#_Toc155528522)

[3.3.2 Lưu đồ thực hiện giải thuật 19](#_Toc155528523)

[3.3.3 Độ phức tạp và số trạng thái tìm kiếm của thuật toán minimax 20](#_Toc155528524)

[3.4 Cài đặt thuật toán kiểm tra thắng, hoà 20](#_Toc155528525)

[3.4.1 Tìm kiếm trên hàng ngang 20](#_Toc155528526)

[3.4.2 Tìm kiếm theo cột dọc 21](#_Toc155528527)

[3.4.3 Tìm kiếm theo các đường chéo 21](#_Toc155528528)

[3.4.3 Xác định bàn cờ đã đầy 22](#_Toc155528529)

[3.4.4 Phân tích độ phức tạp của các thuật toán 22](#_Toc155528531)

[CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU 23](#_Toc155528532)

[4.1 Kết quả đạt được 23](#_Toc155528533)

[4.2 Hạn chế 24](#_Toc155528534)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 26](#_Toc155528535)

[5.1 Kết luận 26](#_Toc155528536)

[5.2 Hướng phát triển 26](#_Toc155528537)

[DANH MỤC TÀI LIỆU THAM KHẢO 27](#_Toc155528538)

**TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH**

Đồ án “Viết chương trình trò chơi tic-tac-toe” sẽ tập trung nghiên cứu, tìm hiểu các cách thức viết và cài đặt trò chơi tic-tac-toe trong đó một người chơi sẽ đấu với máy tính. Chương trình sẽ được viết bằng ngôn ngữ C#, một ngôn ngữ lập trình hướng đối tượng, sửa dụng thư viện lớp đồ hoạ Windows Forms để tạo giao diện người dùng. Người chơi máy sẽ được cài đặt bằng thuật toán minimax, một thuật toán tìm kiếm cho phép người chơi máy tìm và xác định các bước đi thích hợp nhất. Mục tiêu của đồ án sẽ là một sản phẩm trò chơi tic-tac-toe trong đó một người chơi thật sẽ đấu với một người chơi máy trên các bàn cờ có kích thước là 3x3 (3 hàng, 3 cột), 5x5, 7x7 do người chơi thật tuỳ ý lựa chọn. Người chơi thật cũng có thể tuỳ ý lựa chọn ký tự X hay O theo ý muốn của mình. Trò chơi sẽ kết thúc nếu như một trong hai người chơi đi được 3 ô giống nhau hoặc khi bàn cờ đã đầy, tức là không còn nước đi nào nữa.

**MỞ ĐẦU**

Hiện nay, với sự phát triển của khoa học-kĩ thuật, máy tính đã là mộ phần không thể thiếu trong cuộc sống của nhiều người. Đi kèm với sự phát triển của máy tính là sự nở rộ của rất nhiều ứng dụng với các tính đa dạng, phục vụ cho rất nhiều ngành nghề, lĩnh vực trong xã hội. Một trong các ứng dụng được phát triển mạnh mẽ là các ứng dụng trò chơi điện tử. Các trò chơi này, đúng như tên gọi, thường được phát triển với mục đích giải trí cho con người. Ngoài mục tiên giải trí các trò chơi điện tử này có thể sử dụng vào công việc giáo dục như các trò chơi luyện đánh chữ, đố vui, cờ vua,… Đa số các trò chơi điện tử được sử dụng vào mục đích giác dục được dựa trên các trò chơi thật phổ biến. Một trong những trò chơi như vậy, và cũng là một trong các trò chơi điện tử đầu tiên, là trò chơi tic-tac toe.

Tic-tac-toe (hay còn gọi là cờ X-O) là một trò chơi phổ biến với luật chơi khá đơn giản. Trò chơi có thể được chơi với nhiều hình thức khác nhau và có nhiều phiên bản đa dạng. Mục tiêu của Đồ án này sẽ xây dựng và cài đặt một chương trình trò chơi trong đó một người chơi thật sẽ đấu với một người chơi máy cho đến khi có một người thắng hoặc cả hai hoà. Đồ án sẽ tập trung nghiên cứu về cách tạo dựng giao diện cho người chơi có thể tương tác với máy, cài dặt thuật toán để người chơi máy có thể tự chọn nước đi của mình.

# CHƯƠNG 1: TỔNG QUAN

**Về lập trình C# và Windows Forms:** C# là một ngôn ngữ lập trình bậc cao được phát triển bởi Microsoft vào năm 2000. C# là một ngôn ngữ đa năng bao gồm lập trình cấu trúc, lập trình hướng đối tượng, lập trình hướng sự kiện,…

Windows Forms là một lớp thư viện đồ hoạ được phát triển bởi Microsoft vào năm 2003, là một phần trong .NET framework của hãng. Windows Forms cung cấp một nền tảng để viết các chương trình cho máy tính để bàn, máy tính xách tay và máy tính bảng. Một ứng dụng Windows Forms là một ứng dụng hướng sự kiện, tức là ứng dụng sẽ chỉ hoạt động khi có một sự kiện được gọi, như khi người dùng điền vào một hộp văn bản hay nhấn vào một nút nào đó.

Cả ngôn ngữ C# và thư viện Windows Forms đều được hỗ trợ trên phần mềm Visual Studio. Visual Studio là một môi trường phát triển tích hợp cũng được phát triển bởi Microsoft được sử dụng để phát triển các chương trình như website, dịch vụ web, ứng dụng trên điện thoại,…

**Về trò chơi tic-tac-toe:** trò chơi tic-tac-toe (còn được gọi là cờ X-O) là một trò chơi hai người chơi trong đó mỗi người chơi sẽ lần lược điều chữ X hoặc O vào một ô trống trong một bảng thường có kích thước là 3x3. Một người chơi sẽ thắng nếu như có một hàng, cột hoặc đường chéo có cả 3 chữ X hoặc O.

