

**TRƯỜNG ĐẠI HỌC SÀI GÒN**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN MÔN LẬP TRÌNH PYTHON**

**TÊN ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG TRÒ CHƠI CỜ VUA CLI VỚI THU  
VIỆN PYTHON-CHESS, SIMPLE CHALK VÀ TRÍ TUỆ NHÂN TẠO (AI) TỪ  
GIẢI THUẬT ALPHA-BETA PRUNNING BẰNG NGÔN NGỮ LẬP TRÌNH  
PYTHON**

**Họ và tên thành viên trong nhóm: Lê Nguyễn Minh Huy – 3121560036**

**Nguyễn Thanh Duy– 3121560022**

**Tiền Minh Vy – 3121410023**

**Võ Khương Duy - 3121560023**

**TP. HCM tháng 5/2023**

## NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## **LỜI CẢM ƠN**

Chúng em xin chân thành cảm ơn Khoa Công nghệ thông tin, trường Đại học Sài Gòn đã tạo điều kiện cho chúng em được thực hiện đồ án môn học "Ngôn ngữ lập trình Python". Chúng em cũng xin chân thành cảm ơn thầy Trịnh Tấn Đạt, thầy đã giảng dạy cho chúng em những kiến thức cần thiết cho môn học này. Những kiến thức đó đã giúp cho chúng em rất nhiều trong quá trình làm đồ án báo cáo môn học.. Mặc dù nhóm đã cố gắng hoàn thành đồ án môn học với tất cả nỗ lực của từng thành viên trong nhóm, nhưng đồ án chắc chắn không tránh khỏi những thiếu sót nhất định, rất mong nhận được sự cảm thông, chia sẻ và tận tình đóng góp chỉ bảo của quý Thầy Cô.

Chúng em xin chân thành cảm ơn thầy.

# A. MỞ ĐẦU

## 1. Lý do chọn đề tài.

Hiện nay, ngành Công nghệ thông tin đã và đang có những bước phát triển nhanh chóng ứng dụng của nó trong mọi lĩnh vực trong cuộc sống trên thế giới nói chung và Việt Nam nói riêng. Công nghệ thông tin là một phần không thể thiếu của cuộc sống, góp phần đẩy mạnh công cuộc công nghiệp hóa, hiện đại hóa và quá trình phát triển của đất nước. Việc ứng dụng những thành quả của khoa học công nghệ vào trong đời sống, trong công tác là hết sức thiết yếu. Ứng dụng của công nghệ thông tin kết hợp với truyền thông hóa được xem là một trong những yếu tố mang tính quyết định trong mọi hoạt động của các công ty, tổ chức... nó đóng vai trò quan trọng và không thể thiếu. Công nghệ thông tin và truyền thông hóa góp phần làm thay đổi suy nghĩ, lối mòn tư duy của mỗi con người, nó giúp con người năng động hơn, kết nối nhanh hơn ở mọi lúc mọi nơi làm tăng hiệu quả và năng suất của công việc.

Python là ngôn ngữ lập trình hướng đối tượng bậc cao, dùng để phát triển website và nhiều ứng dụng khác nhau. Python được tạo ra bởi Guido van Rossum và được phát triển trong một dự án mã mở (open source). Với cú pháp cực kì đơn giản và thanh lịch, Python là lựa chọn hoàn hảo cho những ai lần đầu tiên học lập trình. cho nên có thể được dùng để tạo ra các ứng dụng để giải quyết các vấn đề về số, xử lý văn bản, tạo ra trò chơi, và nhiều thứ khác.

Giao diện dòng lệnh (tiếng anh là : command-line interface - CLI) là phương tiện tương tác với chương trình máy tính nơi người dùng (hay máy khách) đưa ra lệnh cho chương trình dưới dạng các dòng văn bản (dòng lệnh) liên tiếp. Đây là loại giao diện mà đa số lập trình viên sẽ tiếp cận khi bắt đầu ngôn ngữ lập trình python bởi sự đơn giản . Và để giúp trang trí cho giao diện dòng lệnh tại em sẽ dùng thêm thư viện Simple Chalk để chỉnh màu nền và màu chữ.

Vì trò chơi cờ vua ta có thể khai triển hết không gian trạng thái nhưng khó khăn chủ yếu là phải tính toán được phản ứng và nước đi của đối thủ mình như thế nào? Cách xử lý đơn giản là ta giả sử đối thủ cũng sử dụng kiến thức về không gian trạng thái giống bản thân. Giải thuật minimax áp dụng giả thuyết này để tìm kiếm không gian trạng thái của trò chơi tối ưu. Từ đó giúp AI quyết định được trạng thái có lợi cho bản thân. Alpha-Beta Pruning là một phiên bản sửa đổi của thuật toán minimax giúp tăng hiệu suất khi thực thi thuật toán bằng cách cắt tỉa cành không cần thiết

Python-chess là một bộ công cụ tiện ích, giúp tạo và xác thực nước đi cũng như hỗ trợ cho việc xuất ra bàn cờ nên chúng em đã quyết định sử dụng thư viện python-chess để xây dựng trò chơi cờ vua(chess) cùng với AI từ giải thuật Alpha-beta Pruning trên CLI.

## 2. Mục đích - mục tiêu của đề tài.

- Mục đích:

- + Nắm chắc được được kỹ năng và kiến thức về lập trình, hướng đối tượng.
- + Tìm hiểu về thư viện python-chess, Simple Chalk bằng ngôn ngữ lập trình Python.

- + Tìm hiểu cách tạo AI bằng giải thuật Alpha-beta Pruning
- + Cũng cố, áp dụng, nâng cao kiến thức đã được học.
- + Nắm bắt được quy trình làm trò chơi cơ bản.

- Mục tiêu:

- + Vận dụng được tính chất của lập trình hướng đối tượng vào trò chơi
- + Sử dụng thư viện python-chess và xây dựng ứng dụng trò chơi cờ vua CLI.
- + Xây dựng được AI thông qua giải thuật Alpha beta Pruning
- + Tạo nên CLI thân thiện bằng việc trang trí màu chữ bằng thư viện Simple Chalk

### **3. Bố cục của đề tài**

-Nội dung gồm 2 phần :

Phần I: Mở đầu

Phần này bao gồm các thông tin về đề tài như:

- 1.Tên đề tài.
- 2.Mục đích.
3. Lý do để xây dựng đề tài.

Phần II: Nội dung

Phần này bao gồm các thông tin về đề tài như:

- 1.Tìm hiểu về quy trình làm trò chơi
- 2.Xây dựng trò chơi

## **B. Xây dựng ứng dụng giải thuật minimax vào trò chơi cờ vua với thư viện chess, simple-chalk bằng ngôn ngữ lập trình python**

### **1. Giới thiệu về trò chơi cờ vua(Chess)**

Cờ Vua xuất hiện ở Ấn Độ vào thế kỷ thứ VI sau Công nguyên. Cho đến ngày nay, người ta vẫn không biết chính xác ngày tháng nào và ai là người khởi xướng ra trò chơi này, chỉ biết rằng đây là một trò chơi phức tạp về đủ mọi phương diện: Bàn cờ, hình thức quân, nhất là luật chơi, phong cách, đường lối, chiến thuật và chiến lược. Do vậy, Cờ Vua không phải là sản phẩm của một người nào mà là một trò chơi trí tuệ của cả một tập thể của các dân tộc Phương Đông. Trải qua nhiều thế hệ, trò chơi này đã phát triển thành môn thể thao cuốn hút hàng triệu triệu người tham gia tập luyện và thi đấu như ngày nay. Có thể nói rằng, Cờ Vua xuất hiện là do nhu cầu của đời sống loài người nhằm phát triển trí tuệ, luyện cách suy nghĩ, cách tính toán và là sự đấu tranh với nhau về mặt lí trí mà bắt đầu cuộc đấu này với nhiều điều kiện như nhau, không có yếu tố ngẫu nhiên, trong đó ai là người thông minh hơn sẽ thắng cuộc.

### **2. Giới thiệu vấn đề**

Yêu cầu chương trình của chúng ta ở đây là làm sao thiết kế được một chương trình trò chơi giống (hoặc gần giống) như cờ vua và trí tuệ nhân tạo dựa vào các phương pháp lập trình đã học. Cụ thể là sử dụng phương pháp lập trình Python kết hợp với các mẫu thiết kế và các giải thuật lập trình

### **3. Tổng quan và phân tích**

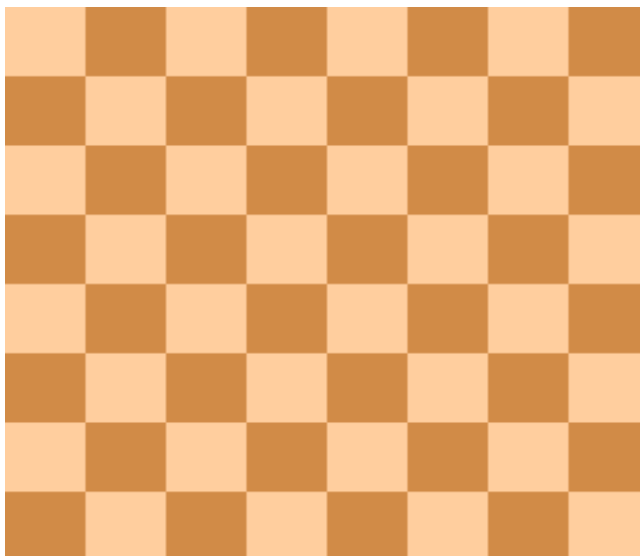
#### **3.1 Khảo sát**

- Bố cục chơi trò chơi gồm 1 panel chính(bàn cờ).

- +Một panel chính:

- Kích thước: có 64 ô cờ, bàn cờ 8x8

- Được bố trí như ma trận gồm 8 cột và 8 dòng ( 1 hình vuông ).



### Minh họa bàn cờ của trò chơi cờ vua (Chess)

Cấu tạo của cờ vua bao gồm: 6 con cờ khác nhau: con xe ( rook ), con mã ( knight ), con tượng ( bishop ), con vua ( king ), con tốt ( pawns ), con hậu ( queen ).



