**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

🙡★🙣



**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**MÔN LẬP TRÌNH JAVA – IS216**

**Họ và tên sinh viên : Đỗ Thanh Tùng**

**Mã số sinh viên: 19522491**

**Lớp : IS216.L22**

**GIẢNG VIÊN HƯỚNG DẪN**

**Mai Xuân Hùng**

***TP. HỒ CHÍ MINH, 6/2021***

Mục lục

[1. Mở đầu 1](#_Toc75587720)

[**1.1.** **Mục tiêu đề tài** 1](#_Toc75587721)

[**1.2.** **Nội dung báo cáo** 1](#_Toc75587722)

[2. Tổng quan về game cờ vua 2](#_Toc75587723)

[**2.1.** **Luật chơi căn bản** 2](#_Toc75587724)

[**2.2.** **Nhập thành** 4](#_Toc75587725)

[**2.3.** **En passant** 6](#_Toc75587726)

[**2.4.** **Kết thúc trò chơi** 7](#_Toc75587727)

[3. Mô hình cơ bản game cờ vua 8](#_Toc75587728)

[**3.1.** **Mô hình phân cấp class** 8](#_Toc75587729)

[**3.2.** **Tile class** 10](#_Toc75587731)

[**3.3.** **Piece class** 11](#_Toc75587732)

[**3.4.** **Class con của Piece Class** 12](#_Toc75587733)

[**3.4.1.** **Bishop** 12](#_Toc75587734)

[**3.4.2.** **Rook** 13](#_Toc75587735)

[**3.4.3.** **Queen** 14](#_Toc75587736)

[**3.4.4.** **Knight** 15](#_Toc75587737)

[**3.4.5.** **Pawn** 16](#_Toc75587738)

[**3.4.6.** **King** 17](#_Toc75587739)

[**3.5.** **Board class** 18](#_Toc75587740)

[**3.6.** **Player class** 21](#_Toc75587741)

[**3.7.** **Alliance Enum** 23](#_Toc75587742)

[**3.8.** **BoardUtils class** 23](#_Toc75587743)

[**3.9.** **Move class** 25](#_Toc75587744)

[**3.10.** **GUI** 26](#_Toc75587745)

[**3.10.1.** **Table class** 26](#_Toc75587746)

[**3.10.2.** **TilePanel class** 28](#_Toc75587747)

[**3.10.3.** **MoveLog class** 29](#_Toc75587748)

[**3.10.4.** **GameHistoryPanel class** 30](#_Toc75587749)

[**3.10.5.** **TakenPiecesClass** 30](#_Toc75587750)

[4. Một số vấn đề đặc biệt trong cờ vua 31](#_Toc75587751)

[**4.1.** **Promotion** 31](#_Toc75587752)

[**4.2.** **Castling** 33](#_Toc75587753)

[**4.3.** **En Passant** 36](#_Toc75587754)

[5. Thuật toán Minimax 40](#_Toc75587755)

[**5.1.** **Sơ lược về thuật toán** 40](#_Toc75587756)

[**5.2.** **Cài đặt** 41](#_Toc75587757)

[**5.2.1.** **MiniMax class** 41](#_Toc75587758)

[**5.2.2.** **StandardBoardEvaluator class** 44](#_Toc75587759)

[**5.3.** **Chạy thử** 46](#_Toc75587760)

[6. Kiểm thử 47](#_Toc75587761)

[7. Tổng kết 49](#_Toc75587762)

[**7.1.** **Kết quả đạt được** 49](#_Toc75587763)

[**7.2.** **Ưu điểm** 49](#_Toc75587764)

[**7.3.** **Hạn chế** 49](#_Toc75587765)

[**7.4.** **Hướng phát triển** 49](#_Toc75587766)

[8. Tài liệu tham khảo 50](#_Toc75587767)

DANH MỤC HÌNH

[**Hình 1: Bàn cờ vua lúc mới bắt đầu trận đấu [1] 2**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588560)

[**Hình 2: Nhập thành gần: 0-0 4**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588561)