Tic-tac-toe có khá nhiều phiên bản khác nhau, như phiên bản cờ ca-rô trong đó người chơi cần thắng 5 trên một dòng và không giới hạn bàn cờ.

**Về thuật toán minimax:** Minimax là một cách ra quyết định được dùng trong trí tuệ nhân tạo, lý thuyết trò chơi, thống kê,… để giảm (minimizing) tối đa các tổn thất có thể có hoặc trường hợp xấu nhât. Ngược lại, nó cũng có thể dùng để tăng (maximizing) tối đa các lợi ích hay trường hợp có lợi nhất.

Thuật toán minimax là một cách áp dụng lý thuyết minimax trong lý thuyết trò chơi. Thuật toán minimax là một thuật toán đệ quy tìm kiếm, được sử dụng để tìm nước đi tiếp theo trong một trò chơi hai người chơi. Một giá trị sẽ được gán cho mỗi trạng thái của trò chơi, giá trị này cho biết nước đi đó là tốt hay xấu. Khi đến lược của một người chơi A bất kì, A sẽ kiểm tra giá trị này và tìm ra nước đi phù hợp.

# CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

## Lập trình C# và Windows Forms

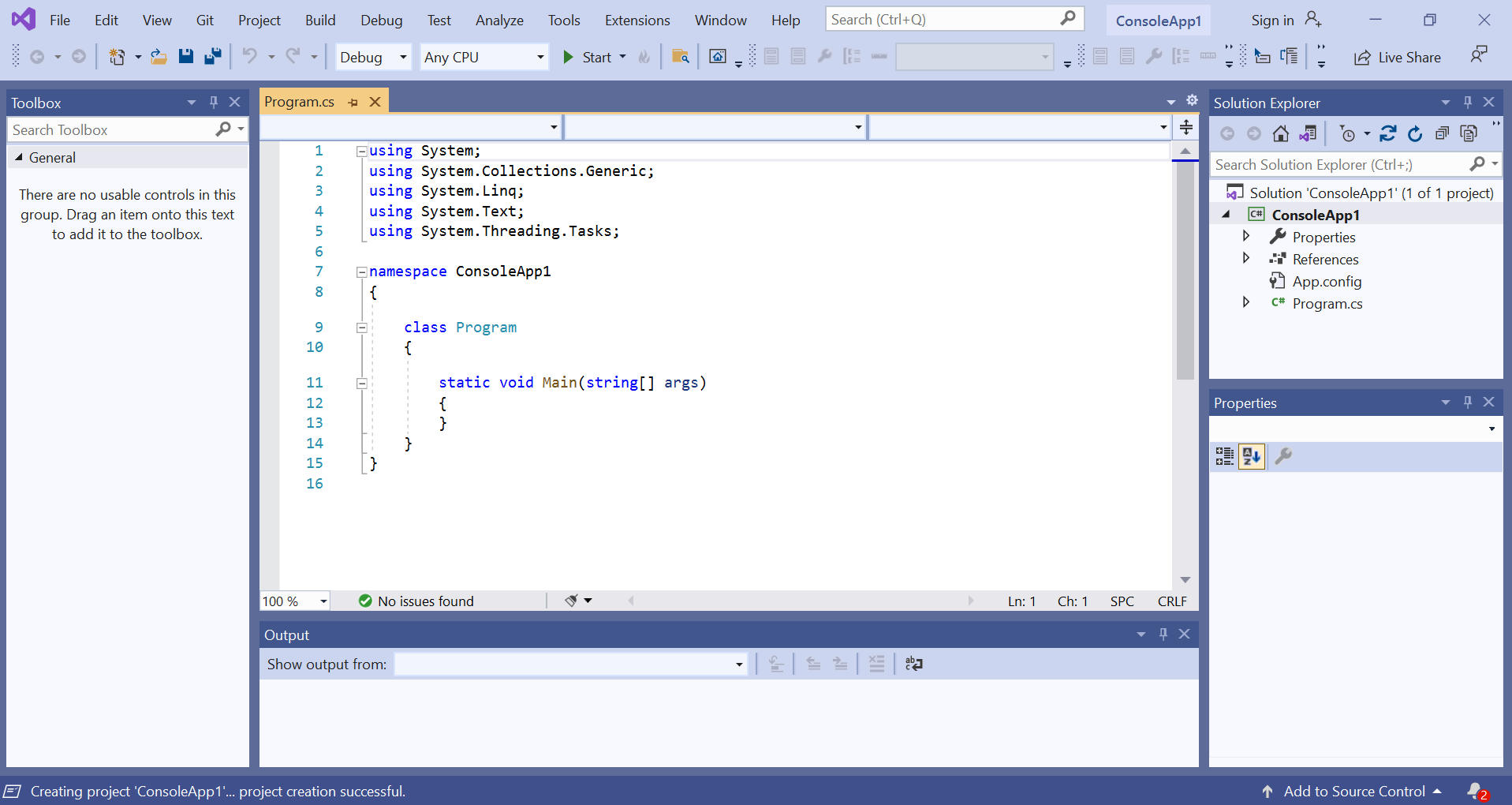
### Giới thiệu về ngôn ngữ C#

C# là một ngôn ngữ lập trình bậc cao được thiết kế Anders Hejlsberg thuộc tập đoàn Microsoft vào năm 2000. C# được phát triển để viết các thư viện lớp của .NET Framework và cả hai được phât hành cùng lúc, cùng với Visual Studio.

C# được phát triển với mục tiêu là một ngôn ngữ hiện đại, đơn giản, hướng đến nhiền mục đích sử dụng và là một ngôn ngữ lập trình hướng đối tượng.