Minh họa 6 con cờ và vị trí của từng con trên bàn cờ

### 3.2 Luật chơi

Các quân cờ có nước đi khác nhau:

- Xe** (ký hiệu quốc tế **R - Rook**) di chuyển theo các đường thẳng dọc theo cột hay hàng tới ô còn trống mà không có quân nào cản trên đường đi hay tới ô bị quân đối phương chiếm giữ (ăn quân) nhưng không thể vượt qua quân đang đứng ở ô đó. Ngoại lệ duy nhất là trường hợp nhập thành. Khi đó nó có thể nhảy qua quân vua của mình để đứng cạnh nó. Chỉ có xe mới có nước đi như thế. Xem thêm nhập thành.
- Tượng** (ký hiệu quốc tế **B - Bishop**) di chuyển theo đường chéo tới ô có cùng màu với nguyên lý tương tự như xe tới ô còn trống hay ô bị quân đối phương chiếm giữ (ăn quân).
- Hậu** (ký hiệu quốc tế **Q - Queen**) có nước đi là tổ hợp đơn giản của chuyển động của xe và tượng. Trong một nước đi nó có thể di chuyển theo đường chéo hoặc đường thẳng dọc theo cột hay hàng, với nguyên lý đi và ăn quân giống như tượng và xe.
- Mã** (ký hiệu quốc tế **N - Knight**) có thể di chuyển tới ô còn trống hay ô bị quân đối phương chiếm giữ (ăn quân) theo dạng hình chữ L (hình chữ nhật 3×2 hay 2×3). Quân mã không bị cản như trong cờ tướng.

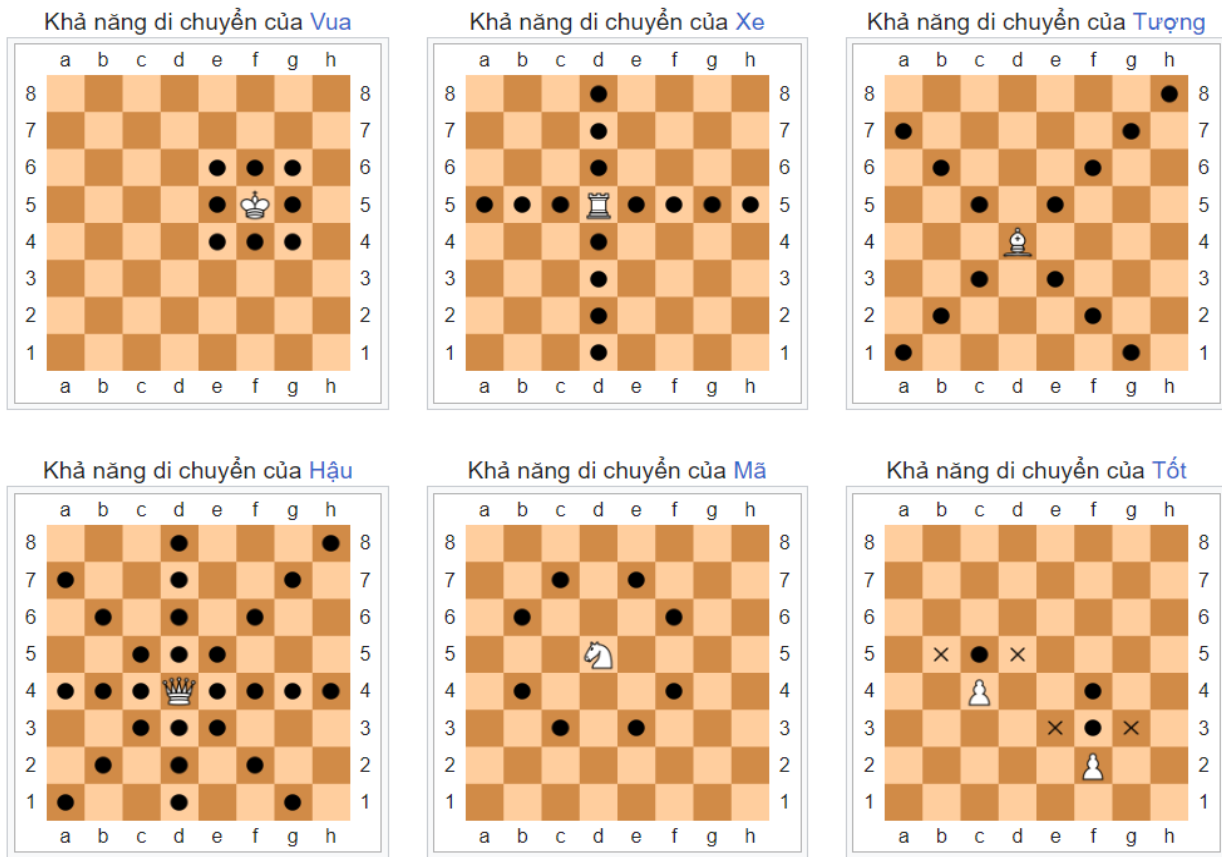
- **Tốt** (không cần ký hiệu) có thể di chuyển thẳng về phía trước chỉ một ô một lần tới ô còn trống (đi mà không ăn quân), nhưng khi di chuyển quân để ăn quân đối phương thì đi chéo. Ví dụ, tốt trắng tại ô c4 có quyền ăn quân đối phương tại b5 hoặc d5 nếu một trong hai ô này có quân đối phương chiếm hoặc di chuyển xuống ô c5 nếu ô này còn trống, trừ hai trường hợp sau:
  1. Nó có thể di chuyển 1 hoặc 2 ô nếu nó đi từ vị trí xuất phát ban đầu tới ô chưa bị chiếm giữ, nhưng không thể nhảy qua một quân khác để tới ô đó. Ví dụ tốt trắng tại g2 có thể đi tới g3 hoặc g4 nếu đây là nước đi đầu tiên của nó và các ô này chưa bị chiếm giữ, nhưng nó không thể đi tới g4 nếu ô g3 đã có một quân nào đó chiếm giữ.
  2. Trong trường hợp khi một quân tốt nào đó của bên trắng đạt tới hàng 5 (ví dụ tới ô e5) và quân tốt thuộc một trong hai cột của bên đen nằm ngay bên cạnh cột mà tốt trắng này đang chiếm giữ (trong trường hợp đã cho là cột d và cột f) đi từ vị trí xuất phát đầu tiên (d7 hay f7) nhảy liên 2 ô tới ô d5/f5 thì tốt trắng tại vị trí e5 ngay tại nước đi sau đó có quyền ăn tốt đen tại ô d5/f5 và di chuyển tiếp tới ô d6/f6. Quyền này sẽ tự động mất, nếu tại nước đi ngay sau đó quân trắng di chuyển quân khác. Tương tự như vậy cho tốt đen khi nó đã chiếm giữ hàng 4. Đây là trường hợp mà trong cờ vua người ta gọi là bắt tốt qua đường (*en passant*). Tốt còn một đặc điểm nữa là khi nó di chuyển đến hàng cuối cùng thì người chơi có quyền phong cấp cho nó thành bất kỳ quân nặng hay nhẹ nào (hậu, xe, tượng, mã).
- **Vua** (ký hiệu quốc tế là **K - King**) là quân quan trọng nhất, nếu mất vua thì người chơi thua cuộc. Mỗi lần đi nó có thể ăn quân hoặc di chuyển sang các ô bao quanh ô mà nó hiện tại đang chiếm giữ, nhưng không thể tới ô mà quân của mình đang chiếm giữ hay các ô bị quân đối phương kiểm soát. Ngoại lệ duy nhất là trường hợp nhập thành. Khi đó nó có thể di chuyển qua hai ô đồng thời với việc di chuyển quân xe của mình để quân xe đó đứng bên cạnh nó về phía cột trung tâm. Ký hiệu của nhập thành là 0-0 (nhập thành gần) và 0-0-0 (nhập thành xa). Xem thêm nhập thành.
  - Lưu ý: Khi thực hiện nhập thành trên thực tế, theo luật của FIDE, bao giờ cũng phải di chuyển vua trước và thực hiện bằng một tay duy nhất.

Khi ăn quân đối phương, quân tấn công sẽ di chuyển tới ô đó và thay thế cho quân đối phương tại vị trí này, bắt tốt qua đường (*en passant*) là ngoại lệ duy nhất. Quân bị ăn được loại ra khỏi bàn cờ. Vua không thể không bảo vệ khỏi nước chiếu, do đó khi bị chiếu thì người chơi phải thực hiện các biện pháp nhằm cứu vua (di chuyển vua khỏi vị trí bị chiếu, ăn quân đang chiếu hay dùng quân khác của mình cản đường chiếu nếu có thể). Nếu không thể có nước đi để cứu vua thì người chơi bị chiếu bí và thua cuộc.

Các ván cờ không phải bao giờ cũng kết thúc bằng chiếu bí. Có thể một bên xin thua, có thể thua do hết giờ hoặc phạm luật chơi. Có thể xảy ra các ván cờ hòa. Một ván cờ vua là hòa khi: do thỏa thuận của hai bên do không bên nào dám mạo hiểm hay khi không đủ lực lượng để chiếu hết, rồi



vào trạng thái hết nước (*stalemate*), cả hai bên lặp lại nước đi ba lần hay luật 50 nước (*perpetual check*).

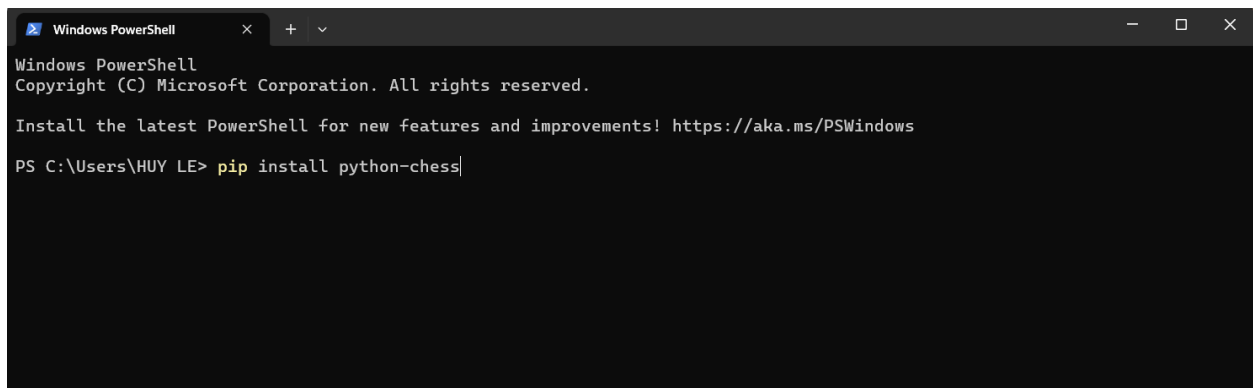


#### 4. Môi trường cài đặt

- Visual Studio Code (64 bit)

- Điều đầu tiên cần làm để lập trình trò chơi trong Python là cài đặt mô-đun chess.py. Để thực hiện cài cho mô-đun này, có thể nhập câu lệnh như sau trong cmd:

**pip install python-chess**

A screenshot of a Windows PowerShell terminal window. The title bar says "Windows PowerShell". The text inside the terminal reads: "Windows PowerShell", "Copyright (C) Microsoft Corporation. All rights reserved.", "Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows", and "PS C:\Users\HUY LE> pip install python-chess".