C# được sử dụng trong rất nhiều lĩnh vực khác nhau, từ phát triển web backend như ASP .NET MVC, ASP .NET core, …, phát triển web fontend như Uno platform, …, đến phát triển các ứng dụng trên desktop như Winform, Mono, …

### Lập trình trên ngôn ngữ C#

Đề lập trình với ngôn ngữ C# có thể dùng Visual Studio, trình soạn thảo mã được phát hàng cùng với C#. Khi tạo một dự án mới trong Visual Studio, bài toán cần được giải quyết hay ứng dụng cần được phát triển được xem là một solution. Trong các solution là các file liên quan đến ứng dụng mà người dùng có thể tuỳ chỉ thông qua các công cụ soạn thảo và phát hiện lỗi mã, các công cụ lớp đồ hoạ và rất nhiều công cụ khác. Các công cụ của Visual Studio được gọi là các VSPackage mà người dùng có thể tải về để tuỳ chỉnh Studio của mình.

*Hình 1. Giao diện làm việc của Visual Studio*

### Windows Forms

Windows Forms là một thư viện lớp đồ hoạ mã nguồn mở được bao gồm trong .NET framework. Các ứng dụng được tạo ra trong Windows Forms là các ứng dụng hướng sự kiện tức là ứng dụng sẽ chỉ hoạt động khi có một sự kiện được gọi, như khi người dùng điền vào một hộp văn bản hay nhấn vào một nút nào đó.

Các nút hay hộp văn bản này được gọi chung là các thành phần của một ứng dụng Windows Forms. Mỗi thành phần như vậy đều có các thuộc tính (property) ảnh hưởng đến hình ảnh, trạng thái của chúng trên ứng dụng.

Các thành phần còn có các sự kiện (event) gắn với từng thành phần đó. Các sự kiện này sẽ kích hoạt tuỳ theo từng loại sự kiện, cho phép người dùng tương tác và sử dụng ứng dụng.

## Trò chơi tic-tac-toe

### Lịch sử và nguồn gốc của trò chơi tic-tac-toe

Các trò chơi theo dạng thắng ba trên một dòng có thể được tìm thấy trên các tấm lợp ngói vào khoảng 1300 trước Công Nguyên ở Ai Cập cổ đại. Một dạng của tic-tac-toe xuât hiện ở khu vực La Mã vào khoảng thế kỉ một trước Công Nguyên. Trong dạng này, hai người chơi sẽ dùng các viên sỏi hoặc đá đặt vào bàn cờ thay vì dùng chữ cái.

Tên gọi tic-tac-toe có thể đã xuất hiện vào khoảng thế kỉ thứ 16, bắt nguồn từ tên “tick-tack”, một trò chơi hai người chơi cổ khác.

Vào năm 1952, OXO (còn gọi là Noughts and Crosses), được phát triển bởi Sandy Douglas thuộc đại học Cambrigde, là một trong những trò chơi điện tử đầu tiên. Người chơi máy có thể chơi một trò chơi hoàn hảo với người chơi thật.

### Luật chơi

Luật chơi tic-tac-toe khá đơn giản. Hai người chơi lần lược điều các chữ cái X hoặc O vào các ô trống trên một bàn cờ, thường có kích thước là 3 hàng, 3 cột. Người thắng cuộc là người điều được ba ô hàng ngang, dọc hoặc đường chéo liên tục cùng một chữ cái X hoặc O.

|  |  |  |
| --- | --- | --- |
| X |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| X |  |  |
|  | O |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| X |  |  |
| X | O |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| X |  |  |
| X | O |  |
| O |  |  |

|  |  |  |
| --- | --- | --- |
| X | X |  |
| X | O |  |
| O |  |  |

|  |  |  |
| --- | --- | --- |
| X | X | O |
| X | O |  |
| O |  |  |

*Hình 2. Mô phỏng một màn chơi, trong đó O thắng*

### Chiến thuật

Nhìn chung, chiến thuật trong trò chơi tic tac toe khác đơn giản. Các chiến thuật trong trò có thể được tổng hợp như sau:

* Thắng: nếu một người chơi có hai chữ trên một hàng, họ nên đặt một chữ nữa để có ba chữ trên một hàng.
* Chặn: nếu đồi thủ có hai chữ trên một hàng, người chơi nên đặt một chữ nữa để chặn nước đi của đối thủ.
* Fork: tạo một tình mà người chơi có hai cách thắng.
* Chặn fork của đối thủ: nếu đối thủ có thể tạo một fork, người chơi nên chặn nó lại. Nếu không thì người chơi nên đi hai chữ cùng một hàng để buộc đối thủ phải phòng thủ.
* Ô giữa: một người chơi đi ô giữa cùa bàng cờ.
* Ô góc đối lập: nếu đối thủ đi một ô góc, người chơi sẽ đi ô đối lập với ô góc đó.
* Ô góc trống: một người chơi đi một ô góc trống.
* Ô cạnh trống: một người chơi đi một ô cạnh trống.

## Thuật toán minimax

### Lý thuyết

Thuật toán minimax là một thuật toán đệ quy được sử dụng trong việc ra quyết định trong lý thuyết trò chơi. Nó cung cấp một nước đi tối ưu cho người chơi giả sử rằng đối thủ cũng đang chơi một cách tối ưu.