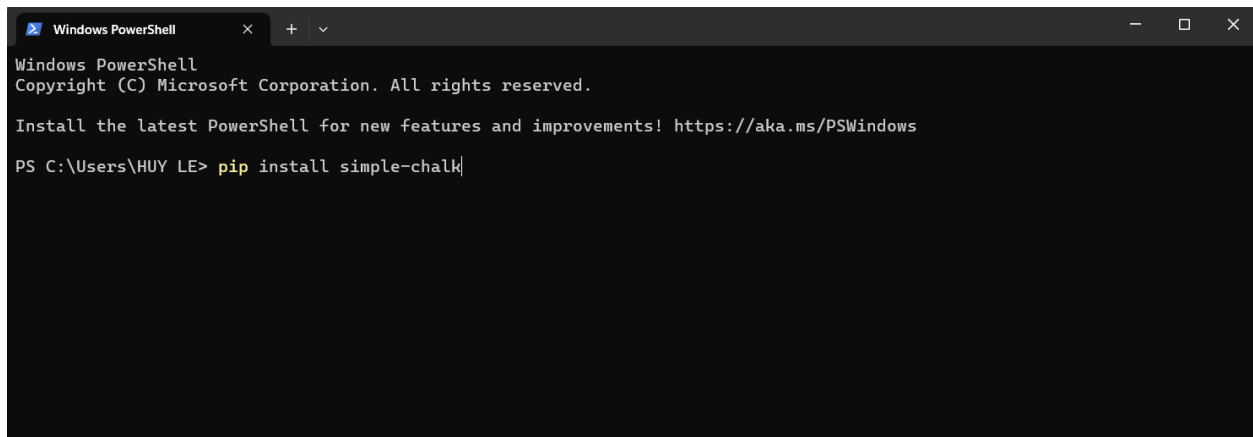
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\HUY LE> pip install python-chess
```

*Cài đặt mô-đun chess.py*

## **pip install simple-chalk**

A screenshot of a Windows PowerShell terminal window. The title bar says "Windows PowerShell". The text inside the terminal reads: "Windows PowerShell", "Copyright (C) Microsoft Corporation. All rights reserved.", "Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows", and "PS C:\Users\HUY LE> pip install simple-chalk".

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows


PS C:\Users\HUY LE> pip install simple-chalk
```

*Cài đặt mô-đun simple-chalk*

## **5. Xây dựng trò chơi**

### **5.1 Khai báo thư viện**

- Sau khi đã cài đặt môi trường xong, ta tiến hành khai báo thư viện những thư viện cần thiết



```
1 import chess
2 from Game import Game
3 game = Game(chess.Board(), chess.WHITE, chess.BLACK)
4 while(True):
5     if(not game.startGame()) : break
```

+ Trong phần Main.py, ta khai báo thư viện python-chess để bắt đầu việc cài đặt trò chơi và khởi tạo đối tượng game từ class Game

+ Trong phần Game.py, ta khai báo thư viện simple\_chalk giúp trang trí chữ khi hiện lên CLI



```
1 import simple_chalk as chalk
```

## 5.2 Tạo cửa sổ bắt đầu trò chơi

- Tiếp theo, ta sẽ tạo màn hình trong cửa sổ CLI để người chơi có thể chọn các chế độ bằng cách tạo trong hàm init(), startGame () và chooseMode().

```

1 def __init__(self, board:ch.Board,WHITE,BLACK):
2     self.board=board
3     self.WHITE=WHITE
4     self.BLACK=BLACK
5     print("")
6     titleColor = chalk.bgRed.white.bold
7     print(titleColor("Chess AI | CLI"))
8     print("")
9 def chooseMode(self):
10    print(chalk.bgBlue.white("Nếu muốn thoát game hãy ấn q "))
11    while(True):
12        choice=input("Hãy chọn chế độ ?\n1.Người và máy \n2.Máy và máy \n3.Người và người \n ")
13        if(choice=='1'):
14            return self.PVAMode()
15        elif(choice=='2'):
16            return self.AVAMode()
17        elif(choice=='3'):
18            return self.PvPmode()
19        elif(choice=='q'):
20            return 0
21        else: print(chalk.bgRed.bold.white("Lựa chọn không hợp lệ "))
22
23 def startGame(self):
24    return self.chooseMode()

```

### *Code tạo cửa sổ CLI*

```

PROBLEMS 98 OUTPUT TERMINAL COMMENTS DEBUG CONSOLE

Microsoft Windows [Version 10.0.19045.2728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Duy-Admin\OneDrive\1.hoc tap\3.tai-lieu-mon-hoc-cac
cáo python\chess_bot\Main.py

Chess AI | CLI

Nếu muốn thoát game hãy ấn q
Hãy chọn chế độ ?
1.Người và máy
2.Máy và máy
3.Người và người

```

### *Cửa sổ CLI sau khi thiết lập theo hàm đã tạo*

- Sau khi tạo ra giao diện trò chơi, sẽ có 3 chế độ chơi để chọn và nếu người chơi muốn thoát trò chơi thì chọn phím “q”

```

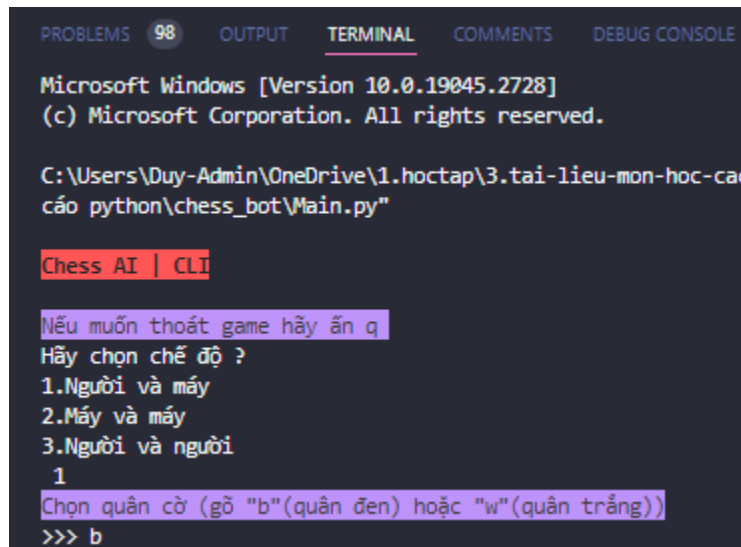
1 def PVAMode(self):
2     color=None
3     while(True):
4         print(chalk.bgBlue.white("""Chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))"""))
5         color = input(">>> ")
6         if(color=="b" or color=="w"): break
7         elif(color=='q'): print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
8         else : print(chalk.bgRed.bold.white("Lựa chọn không hợp lệ "))
9
10    #tìm độ khóa
11    maxDepth=self.getValidDepth()
12
13    if color=="b":
14        while (True):
15            print(chalk.bgBlue.white("AI đang suy nghĩ..."))
16            if(self.board.is_game_over()): break
17            self.playEngineMove(maxDepth, self.WHITE)
18            self.printBoard()
19            if(self.board.is_game_over()): break
20            if(not self.playHumanMove()):print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
21            self.printBoard()
22
23        self.printOutcome()
24    elif color=="w":
25        while (True):
26            if(self.board.is_game_over()): break
27            self.printBoard()
28            if(not self.playHumanMove()):print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
29            if(self.board.is_game_over()): break
30            print(chalk.bgBlue.white("AI đang suy nghĩ..."))
31            self.playEngineMove(maxDepth, self.BLACK)
32            self.printBoard()
33
34        self.printOutcome()
35    #reset the board
36    self.board.reset()
37    #start another game
38    return 1
39 def AVAMode(self):
40
41    maxDepth=self.getValidDepth()
42
43    while (True):
44        #tìm kiếm hàm trong module board
45        if(self.board.is_game_over()): break
46        print(chalk.bgBlue.white("AI 1 đang suy nghĩ..."))
47        self.playEngineMove(maxDepth, self.WHITE)
48        self.printBoard()
49        if(self.board.is_game_over()): break
50        print(chalk.bgBlue.white("AI 2 đang suy nghĩ..."))
51
52        self.playEngineMove(maxDepth, self.BLACK)
53        self.printBoard()
54
55    self.printBoard()
56    self.printOutcome()
57    #reset the board
58    self.board.reset()
59    #start another game
60    return 1
61
62 def PVPMode(self):
63     color=None
64     while(True):
65         print(chalk.bgBlue.white("""người chơi 1 chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))"""))
66         color = input(">>> ")
67         if(color=="b" or color=="w"): break
68         elif(color=='q'): print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
69         else : print(chalk.bgRed.bold.white("Lựa chọn không hợp lệ "))
70
71
72     while(True):
73         if(self.board.is_game_over()): break
74         self.printBoard()
75         print(chalk.bgWhite.white("""Xin mời quân trắng đi"""))
76         if(not self.playHumanMove()):print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
77
78         self.printBoard()
79         if(self.board.is_game_over()): break
80         print(chalk.bgBlue.white("""Xin mời quân đen đi"""))
81         if(not self.playHumanMove()):print('Hi vọng bạn đã có những ván chơi thật tuyệt'); return 0
82
83         self.printBoard()
84
85     self.printOutcome()

```

### Code tạo cửa sổ CLI tương tác với người chơi để bắt đầu ván đấu

- Nếu chúng ta chọn chế độ chơi có người đấu ( 1 và 3), chúng ta sẽ được chọn hai phe: quân đen ( black) hoặc quân trắng (white). Rồi sau đó in ra những dòng khiến người chơi chọn phe để có thể bắt đầu trò chơi và người chơi có thể thoát trò chơi nếu nhấn q :