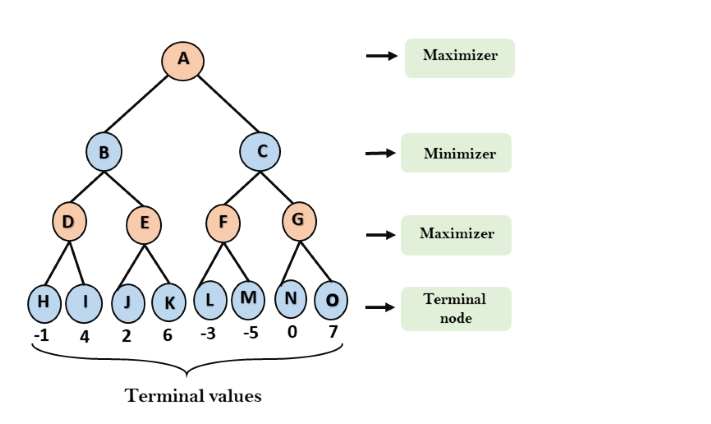
Thuật toán minimax chủ yếu được sử dụng để người chơi máy chơi các trò chơi như cờ vua, cờ caro, tic-tac-toe, cờ vây,… Thuật toán này tính toán quyết định minimax cho trạng thái hiện tại và trả về kết quả đó.

Trong thuật toán này, sẽ có hai người chơi. Một người chơi tối thiểu (minimizing player) và một người chơi tối đa (maximizing player). Hai người chơi sẽ đấu với nhau cho đến khi có một kết quả nhât định (thắng, thua hoặc hoà). Người chơi tối thiểu sẽ tìm nước đi có số điểm thấp nhất. Ngược lại, Người chơi tối đa sẽ tìm nước đi có số điểm cao nhất. Thuật toán sẽ trả về số điểm của người chơi tối thiểu hoặc tối đa tuỳ theo nhu cần của người sử dụng thuật toán.

Có thể xem thuật toán minimax như một thuật toán tìm kiếm trên đồ thị cây với điểm gốc là lúc bắt đầu trò chơi, các nút lá là khi trò chơi kết thúc và có kết quả điểm số cụ thể. Khi di chuyển từ điểm gốc đếm các nút lá sẽ phát sinh ra các trạng thái mới, chính là các nước đi mà thuật toán cần phải xét.

### Ví dụ trên cây trò chơi

Bước 1: Trong bước đầu tiên, thuật toán tạo ra cây trò chơi và áp dụng hàm tiện ích để nhận các giá trị và các trạng thái kết thúc.



Bước 1: khởi tạo cây, gánh giá trị

*Hình 3. Bước 1 thuật toán Minimax*

Trong hình trên, lấy A là điểm xuất phát của cây. Người chơi tối đa (Maximizer) sẽ đi trước và được gánh giá trị là -∞. Người chơi tối thiểu (Minimizer) sẽ đi sau và được gánh giá trị là +∞.

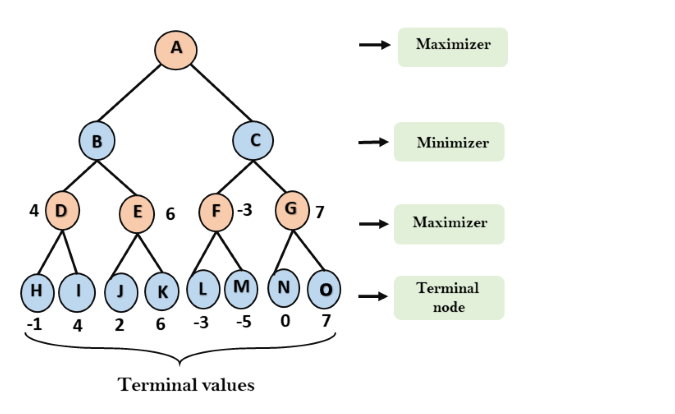
Bước 2: Người chơi tối đa sẽ tìm giá trị lớn nhất ở mỗi nút đầu cuối và truyền về nút kế cận. Trong hình trên sẽ có:

Đối với nút D: max (H, -∞) = max (-1, -∞) = -1 => max (-1,4) = 4

Đối với nút E: max (J, -∞) = max (2, -∞) = 2 => max (2, 6) = 6

Đối với nút F: max (L, -∞) = max (-3, -∞) = -3 => max (-3, -5) = -3

Đối với nút G: max (N, -∞) = max (0, -∞) = 0 => max (0, 7) = 7

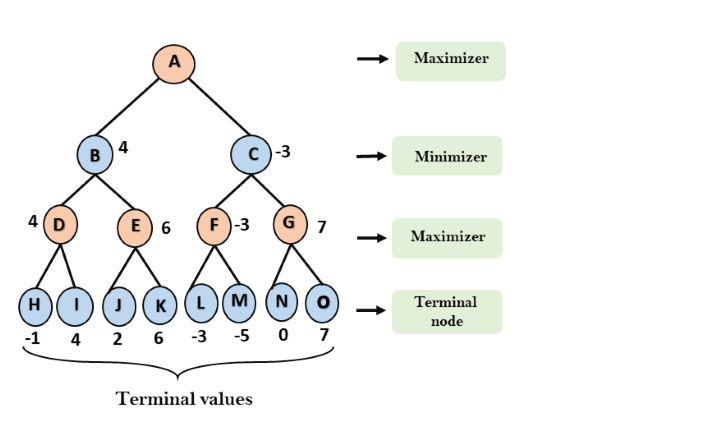


Bước 2: Lược của Maximizer

Bước 1: khởi tạo cây, gánh giá trị

*Hình 4. Bước 2 thuật toán Minimax*

Bước 3: Tương tự như vậy, người chơi tối thiểu sẽ tìm truyền về các nút các nút kế cận một giá trị nhỏ nhất.



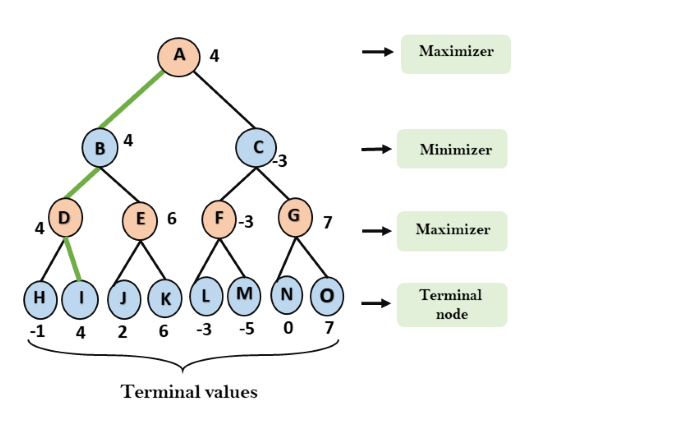
Bước 3: Lược của Minimizer

Bước 2: Lược của Maximizer

Bước 1: khởi tạo cây, gánh giá trị

*Hình 5. Bước 3 thuật toán Minimax*

Bước 4: Lúc này, thuật toán đã đi đến điểm xuất phát của cây. Người chơi tối đa sẽ tiếp tục tìm giá trị lớn nhất và thuật toán sẽ trả về kết quả đã tìm được của người chơi tối đa.



Bước 4: Lược của Maximizer. tre

Bước 3: Lược của Minimizer

Bước 2: Lược của Maximizer

Bước 1: khởi tạo cây, gánh giá trị

*Hình 6. Bước 4 thuật toán Minimax*

Trong thực tế, sẽ có nhiều hơn bước 4 di chuyển và thuật có thể trả về giá trị của người chơi tối thiểu theo yêu cầu.

### Áp dụng vào trò chơi tic-tac-toe

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  |  |  |

Trạng thái ban đầu

+10

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X |  |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  |  | X |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  | X |  |

Lược của X (Maximizer)

# 

-10

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  | X  0 | O |

# 

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | O |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | 0 | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| O | X  +10 |  |

# 

Lược của O (Minimizer)

# 

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | X | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | X | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| O | X | X |

# 

Lược của X (Maximizer)

*Hình 7. Áp dụng thuật toán minimax vào trò chơi tic-tac-toe*

Khi áp dụng thuật toán minimax vào trò chơi tic-tac-toe, có thể cho người chơi tối thiểu là O, người chơi tối đa là X. Khi đó X (người chơi tối đa) sẽ tìm các bước đi có nhiều điểm nhất, hay các bước đi để X thắng hoặc hoà. Ngược lại, O (người chơi tối thiểu) sẽ tìm các bước đi có ít điểm nhất. Trong ví dụ hình 6, kết quả X được gán là +10, O thắng là -10, kết quả hoà là 0. X sẽ đi trước và kết quả trà về sẽ là kết quả X thắng.

Thuật toán minimax sẽ nhận trạng thái của bàn cờ hiện tại và tiến hành xét qua từng bước đi của X hoặc của O tuỳ vào chủ ý của người cài đặt. Khi gặp một trạng thái đặc biệt (X thắng, O thắng, hoà) thuật toán sẽ trả về giá trị đặc biệt đó, nếu chưa tìm thấy thì thuật toán sẽ tiếp tục đến khi tìm thấy thì thôi.

# CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

## 3.1 Yêu cầu của chương trình

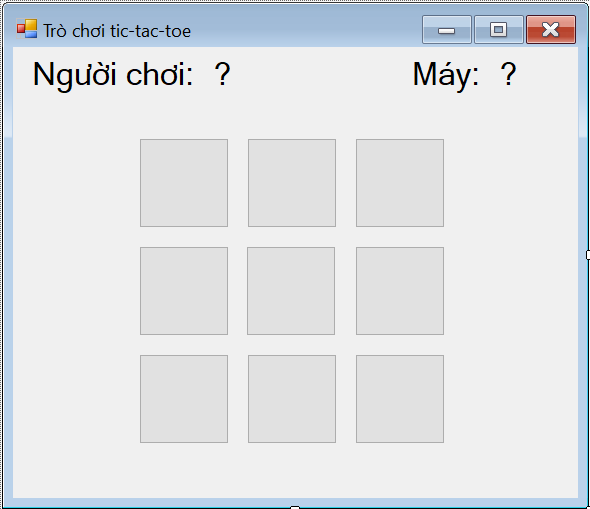
Chương trình sẽ có một giao diện chính và 3 giao diện trò chơi ứng với các kích thước bàn cờ là 3x3, 5x5, 7x7. Bàn cờ sẽ được tạo thành các nút lệnh mà khi người dùng nhấn chuột vào sẽ hiển thị ký tự của mình. Người chơi thật và người chơi máy sẽ lần lược di chuyển trên bàn cờ như vậy cho đến khi có thắng hoặc khi bàn cờ đã đầy, khi đó cả hai sẽ hoà. Trong chương trình này X sẽ được quyền đi trước, O sẽ đi sau.

## 3.2 Thiết kế giao diện

Giao diện chương trình sẽ bao gồm hai thành phần chính: một menu chính cho phép người chơi chọn ký tự là X hay O và chọn chế độ chơi là 3x3, 5x5 hoặc 7x7. Thành phần thứ hai sẽ là màn hình trò chơi cho phép người dùng di chuyển trên bàn cờ là các nút lệnh.

## 

*Hình 8. Giao diện chính của trò chơi.*



## 

*Hình 9. Giao diện trò chơi 3x3, các mục người chơi và máy sẽ thay đổi tuỳ theo lựa chọn của người chơi.*

## 3.3 Cài đặt thuật toán minimax

Trong chương trình này, đặt người chơi tối đa là X, người chơi tối thiểu là O. Kết quả khi X thắng sẽ là 1, khi O thắng sẽ là -1 và kết quả hoà sẽ là 0.