**Lưu ý:** quân trắng sẽ được đi trước



```
PROBLEMS 98 OUTPUT TERMINAL COMMENTS DEBUG CONSOLE

Microsoft Windows [Version 10.0.19045.2728]
(c) Microsoft Corporation. All rights reserved.

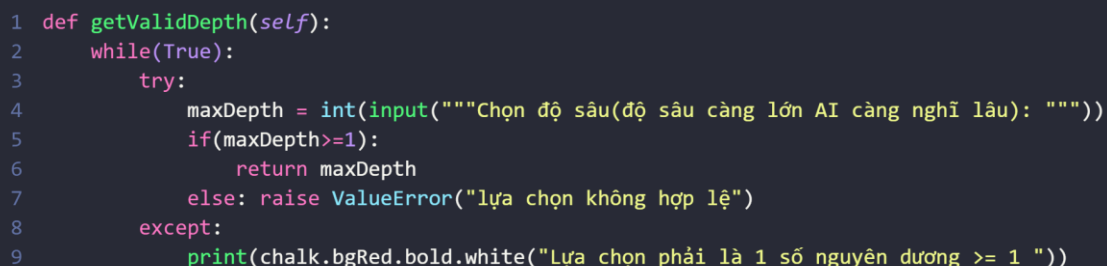
C:\Users\Duy-Admin\OneDrive\1.hoc tap\3.tai-lieu-mon-hoc-cac
cáo python\chess_bot\Main.py"

Chess AI | CLI

Nếu muốn thoát game hãy ấn q
Hãy chọn chế độ ?
1.Người và máy
2.Máy và máy
3.Người và người
1
Chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))
>>> b
```

### Cửa sổ CLI khi chọn quân cờ

-Tiếp đến nếu ta chọn chế độ có đấu với máy( 1 và 2) thì sau khi người chơi chọn quân cờ (chế độ 1) màn hình sẽ hiện ra chọn độ sâu mà máy có thể suy nghĩ hay còn là độ sâu của giải thuật Alpha-beta Pruning



```
1 def getValidDepth(self):
2     while(True):
3         try:
4             maxDepth = int(input("""Chọn độ sâu(độ sâu càng lớn AI càng nghĩ lâu): """))
5             if(maxDepth>=1):
6                 return maxDepth
7             else: raise ValueError("lựa chọn không hợp lệ")
8         except:
9             print(chalk.bgRed.bold.white("Lựa chọn phải là 1 số nguyên dương >= 1 "))
```

### Code chọn độ sâu

```

PROBLEMS 100 OUTPUT TERMINAL COMMENTS DEBUG CONSOLE

C:\Users\Duy-Admin\OneDrive\1.hoctap\3.tai-lieu-mon-hoc-cac

Chess AI | CLI

Nếu muốn thoát game hãy ấn q
Hãy chọn chế độ ?
1.Người và máy
2.Máy và máy
3.Người và người
1
Chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))
>>> b
Chọn độ sâu(độ sâu càng lớn AI càng nghĩ lâu): 4

```

### *Cửa sổ CLI khi chọn độ sâu*

Mặc định bàn cờ của python chess sẽ có quy ước

+ Viết thường là quân đen và viết hoa là quân trắng

+ Ý nghĩa các kí hiệu:

- R: xe
- N: ngựa
- B: tượng
- Q: hậu
- K: vua

```











r n b q k b n r
p p p p p p p p
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . N
P P P P P P P P
R N B Q K B . R

```

### *bàn cờ mặc định của thư viện python chess*

Để in ra bàn cờ thân thiện người dùng ta sẽ dùng kí tự unicode<sup>[4]</sup> như sau :

Tên	Biểu tượng	Ký hiệu Unicode
Vua trắng	♔	U+2654
Hậu trắng	♕	U+2655

Xe trắng		U+2656
Tượng trắng		U+2657
Ngựa trắng		U+2658
Tốt trắng		U+2659
Vua đen		U+265A
Hậu đen		U+265B
Xe đen		U+265C
Tượng đen		U+265D
Ngựa đen		U+265E
Tốt đen		U+265F

Ta thay thế bàn cờ in ra mặc định của thư viện python chess ta sẽ viết code như sau :



```

1  def printBoard(self):
2      string = self.board.__str__()
3      string = string.replace('\n', ' ')
4      #quân đen
5      string = string.replace('p',u'\u265F')
6      string = string.replace('r',u'\u265C')
7      string = string.replace('n',u'\u265E')
8      string = string.replace('b', u'\u265D')
9      string = string.replace('q',u'\u265B')
10     string = string.replace('k',u'\u265A')
11     # quân trắng
12     string = string.replace('P',u'\u2659')
13     string = string.replace('R', u'\u2656')
14     string = string.replace('N', u'\u2658')
15     string = string.replace('B', u'\u2657')
16     string = string.replace('Q',u'\u2655')
17     string = string.replace('K', u'\u2654')
18     string = string.replace('\n', ' ')
19
20     string= string.split(' ')
21
22     #bàn cờ 15x8 do khoảng trắng
23     print(chalk.bgWhite.black.bold("  A  B  C  D  E  F  G  H "))
24     for y in range(8):
25         #vì bàn cờ in ngược
26         print(chalk.bgWhite.black.bold(str(8-y)+'|'),end=' ')
27         for x in range(8):
28             print(string[y * 8 + x],end=' ')
29         print()

```

### Code tạo bàn cờ CLI

Trong hàm printBoard(), vì bàn cờ mặc định gồm 15x8 , 15 cột và 8 dòng (tính cả khoảng trắng) nên khi ta tách bằng split sẽ nhận được mảng 1 chiều, và dựa vào công thức :(dòng\*số dòng + cột hiện tại ) ta có thể in lần lượt được từng con cờ ( dựa vào unicode ) trong màn hình console và cuối cùng là trang trí và in vào bàn cờ:



*bàn cờ sau khi chỉnh sửa lại*

Sau khi chọn độ khó và quân cờ thì chương trình sẽ thực hiện tuần tự nước đi của người và máy thông qua hai hàm `playHumanMove()` và `playEngineMove()`

```

1 def playHumanMove(self):
2     while(True):
3         try:
4             print(chalk.bgGreen.bold.white('Ví dụ nước đi hợp lệ: a2 a3'))
5
6             play = input(chalk.bgBlue.white.underline("Nước đi >>> ")).lower().replace(' ', '')
7             if(play=='q'): return 0
8             self.board.push_san(play)
9             return 1
10        except:
11            print(chalk.bgRed.bold.white(" Nước đi không hợp lệ "))
12
13 def playEngineMove(self, maxDepth, color):
14     engine = Ai(self.board, maxDepth, color)
15
16     if(not self.board.is_game_over()):
17         move=engine.getBestMove()
18         if(move!=None): self.board.push(move)
19

```

*Code di chuyển nước đi trên bàn cờ*

Với hàm `playHumanMove` sau khi người chơi chọn nước đi ta sẽ dùng hàm `push_san` của thư viện `python-chess` để đi nước đó nếu hợp lệ và nếu không hợp lệ thì hàm sẽ truyền ra ngoại lệ và người chơi phải đi lại

Còn với hàm `playEngineMove` thì sẽ khởi tạo đối tượng `Ai` với tham số `board` là bàn hiện tại, độ sâu tối đa (`maxDepth`) và màu `Ai` sẽ cầm (`color`) để lấy về nước đi tốt nhất thông qua hàm `getBestMove()`, hằng số `infinity` sẽ giải thích ở phần xử lý thuật toán. Nếu nước đi đó hợp lệ thì ta sẽ truyền vào hàm `push_san`.

```
1 class Ai:
2     INFINITY=9999
3     # color : black/ white
4     def __init__(self, board: ch.Board, maxDepth, color):
5         self.board=board
6         self.color=color
7         self.maxDepth=maxDepth
8
9     def getBestMove(self):
10        move=self.engine(1,-self.INFINITY,self.INFINITY)
11        #print(move)
12        return move
```

*Code khởi tạo AI trên bàn cờ*

1 nước đi hợp lệ gồm 2 điều kiện:

- + Đúng cú pháp (A2 A8): quân tại A2 đi nước A8
- + Quân đi đúng theo luật

```

1
Chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))
>>> b
Chọn độ sâu(độ sâu càng lớn AI càng nghĩ lâu): 4
AI đang suy nghĩ...
  A B C D E F G H
8| ♜ ♞ ♟ ♚ ♛ ♜ ♞ ♜
7| ♟ ♟ ♟ ♟ ♟ ♟ ♟ ♟
6| . . . . . . . .
5| . . . . . . . .
4| . . . . . . . .
3| . . . . . . ♜
2| ♟ ♟ ♟ ♟ ♟ ♟ ♟ ♟
1| ♜ ♞ ♟ ♚ ♛ ♜ . ♜
Ví dụ nước đi hợp lệ: a2 a3
Nước đi >>> A2 A3
Nước đi không hợp lệ
Ví dụ nước đi hợp lệ: a2 a3
Nước đi >>> A7 A6
  A B C D E F G H
8| ♜ ♞ ♟ ♚ ♛ ♜ ♞ ♜
7| . ♟ ♟ ♟ ♟ ♟ ♟ ♟
6| ♟ . . . . . . .
5| . . . . . ♜ .
4| . . . . . . . .
3| . . . . . . ♜
2| ♟ ♟ ♟ ♟ ♟ ♟ ♟ ♟
1| ♜ ♞ ♟ ♚ ♛ ♜ . ♜