### 3.3.1 Mã giả của thuật toán minimax

**function** minimax(trạng thái bàn cờ, vị trí, maximizingPlayer) **is**

**if** X thắng **then**

**return** 1

**if** O thắng **then**

**return** 1

**if** hoà **then**

**return 0**

**if** maximizingPlayer **then**

value := −∞

**for** chiều cao bàn cờ  **do**

**for** chiều rộng bàn cờ **do**

value := max(value, minimax(trạng thái bàn cờ, vị trí cần xét mới, FALSE))

**return** value

**else**

value := +∞

**for** chiều cao bàn cờ  **do**

**for** chiều rộng bàn cờ **do**

value := min(value, minimax(trạng thái bàn cờ, vị trí càn xét mới, TRUE))

**return** value

### 3.3.2 Lưu đồ thực hiện giải thuật

Gọi đệ quy

Gọi đệ quy

Trạng thái của bàn cờ, vị trí cần xét, lược của người chơi Max

X thắng?

Đ

value = 1

S

Y thắng?

value = -1

Đ

S

value = 0

Hoà?

Đ

S

Lược người chơi Max = true?

S

Đ

i = 0

j = 0

values = -∞

i = 0

j = 0

values = +∞

i = chiều cao của bàn cờ?

i = chiều cao của bàn cờ?

Đ

S

Đ

S

Có thể thực hiện bước đi không?

Trả lại giá trị value

Thực hiện bước đi đó

value = Max(value, minimax(trạng thái bàn cờ mới, vị trí mới, lược của người chơi Min)

S

Đ

Đ

S

j = chiều rộng của bàn cờ?

j = chiều rộng của bàn cờ?

S

Có thể thực hiện bước đi không?

Đ

S

Đ

Thực hiện bước đi đó

value = Max(value, minimax(trạng thái bàn cờ mới, vị trí mới, lược của người chơi Max)

i++

j++

i++

j++

*Hình 10. Lưu dồ thực hiện giải thuật minimax.*

### 3.3.3 Độ phức tạp và số trạng thái tìm kiếm của thuật toán minimax

Thuật toán minimax thực hiện tìm kiếm suốt theo chiều sâu trên một cây trò chơi, trong đó sẽ tìm đếm các nút lá của cây, nên độ phức tạp của thuật toán sẽ là O(bm), trong đó b là số lượng phân nhánh của cây, m là chiều sâu tối đa của cây.

Trong trường hợp bàn cờ 3x3 rỗng, tức là chưa có người chơi nào điều ký tự vào ô trống hết, sẽ có 9 nước đi mở đầu. Ở lược kế tiếp sẽ có 8 nước đi do một ô đã được chọn. Lược tiếp theo nữa sẽ có 7 nước đi,… Vậy có tổng cộng 9! = 362880 trạng thái của bàn cờ mà thuật toán cần phải duyệt qua và chiều sâu của cây trò chơi là 9. Tương tự như vậy, đối với bàn cờ 5x5 sẽ là 25! trạng thái cần duyệt và chiều sâu cây trò chơi là 25, đối với bàn cờ 7x7 sẽ là 49! trạng thái và chiều sâu là 25.

## 3.4 Cài đặt thuật toán kiểm tra thắng, hoà

Để trả về giá trị kết quả cho thuật toán minimax và cũng để xác định người thắng cuộc trong trò chơi, cần cài đặt một thuật thuật toán kiểm tra thắng thua. Ý tưởng chính của thuật toán sẽ là các thuật tìm kiếm tuần tự theo hàng ngang, cột dọc và đường chéo. Nếu tìm thấy 3 chữ cái X hoặc O liên tiếp thì sẽ trả về kết quả tương ứng (1 đối với X, -1 đối với O). Nếu bàn cờ bàn đã đầy, tức là tất cả các ô trên bàn cờ đều đã được đi rồi, thì sẽ trả về kết quả hoà là 0.

### 3.4.1 Tìm kiếm trên hàng ngang

Tìm kiếm theo hàng ngang dơn giản là tìm ba chữ cái liên tiếp giống nhau trên một hàng ngang bất kì và trả lại kết quả nếu tìm thấy. Nếu không có thì tức là chưa có kết quả thắng trên hàng và không trả lại kết quả.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X |  |  |  |  |
|  | X | X | O | X |
|  |  | O | O |  |
|  |  | O |  |  |
|  |  |  |  |  |

Hàng đang xét, không tìm thấy 3 chữ cái liên tiếp nên không trả lại kết quả

*Hình 11. Mô phỏng tìm trên hàng với bàn cờ 5x5*

Mã giả của thuật toán:

**function** timhangngang (trạng thái bàn cờ, hàng cần xét) **is**

**for** chiều rộng bàn cờ  **do**

**if**  3 kí tự liên tiếp trên hàng = kí tự cần xét **then**

**return** kết quả xét;

**return** kết quả xét;

### 3.4.2 Tìm kiếm theo cột dọc

Tìm kiếm trên cột cũng tương tự như tìm kiếm trên hàng.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X |  |  |  |  |
|  | X | X | O |  |
|  |  | O | O |  |
|  | X |  | O |  |
|  |  |  |  |  |

Cột đang xét, tìm thấy 3 chữ cái O liên tiếp nên sẽ trả lại kết quả là -1

*Hình 12. Mô phỏng tìm trên cột*

Mã giả của thuật toán:

**function** timcotdoc(trạng thái bàn cờ, cột cần xét) **is**

**for** chiều cao bàn cờ  **do**

**if**  3 kí tự liên tiếp trên cột = kí tự cần xét **then**

**return** kết quả xét;

**return** kết quả xét;

### 3.4.3 Tìm kiếm theo các đường chéo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X |  |  |  |  |
|  |  |  |  |  |
|  | X | X | O |  |
|  | O | O | X |  |
|  | X | O |  | X |

Đối với tìm kiém trên đường chéo trên bàn cờ 3x3 ta chỉ cần tìm trên đường chéo chính và phụ.Tuy nhiên ở các bàn cờ có kích thước lớn hơn ngoài hai đường chéo chính phụ cần tìm trên các đường chéo lân cận.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | X |  |
|  | X | X | O |  |
|  | X | O |  |  |
|  |  | O |  |  |
|  |  |  |  |  |

Các đường chéo đang xét, tìm thấy 3 chữ cái X liên tiếp nên sẽ trả lại kết quả là 1

*Hình 13. Mô phỏng tìm trên đường chéo*

Mã giả của thuật toán:

**function** timduongcheo(trạng thái bàn cờ, vị trí cần xét) **is**

**for** kích thước bàn cờ  **do**

**if** vị trí cần xét không ở điểm bắt đầu đường chéo **then**

đưa vị trí cần xét về điểm bắt đầu đường chéo;

**return** vị trí mới;

**for** độ dài đường chéo **do**

**if** 3 ký tự liên tiếp trên đường chéo = ký tự cần xét **then**

**return** kết quả**;**

**return** kết quả**;**

### 3.4.3 Xác định bàn cờ đã đầy

Khi bàn cờ đã đầy tức là không thể đi thêm nước đi nào nữa, khi đó cả hai người sẽ hoà và một ván chơi mới sẽ bắt đầu. Để xác định bàn cờ đã đầy đơn giản là duyệt qua tất cả các nút trên cờ, nến các nút đã được đánh chữ thì trả lại kết quả 0.

|  |  |  |
| --- | --- | --- |
| X | X | O |
| O | O | X |
| X | O | X |

Bàn cờ đã đầy, kết quả trả về là 0

### 

*Hình 14. Mô phỏng trạng thái đầy của bàn cờ*

Mã giả của thuật toán:

**function** xacdinhhoa(trạng thái bàn cờ) **is**

**for** chiều cao bàn cờ **do**

**for** chiều rộng bàn cờ **do**

**if** nút đã được đi **then**

biến đếm++;

**if** biến đếm = kích thước bàn cờ **then**

**return** kết quả hoà**;**

### 3.4.4 Phân tích độ phức tạp của các thuật toán

Đối với các thuật toán tìm trên hàng và trên cột độ, vì là thuật toán tìm kiếm tuyết tính đơn thuần, độ phức tạp sẽ là O(m) với là số nút (hay độ dài) trong một hàng hay cột. Tương tự như vậy với thuật toán tìm kiếm trên đường chéo độ phức tạp sẽ là O(n) với n là số ô trên đường chéo cần xét. Với thuật toán xác định bàn cờ đã đầy thì độ phức tạp sẽ là O(b) với b là tổng số các nút trên bàn cờ (9 ô với bàn cờ 3x3, 25 ô trên bàn cờ 5x5, 49 ô trên bàn cờ 7x7) vì thuật toán này cần phải kiểm tra hết cả bàn cờ.

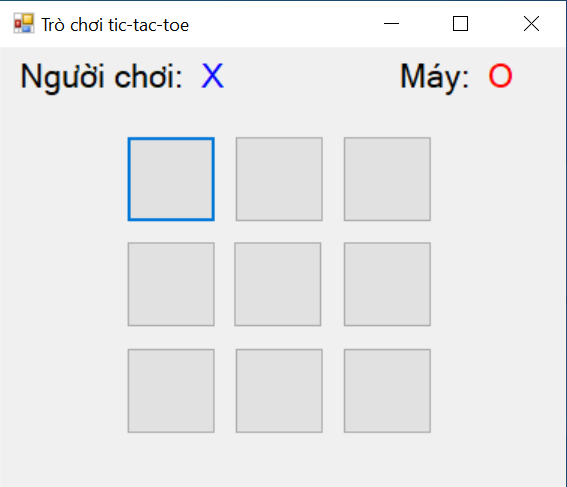
# CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

## 4.1 Kết quả đạt được

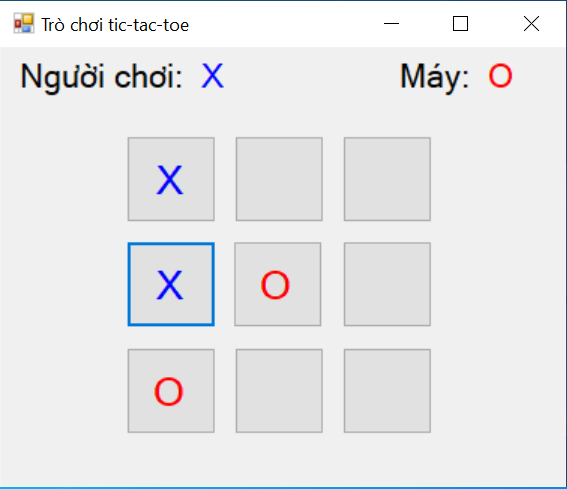
Qua quá trình thực hiện đề án, đã xây dựng được một ứng dụng trò chơi tic-tac-toe với một người chơi máy có thể chơi tìm kiếm và chọn một nước đi phù hợp. Trò chơi cho phép người chơi thật có thể tuỳ ý chọn ký tự X hoặc O cho mình, chọn kích thước bàn cờ là 3x3, 5x5 hoặc 7x7. Khi có kết ai thắng hoặc hoà trò chơi sẽ thông báo kết quả và tự động bắt đầu một ván chơi mới.