```

Nước đi đúng cú pháp nhưng quân lại không đúng luật

```

  A B C D E F G H
8| ♜ ♞ ♟ ♚ ♛ ♜ ♞ ♜
7| . ♟ ♟ ♟ ♟ ♟ ♟ ♟
6| ♟ . . . . . . .
5| . . . . . ♜ .
4| . . . . . . . .
3| . . . . . . ♜
2| ♟ ♟ ♟ ♟ ♟ ♟ ♟ ♟
1| ♜ ♞ ♟ ♚ ♛ ♜ . ♜
Ví dụ nước đi hợp lệ: a2 a3
Nước đi >>> HA7 CA8
Nước đi không hợp lệ
Ví dụ nước đi hợp lệ: a2 a3
Nước đi >>>

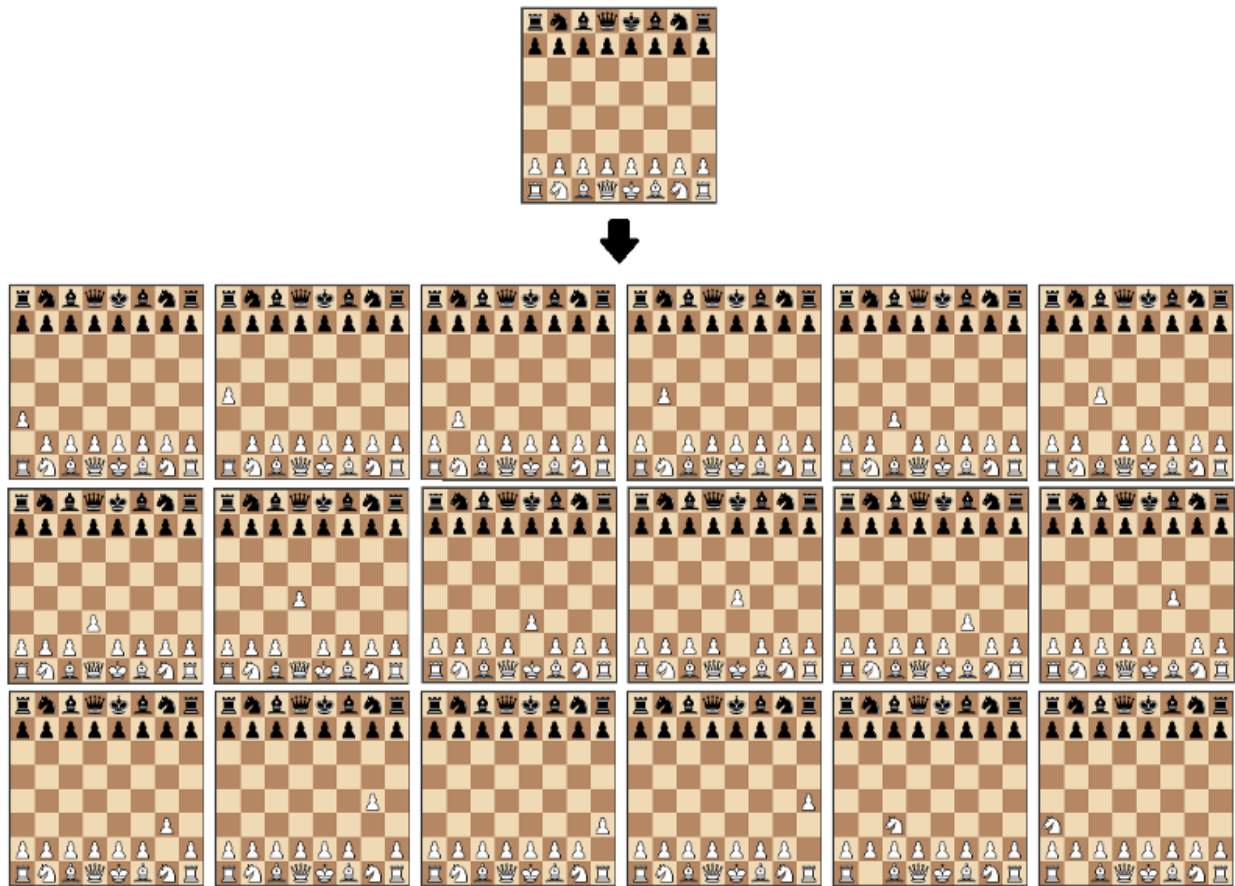
```

Nước đi sai cú pháp

### 5.3 Lý thuyết thuật toán Alpha-beta Prunning của AI :

Trước khi tìm hiểu về Alpha-beta Prunning ta cần tìm hiểu sơ lược về cây trò chơi và minimax

**Cây trò chơi:** một sơ đồ hình cây thể hiện từng trạng thái, từng trường hợp của trò chơi theo từng nước đi.



*Minh họa cây trò chơi*

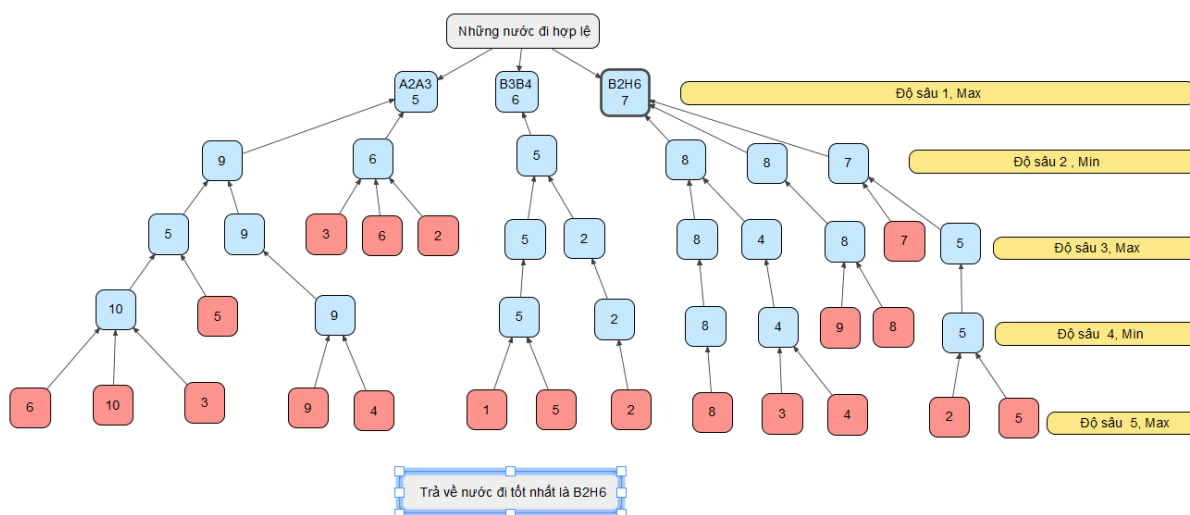
- Mỗi node biểu diễn 1 trạng thái của trò chơi hiện tại trên cây trò chơi.
- Node được gọi nút lá là tại đó trò chơi kết thúc (trạng thái trò chơi lúc đó có thể thắng, thua hoặc hòa).

Mỗi Node biểu diễn cho một trạng thái trên cây trò chơi. Node lá là Node chứa trạng thái kết thúc của trò chơi.

**Giải thuật Minimax** là một thuật toán đệ quy lựa chọn bước đi kế tiếp trong một trò chơi có hai người. Hai đối thủ trong trò chơi được gọi là MIN và MAX luân phiên thay thế nhau đi. MAX đại diện cho người quyết định thắng lợi và cố gắng tối đa hóa ưu thế của mình, ngược lại người chơi đại diện cho MIN lại cố gắng giảm điểm số của MAX và cố gắng làm cho điểm số của mình càng âm càng tốt. Giả thiết đưa ra MIN và MAX có kiến thức như nhau về không gian trạng thái trò chơi và cả hai đối thủ đều cố gắng như nhau. Giải thuật Minimax thể hiện bằng cách định trị các Node trên cây trò chơi:

- Node thuộc lớp MAX thì gán cho nó giá trị lớn nhất của con Node đó.
- Node thuộc lớp MIN thì gán cho nó giá trị nhỏ nhất của con Node đó.

Từ các giá trị này người chơi sẽ lựa chọn cho mình nước đi tiếp theo hợp lý nhất.



*Minh họa sơ đồ cây min max*

Như hình minh họa sơ đồ cây min max, đệ quy kết thúc khi ở độ sâu thứ 5, max là người chơi tìm giá trị lớn nhất nên giá trị trả về nút cha sẽ là giá trị lớn nhất của nút con, với min thì ngược lại là giá trị nhỏ nhất. Cứ như vậy cho tới độ sâu thứ 1 ta sẽ trả về nước đi tốt nhất là B2H6 với giá trị là 7

Như chúng ta đã thấy trong thuật toán tìm kiếm minimax giả sử số nhánh của cây game là  $a$ . Xét độ sâu depth  $b$  thì số nút cần phải tính là  $a^b$ . Đây là con số khá lớn. Vì chúng ta không thể loại bỏ số mũ, nhưng chúng ta có thể cắt giảm nó. Từ đó, có một kỹ thuật mà không cần kiểm tra từng nút của cây trò chơi nhưng vẫn có thể tìm ra quyết định minimax chính xác và kỹ thuật này được gọi là cắt tỉa. Điều này liên quan đến hai tham số ngưỡng alpha và beta để mở rộng trong tương lai, vì vậy nó được gọi là Alpha-Beta Pruning

**Alpha – beta pruning** là một thuật toán tìm kiếm nâng cao của minimax, thuật toán này làm giảm số lượng các node cây được đánh giá bởi thuật toán minimax trong cây tìm kiếm. Thuật toán này dựa theo tìm kiếm đối nghịch trong một số trò chơi với máy (Tic-tac-toe, Cờ vua, v.v.).

Alpha-Beta Pruning có thể được áp dụng ở bất kỳ độ sâu nào của cây, và đôi khi nó không chỉ cắt tỉa lá cây mà còn cắt tỉa toàn bộ cây phụ.

Hai tham số có thể được định nghĩa là:

- Alpha: Sự lựa chọn tốt nhất (giá trị cao nhất) mà chúng tôi đã tìm thấy cho đến nay tại bất kỳ điểm nào trên con đường của Maximizer. Giá trị ban đầu của alpha là  $-\infty$ .
- Beta: Lựa chọn tốt nhất (giá trị thấp nhất) mà chúng tôi đã tìm thấy cho đến nay tại bất kỳ điểm nào dọc theo đường dẫn của Minimizer. Giá trị ban đầu của beta là  $+\infty$ .