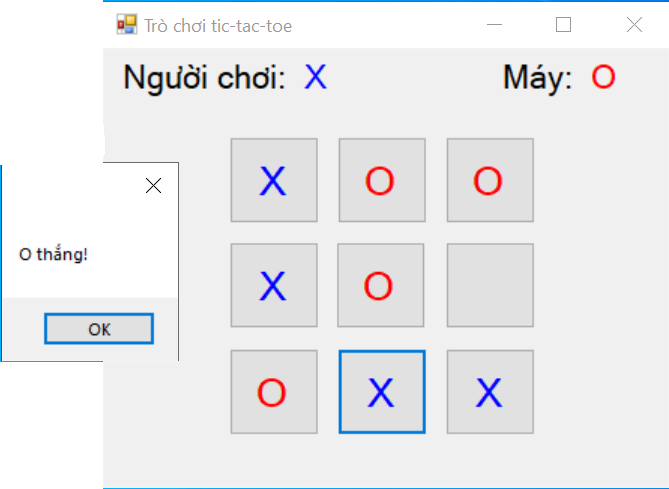
*Hình 11. Giao diện chính.*



*Hình 11. Giao diện trò chơi.*



*Hình 13. Một ván chơi với người chơi thật là X, máy là O.*



*Hình 14. Kết quả một ván chơi.*

## 4.2 Hạn chế

Ngoài những kết quả đã đạt được, vẫn còn một số hạn chế trong việc thực hiện đồ án. Đầu tiên là giao diện nhìn chung còn quá đơn giản, có điểm nổi bật. Một điểm hạn chế nữa là chương trình chưa cho biết tổng kết quả qua mỗi ván chơi ai thắng bao nhiêu lần, hoà bao nhiêu lần.

Một hạn chế quan trọng của chương trình là thời gian thực hiện thuật toán minimax của người chơi máy. Như đã mô tả ở phần hiện thực hoá nghiên cứu, thuật toán có độ phức tạp là O(bm), trong đó b là số lượng phân nhánh của cây, m là chiều sâu tối đa của cây. Với bàn cờ 3x3, chiều sâu của cây trò chơi sẽ là 9. Chưa tính đến độ phân nhánh của cây trò chơi, độ phức tạp của bàn cờ 3x3 sẽ là b9, một số khá lớn. Đối với bàn cờ 5x5 độ phức tạp sẽ là b25, bàn cờ 7x7 sẽ là b49. Số trạng thái của các bàn cờ 5x5, 7x7 cũng là rất lớn, lần lược là 25! và 49!. Như vậy, người chơi máy sẽ mất rất nhiều thời gian duyệt qua từng trạng thái của bàn cờ.

Một hướng khắc phục vấn đề trên là hạn chế chiều sâu của cây được duyệt. Khi đó, thay vì thực hiện tất cả các đỉnh của cây trò cây, người chơi máy sẽ chỉ duyệt đến một chiều sâu nhất định (cụ thể là 8 đối với bàn cờ 3x3, 2 đối với bàn cờ 5x5 và 1 đối với bàn cờ 7x7), nếu không tìm thấy đỉnh của cây thì sẽ trả lại kết quả là 0, tương tự như kết quả hoà. Tuy cách khắc phục này giúp người chơi máy hoạt động nhanh hơn, nó chưa phải là cách tối ưu nhất vì người chơi máy sẽ bỏ qua nhiều trường hợp thắng thua quan trọng.

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  |  |  |

Chiều sâu của cây là 0, xét cho người chơi X

0

0

+10

Chiều sâu của cây là 1, thuật toán ngưng tìm kiếm

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X |  |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  | X |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  |  | X |

-10

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | 0 | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | O |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| O | X  +10 |  |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
|  | X  0 | O |

Thuật toán không tìm đến độ sâu cao hơn, và đã bỏ qua trường hợp khác

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | X | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| X | X | O |

|  |  |  |
| --- | --- | --- |
| X | O | X |
| O | O | X |
| O | X | X |

*Hình 15. Ví dụ về cách khắc phục hạn chế chiều sâu của cây. Lưu ý các trường hợp bị bỏ qua ở độ sâu cao hơn.*

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1 Kết luận

Kết quả thực hiện đồ án đã xây dựng được một chương trình trò chơi tic-tac-toe cơ bản. Qua quá trình thực hiện đồ án, đã tạo ra một giao diện người dùng cơ bản với các nút lệnh cho phép người dùng chọn kích thước bàn cờ, chọn ký tự X hay O. Đã tạo được một giao diện trò chơi với các bàn cờ kích thước 3x3, 5x5, 7x7 và đã cài đặt được một người chơi máy với thuật toán minimax. Trò chơi có thể tự động kiểm kết quả ai thắng hay kết quả hoà và bắt đàu một màn chơi mới.

## 5.2 Hướng phát triển

Ứng dụng trò chơi tic-tac-toe vẫn còn nhiều hạn chế có thể khắc phục được và phát triển thêm nữa. Một trong những cải tiến có thể nói là cải thiện giao diện của ứng dụng thân thiện hơn với người dùng. Ứng dụng có thể được phát triển thêm chế độ hai người chơi thật chơi trực tiếp với nhau. Có thể mở rộng bàn cờ của trò chơi, cho phép người dùng tuỳ ý chọn kích thước chiều cao, chiều rộng. Một vấn đề quan trọng là tốc độ xử lý của thuật toán minimax còn khá chậm, đặt biệt là đối với các bàn cờ có kích thước lớn. Để khắc phục vấn đề này có thể sử dụng giải pháp cắt tỉa alpha-beta để cải tiến thuật toán minimax.

# DANH MỤC TÀI LIỆU THAM KHẢO

(1): Stuart Jonathan Russell, Peter Norvig, Artificial Intelligence: A Modern Approach, tái bản lần thứ 4, Pearson, London, 2021.

(2): Claudi Zaslavsky, Tic tac toe : and other three-in-a row games from ancient Egypt to the modern computer, Crowell, New York, 1982.

(3): Thomas Bolon, How to never lose at Tic-tac-toe, Book Country, New York, 2013

(4): Feng-Hsiung Hsu, IBM's Deep Blue chess grandmaster chips, IEEE Micro, 1999, pp. 70-81.

(5): Kevin Crowley, Robert S. Siegler, Flexible Strategy Use in Young Children's Tic-Tac-Toe, Cognitive Science, pp. 531-561.