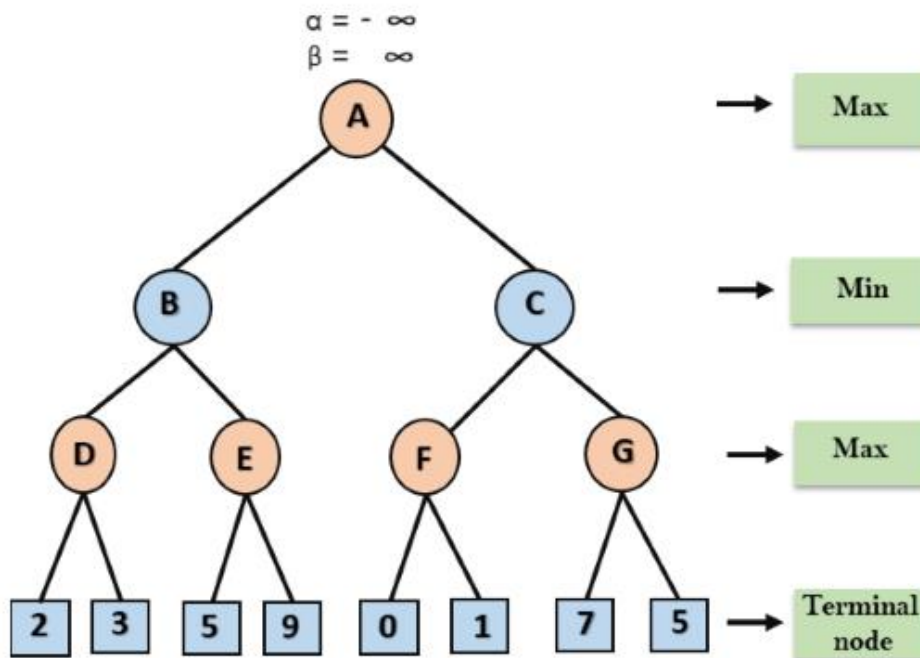
Điều kiện chính cần thiết để Alpha-Beta Pruning là:  $\alpha \geq \beta$

### Những điểm chính về việc Alpha-Beta Pruning:

- Max player sẽ chỉ cập nhật giá trị của alpha.
- Min player sẽ chỉ cập nhật giá trị của phiên bản beta.
- Trong khi đệ quy (backtracking) lại cây, các giá trị của nút sẽ được chuyển đến các nút phía trên thay vì các giá trị của alpha và beta.
- Ta sẽ chỉ chuyển các giá trị alpha, beta cho các nút con.

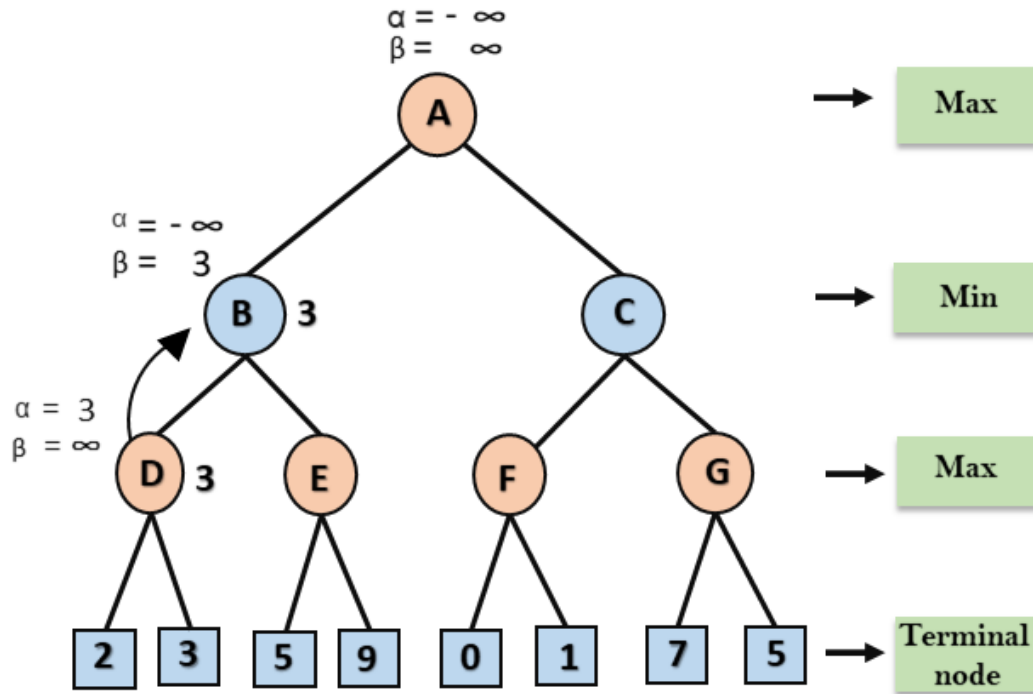
### Giải thích cách hoạt động của Alpha-beta Pruning:

**Bước 1:** Ở bước đầu tiên, người chơi Max sẽ bắt đầu di chuyển đầu tiên từ nút A nơi  $\alpha = -\infty$  và  $\beta = +\infty$ , những giá trị alpha và beta này được truyền lại cho nút B nơi lại  $\alpha = -\infty$  và  $\beta = +\infty$  và Nút B chuyển cùng một giá trị cho nút con D của nó.



**Bước 2:** Tại nút D, giá trị của  $\alpha$  sẽ được tính theo lượt của nó cho Max. Giá trị của  $\alpha$  được so sánh với đầu tiên là 2 và sau đó là 3, và  $\max(2, 3) = 3$  sẽ là giá trị của  $\alpha$  tại nút D và giá trị của nút cũng sẽ là 3.

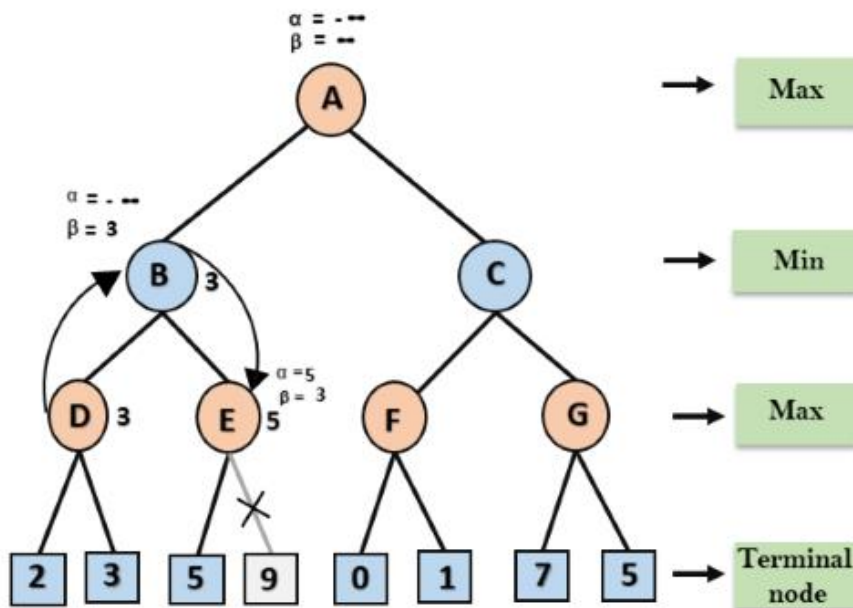
**Bước 3:** Bây giờ thuật toán quay ngược lại nút B, trong đó giá trị của  $\beta$  sẽ thay đổi vì đây là lượt của Min, Bây giờ  $\beta = +\infty$ , sẽ so sánh với giá trị của các nút tiếp theo có sẵn, tức là  $\min(\infty, 3) = 3$ , do đó tại nút B bây giờ  $\alpha = -\infty$  và  $\beta = 3$ .



Trong bước tiếp theo, thuật toán duyệt qua nút kế tiếp tiếp theo của nút B là nút E, và các giá trị của  $\alpha = -\infty$  và  $\beta = 3$  cũng sẽ được chuyển.

**Bước 4:** Tại nút E, Max sẽ đến lượt, và giá trị của alpha sẽ thay đổi. Giá trị hiện tại của alpha sẽ được so sánh với 5, vì vậy  $\max(-\infty, 5) = 5$ , do đó tại nút E  $\alpha = 5$  và  $\beta = 3$ , trong đó  $\alpha \geq \beta$ , vì vậy kế thừa bên phải của E sẽ bị lược bỏ, và thuật toán sẽ không đi qua nó, và giá trị tại nút E sẽ là 5.

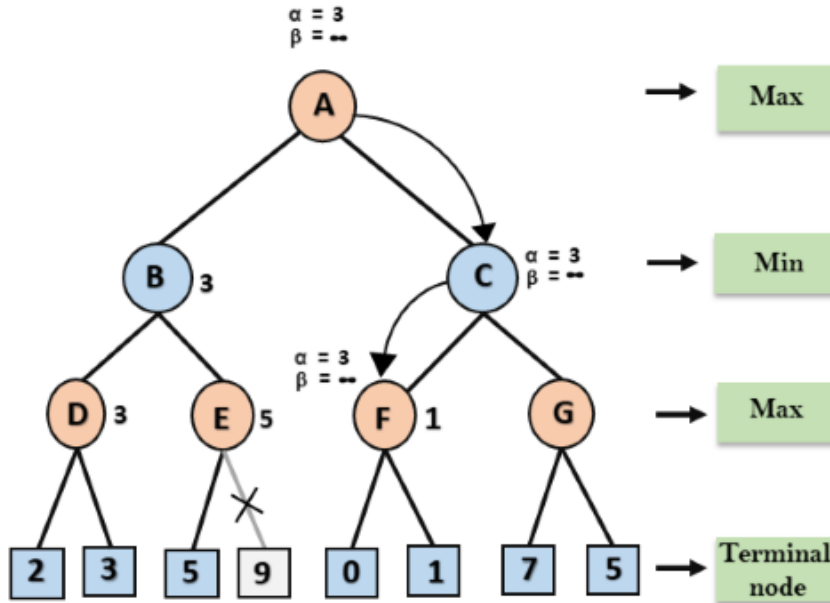




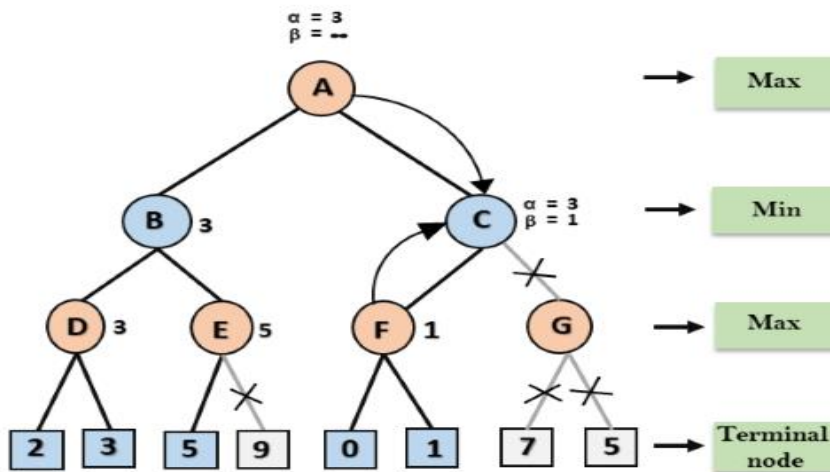
**Bước 5:** Ở bước tiếp theo, thuật toán lại chiếu ngược cây, từ nút B đến nút A. Tại nút A, giá trị của alpha sẽ được thay đổi giá trị lớn nhất có sẵn là 3 như  $\max(-\infty, 3) = 3$  và  $\beta = +\infty$ , hai giá trị này bây giờ được chuyển đến người kế nhiệm bên phải của A là Nút C.

Tại nút C,  $\alpha = 3$  và  $\beta = +\infty$ , và các giá trị tương tự sẽ được chuyển cho nút F.

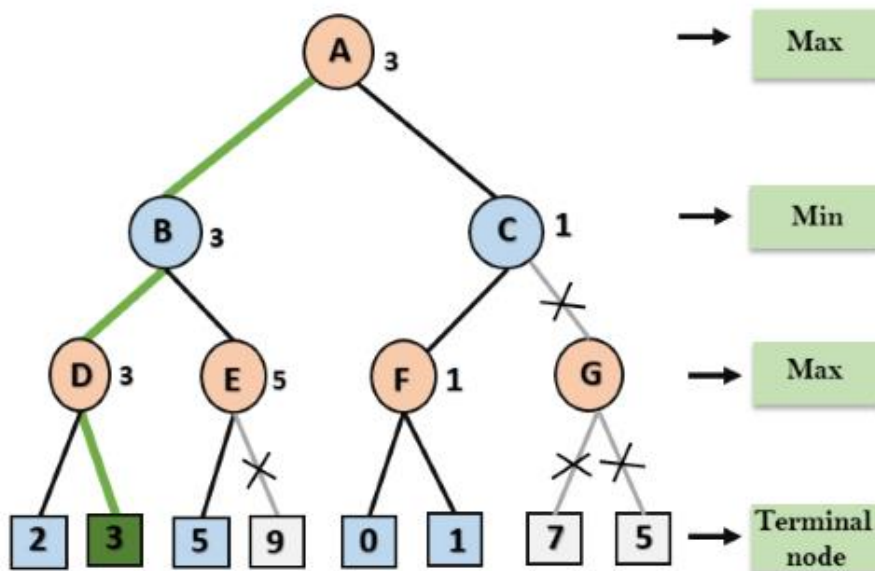
**Bước 6:** Tại nút F, một lần nữa giá trị của  $\alpha$  sẽ được so sánh với nút con bên trái là 0 và  $\max(3, 0) = 3$ , sau đó so sánh với nút con bên phải là 1 và  $\max(3, 1) = 3$  vẫn còn  $\alpha$  vẫn là 3, nhưng giá trị nút của F sẽ trở thành 1.



**Bước 7:** Nút F trả về giá trị nút 1 cho nút C, tại C  $\alpha = 3$  và  $\beta = +\infty$ , ở đây giá trị beta sẽ được thay đổi, nó sẽ so sánh với 1 nên  $\min(\infty, 1) = 1$ . Bây giờ tại C,  $\alpha = 3$  và  $\beta = 1$ , và một lần nữa nó thỏa mãn điều kiện  $\alpha > \beta$ , vì vậy con tiếp theo của C là G sẽ bị lược bớt, và thuật toán sẽ không tính toàn bộ cây con G.









**Bước 8:** C bây giờ trả về giá trị từ 1 đến A ở đây giá trị tốt nhất cho A là  $\max(3, 1) = 3$ . Sau đây là cây trò chơi cuối cùng là hiển thị các nút được tính toán và các nút chưa bao giờ tính toán. Do đó, giá trị tối ưu cho bộ cực đại là 3 cho ví dụ này.



#### 5.4 Áp dụng giải thuật Alpha-beta pruning vào trí tuệ nhân tạo trò chơi cờ vua

Đầu tiên ta sẽ xây dựng hệ thống tính điểm dựa trên quân cờ. Thực tế có rất nhiều hệ thống về tiêu chuẩn giá trị quân cờ trong cờ vua <sup>[8]</sup> dưới đây là 1 trong số đó dựa theo đánh giá của hệ thống theo kiện tướng Hans Berliner

Ký hiệu					
Quân cờ	Tốt	Ngựa	Tượng	Xe	Hậu
Giá trị	1	3.2	3.33	5.1	8.8



```

1  def squareResPoints(self, square):
2      pieceValue = 0
3      typePiece=self.board.piece_type_at(square)
4      if(typePiece==None): return pieceValue
5
6      if(typePiece == ch.PAWN):
7          pieceValue = 1
8      elif (typePiece == ch.ROOK):
9          pieceValue = 5.1
10     elif (typePiece == ch.BISHOP):
11         pieceValue = 3.33
12     elif (typePiece == ch.KNIGHT):
13         pieceValue = 3.2
14     elif (typePiece == ch.QUEEN):
15         pieceValue = 8.8
16
17     if (self.board.color_at(square)!=self.color):
18         return -pieceValue
19     else:
20         return pieceValue

```

Và để tính giá trị của 1 trạng thái trong cây sơ đồ ta sẽ lặp 64 ô trên bàn cờ vua và tính tổng. Nếu là lượt của đối phương(người chơi min) ta sẽ trả về giá trị âm, ngược lại ta sẽ trả về giá trị dương(người chơi max)

```
1 def evalFunct(self)->float:
2     compt = 0
3     for i in range(64):
4         compt+=self.squareResPoints(ch.SQUARES[i])
5
6     compt += self.endgameOpportunity()
7     # print(compt)
8     return compt
9
10 def endgameOpportunity(self):
11     if (self.board.is_checkmate()):
12         #minimize player
13         if (self.board.turn == self.color):
14             return -self.INFINITY
15         #maximizeplayer
16     else:
17         return self.INFINITY
18
19     return 0
```

Nếu như ở lượt người chơi max bị chiếu ta sẽ cho trừ vô cực để tránh đi nước đó, ngược lại nếu là người chơi min bị chiếu, ta sẽ cộng vô cực để AI đi nước đó

Tiếp theo là giải thuật Alpha-beta pruning:

```

1  def engine(self,depth,alpha,beta):
2
3      if ( depth == self.maxDepth or self.board.is_game_over()):
4          return self.evalFunc()
5
6      else:
7          moveList = list(self.board.legal_moves)
8          firstMove=None
9          newValueComputation = None
10
11         if(depth % 2 != 0):
12             newValueComputation = -self.INFINITY
13             for move in moveList:
14
15                 #Play move i
16                 self.board.push(move)
17
18                 value = self.engine(depth + 1,alpha,beta)
19
20                 if(value > newValueComputation ):
21                     if (depth == 1):
22                         firstMove=move
23                         newValueComputation = value
24                         alpha= max(newValueComputation,alpha)
25                 self.board.pop()
26                 if(alpha>=beta):
27                     break
28         else:
29             newValueComputation = self.INFINITY
30             for move in moveList:
31
32                 self.board.push(move)
33
34                 value = self.engine(depth + 1,alpha,beta)
35                 if(value < newValueComputation):
36                     newValueComputation = value
37                     beta= min(newValueComputation,beta)
38
39                 self.board.pop()
40                 if(alpha>=beta):
41                     break
42         if (depth>1):
43             return newValueComputation
44         else:
45             return firstMove

```

Ta xác định điểm kết thúc của đệ quy khi đạt tới độ sâu tối đa hoặc là trò chơi kết thúc thì ta sẽ tính toán. Nếu độ sâu chia hết cho 2 ta nói đó là lượt của người chơi min, ngược lại là lượt của người chơi max. Với mỗi nước đi sau khi tính toán nếu như độ sâu  $n > 1$  thì ta sẽ trả giá trị đó cho  $n - 1$  để tính tiếp nước khả thi tiếp theo đồng thời lưu lại giá trị alpha, beta. Đặc biệt nếu ở nước đi tiếp theo ta có giá trị  $\alpha > \beta$  ta sẽ ngắt vòng lặp và trả về giá trị nước đi đó, lập tức đi đến độ sâu  $n - 1$ . Nếu độ sâu = 1 ta sẽ trả về nước đi chuyển

## 5.5 Tạo cửa sổ kết thúc

```
1 def is_game_over(self):
2     if(self.board.is_checkmate()):
3         return True
4     elif(self.board.is_stalemate()):
5         return True
6     elif(self.board.is_repetition()):
7         return True
8     elif(self.board.is_fifty_moves()):
9         return True
10    elif(self.board.is_insufficient_material()):
11        return True
12    return False
13 def printOutcome(self):
14    if(self.board.is_checkmate()):
15        print('Trận đấu kết thúc do chiếu hết')
16    elif(self.board.is_stalemate()):
17        print('Trận đấu hòa do hết nước đi')
18    elif(self.board.is_repetition()):
19        print('Trận đấu hòa do lặp lại nước đi 3 lần')
20    elif(self.board.is_fifty_moves()):
21        print('Trận đấu hòa do 50 nước đi không có quân bị bắt')
22    elif(self.board.is_insufficient_material()):
23        print("Trận đấu hòa do cả hai bên không có đủ lực lượng để chiến thắng")
24
```

Để xác định 1 ván chơi kết thúc ta sẽ dựa vào

- Hòa về lực lượng chiến đấu

Một ván cờ được trọng tài xử hòa khi không bên nào có đủ lực lượng ở trên bàn cờ có cơ hội để chiếu hết cờ đối phương. Và ván cờ dạng hòa thường xuất hiện ở những trường hợp dưới đây:

- Vua đối mặt Vua
- Vua + Mã chống Vua
- Vua + Tượng chống Vua
- Hòa do hết nước đi (PAT)

Khi đến lượt một bên đi, vua không bị chiếu hết nhưng hết nước đi hợp lệ thì ván cờ đó được xử hòa theo Luật “hòa PAT”.

- Hòa do bắt biến 3 lần

Trong một ván cờ, nếu một thế cờ được di chuyển lặp lại 3 lần liên tiếp thì ván cờ đó coi như được xử hòa. Lúc này người chơi ngay lập tức báo trọng tài để xác nhận kết quả cờ hòa này.

- Hòa cờ sau 50 nước

Khi chơi cờ vua, nếu trong 50 nước cờ không có quân cờ nào bị bắt và không có nước Tốt nào được thực hiện thì ván cờ đó được xử hòa.

- Khi chiếu hết đối phương

Nếu vua bị chiếu và không có nước đi hợp lệ sẽ được xử thắng nếu là bên chiếu và thua nếu bị chiếu



```
2| ♀ ♀ ♀ ♀ ♀ ♀ ♀ ♀
1| ♀ . ♀ ♀ ♀ ♀ . ♀
AI 2 đang suy nghĩ...
```

	A	B	C	D	E	F	G	H
8	♂ ♀ ♀ ♀ ♀ ♀ . ♀							
7	♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀							
6	. . . . . . ♀							
5	. . . . . ♀ .							
4	. . . . . . .							
3	. . ♀ . . . . .							
2	♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀							
1	♂ . ♀ ♀ ♀ ♀ ♀ . ♀							

```
AI 1 đang suy nghĩ...
A B C D E F G H
8| ♂ ♀ ♀ ♀ ♀ ♀ . ♀
7| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
6| . . . . . . ♀
5| . . . . . . .
4| . . . . ♀ . .
3| . . ♀ . . . . .
2| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
1| ♂ . ♀ ♀ ♀ ♀ ♀ . ♀
```

```
AI 2 đang suy nghĩ...
A B C D E F G H
8| ♂ ♀ ♀ ♀ ♀ ♀ . ♀
7| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
6| . . . . . . ♀
5| . . . . . . .
4| . . . . ♀ . .
3| . . ♀ . . . . .
2| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
1| ♂ . ♀ ♀ ♀ ♀ ♀ . ♀
```

```
AI 1 đang suy nghĩ...
A B C D E F G H
8| ♂ ♀ ♀ ♀ ♀ ♀ . ♀
7| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
6| . . . . . . ♀
5| . . . . . ♀ .
4| . . . . . . .
3| . . ♀ . . . . .
2| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
1| ♂ . ♀ ♀ ♀ ♀ ♀ . ♀
```

```
A B C D E F G H
8| ♂ ♀ ♀ ♀ ♀ ♀ . ♀
7| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
6| . . . . . . ♀
5| . . . . . ♀ .
4| . . . . . . .
3| . . ♀ . . . . .
2| ♂ ♀ ♀ ♀ ♀ ♀ ♀ ♀
1| ♂ . ♀ ♀ ♀ ♀ ♀ . ♀
```

Trận đấu hòa do lặp lại nước đi 3 lần

Nếu muốn thoát game hãy ấn q

Hãy chọn chế độ ?

1. Người và máy
2. Máy và máy
3. Người và người

|

## 6. Demo trò chơi

1 vài hình ảnh về trò chơi

```
Chess AI | CLI

Nếu muốn thoát game hãy ấn q
Hãy chọn chế độ ?
1.Người và máy
2.Máy và máy
3.Người và người
2
Chọn độ sâu(độ sâu càng lớn AI càng nghĩ lâu): 4
AI 1 đang suy nghĩ...
  A B C D E F G H
8| ♔ ♕ ♖ ♗ ♘ ♙ ♚ ♛
7| ♜ ♞ ♟ ♡ ♢ ♣ ♤ ♥
6| . . . . . . . .
5| . . . . . . . .
4| . . . . . . . .
3| . . . . . . ♜
2| ♜ ♞ ♟ ♡ ♢ ♣ ♤ ♥
1| ♔ ♕ ♖ ♗ ♘ ♙ ♚ ♛
AI 2 đang suy nghĩ...
  A B C D E F G H
8| ♔ ♕ ♖ ♗ ♘ ♙ ♚ ♛
7| ♜ ♞ ♟ ♡ ♢ ♣ ♤ ♥
6| . . . ♜ . . . .
5| . . . . . . . .
4| . . . . . . . .
3| . . . . . . ♜
2| ♜ ♞ ♟ ♡ ♢ ♣ ♤ ♥
1| ♔ ♕ ♖ ♗ ♘ ♙ ♚ ♛
```

Nước đi >>>		A7 A6						
	A	B	C	D	E	F	G	H
8	♖	♗	♘	♙	♚	♛	♜	♞
7	.	♙	♙	♙	♙	♙	♙	♙
6	♙	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	♜
2	♙	♙	♙	♙	♙	♙	♙	♙
1	♞	♜	♘	♙	♚	♛	.	♞
AI đang suy nghĩ...								
	A	B	C	D	E	F	G	H
8	♖	♗	♘	♙	♚	♛	♜	♞
7	.	♙	♙	♙	♙	♙	♙	♙
6	♙	.	.	.	.	.	.	.
5	.	.	.	.	.	.	♜	.
4	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	.
2	♙	♙	♙	♙	♙	♙	♙	♙
1	♞	♜	♘	♙	♚	♛	.	♞

C:\Users\Duy-Admin\OneDrive\1.hoc tap\3.tai-lieu-mon-hoc-cac-ho

### Chess AI | CLI

Nếu muốn thoát game hãy ấn q

Hãy chọn chế độ ?

- 1.Người và máy
  - 2.Máy và máy
  - 3.Người và người
- 3

người chơi 1 chọn quân cờ (gõ "b"(quân đen) hoặc "w"(quân trắng))  
>>> b

	A	B	C	D	E	F	G	H
8	♜	♞	♝	♚	♛	♞	♟	♠
7	♙	♙	♙	♙	♙	♙	♙	♙
6	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	.
2	♟	♟	♟	♟	♟	♟	♟	♟
1	♠	♞	♝	♚	♛	♞	♟	♜

Xin mời quân trắng đi

Ví dụ nước đi hợp lệ: a2 a3

Nước đi >>> A7 A6

Nước đi không hợp lệ

Ví dụ nước đi hợp lệ: a2 a3

Nước đi >>> A2 A3

	A	B	C	D	E	F	G	H
8	♜	♞	♝	♚	♛	♞	♟	♠
7	♙	♙	♙	♙	♙	♙	♙	♙
6	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
3	♟	.	.	.	.	.	.	.
2	.	♟	♟	♟	♟	♟	♟	♟
1	♠	♞	♝	♚	♛	♞	♟	♜

Xin mời quân đen đi

Ví dụ nước đi hợp lệ: a2 a3

Nước đi >>> A7 A6

	A	B	C	D	E	F	G	H
8	♜	♞	♝	♚	♛	♞	♟	♠
7	.	♟	♟	♟	♟	♟	♟	♟
6	♙	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
3	♟	.	.	.	.	.	.	.
2	.	♟	♟	♟	♟	♟	♟	♟
1	♠	♞	♝	♚	♛	♞	♟	♜

Xin mời quân trắng đi

Ví dụ nước đi hợp lệ: a2 a3

## Tài liệu tham khảo

### Trang điện tử

- [1] <https://websitehcm.com/thuat-toan-alpha-beta-pruning/> truy cập lần cuối ngày 3/5/2022.
- [2] <https://python-chess.readthedocs.io/en/latest/> truy cập lần cuối ngày 3/5/2022.
- [3] <https://pypi.org/project/simple-chalk/> truy cập lần cuối ngày 3/5/2022.
- [4] [https://en.wikipedia.org/wiki/Chess\\_symbols\\_in\\_Unicode](https://en.wikipedia.org/wiki/Chess_symbols_in_Unicode) truy cập lần cuối ngày 3/5/2022
- [5] <https://www.youtube.com/> truy cập lần cuối ngày 3/5/2022
- [6] <https://viblo.asia/p/thuat-toan-minimax-ai-trong-game-APqzeaVVzVe> truy cập lần cuối ngày 3/5/2022
- [7] [https://viblo.asia/p/code-game-chess-ai-bang-c-3P0IPD9olox#\\_iv-phan-ai-cay-tro-choi-chien-luoc-minimax-cat-tia-alpha-beta-14](https://viblo.asia/p/code-game-chess-ai-bang-c-3P0IPD9olox#_iv-phan-ai-cay-tro-choi-chien-luoc-minimax-cat-tia-alpha-beta-14) truy cập lần cuối ngày 3/5/2022
- [8] [https://en.wikipedia.org/wiki/Chess\\_piece\\_relative\\_value](https://en.wikipedia.org/wiki/Chess_piece_relative_value) truy cập lần cuối ngày 3/5/2022
- [9] [https://vi.wikipedia.org/wiki/Cờ\\_vua](https://vi.wikipedia.org/wiki/Cờ_vua) truy cập lần cuối ngày 3/5/2022.
- [10] <https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/> truy cập lần cuối ngày 3/5/2022.

### Tài liệu khác

- [1] Slide bài giảng Ngôn ngữ lập trình Python của thầy Trịnh Tấn Đạt.

## Mục lục

A. MỞ ĐẦU.....	4
1. Lý do chọn đề tài.....	4
2. Mục đích - mục tiêu của đề tài.....	4
3. Bố cục của đề tài .....	5
B. Xây dựng ứng dụng giải thuật minimax vào trò chơi cờ vua với thư viện chess, simple-chalk bằng ngôn ngữ lập trình python.....	6
1. Giới thiệu về trò chơi cờ vua(Chess) .....	6
2. Giới thiệu vấn đề.....	6
3. Tổng quan và phân tích.....	6
3.1 Khảo sát .....	6
3.2 Luật chơi .....	7
4. Môi trường cài đặt.....	9
5. Xây dựng trò chơi .....	10
5.1 Khai báo thư viện.....	10
5.2 Tạo cửa sổ bắt đầu trò chơi.....	11
5.3 Lý thuyết thuật toán Alpha-beta Prunning của AI : .....	20
5.4 Áp dụng giải thuật Alpha-beta prunning vào trí tuệ nhân tạo trò chơi cờ vua .....	27
5.5 Tạo cửa sổ kết thúc .....	31
6. Demo trò chơi .....	34
Tài liệu tham khảo .....	37