[**Hình 3: Nhập thành xa 0-0-0 5**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588562)

[**Hình 4: Minh hoạ một nước bắt tốt qua đường 7**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588563)

[**Hình 5: Vua đen đang bị chiếu bởi xe trắng 8**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588564)

[**Hình 6: Sơ đồ phân cấp class cơ bản của game cờ vua 10**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588565)

[**Hình 7: Hierachy của Tile class 11**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588566)

[**Hình 8: Hierachy của EmptyTile class 11**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588567)

[**Hình 9: Hierachy của OccupiedTile class 11**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588568)

[**Hình 10: Hierachy class của Piece class 12**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588569)

[**Hình 11: Hierachy của Piece class 12**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588570)

[**Hình 12: Cách di chuyển của quân tướng 13**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588571)

[**Hình 13: Thuật toán tìm các nước đi của quân tướng 14**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588572)

[**Hình 14: Cách di chuyển của quân xe 15**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588573)

[**Hình 15: Cách di chuyển của quân hậu 15**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588574)

[**Hình 16: Cách di chuyển của quân mã 16**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588575)

[**Hình 17: Thuật toán tính toán cách di chuyển của quân mã 17**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588576)

[**Hình 18: Cách di chuyển của quân tốt 18**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588577)

[**Hình 19: Cách di chuyển của quân vua 19**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588578)

[**Hình 20: Hierachy của board class 20**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588579)

[**Hình 21: Khởi tạo bàn cờ 21**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588580)

[**Hình 22: Hierachy của builder class 21**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588581)

[**Hình 23: Hierachy của player class 22**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588582)

[**Hình 24: Hierachy của BlackPlayer class 23**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588583)

[**Hình 25: Hierachy của WhitePlayer class 23**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588584)

[**Hình 26: Hierachy của Alliance enum 24**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588585)

[**Hình 27: Hierachy của BoardUtils class 25**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588586)

[**Hình 28: Hierachy của Move class 26**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588587)

[**Hình 29: Hierachy class của Move class 26**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588588)

[**Hình 30: Giao diện của trò chơi 27**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588589)

[**Hình 31: Observer interface 28**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588590)

[**Hình 32: Thông báo trò chơi kết thúc 29**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588591)

[**Hình 33: 1 instance của TilePanel class 29**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588592)

[**Hình 34: Quân cờ được chọn và nước đi có thể đi của nó 30**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588593)

[**Hình 35: Hierachy của MoveLog class 30**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588594)

[**Hình 36: Bảng lịch sử các nước đi 31**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588595)

[**Hình 37: Bảng các quân cờ đã bị ăn mất 31**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588596)

[**Hình 38: Trước khi phong cấp 32**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588597)

[**Hình 39: Sau khi phong cấp 32**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588598)

[**Hình 40: Hàm isPromotionSquare của quân trắng 33**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588599)

[**Hình 41: Xử lý promtion 33**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588600)

[**Hình 42: Trước khi phong hậu 33**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588601)

[**Hình 43: Sau khi phong hậu 34**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588602)

[**Hình 44: Gom tất cả các move thành legalMove trong lớp PLayer 34**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588603)

[**Hình 45: Ví dụ trước khi nhập thành 35**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588604)

[**Hình 46: Xử lý castleMove của BlackPlayer class 35**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588605)

[**Hình 47: Sau khi nhập thành bên vua quân trắng 36**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588606)

[**Hình 48: Trước khi nhập thành bên quân hậu cờ đen 36**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588607)

[**Hình 49: Sau khi nhập thành bên quân hậu cờ đen 36**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588608)

[**Hình 50: Set enpassant ở class PawnJump 37**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588609)

[**Hình 51: Xử lý enpassant move tại Pawn class 38**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588610)

[**Hình 52: Trước khi enpassant move 39**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588611)

[**Hình 53: Thực thi enpassant move 40**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588612)

[**Hình 54: Mô hình cây của thuật toán minimax lớp depth bằng 4 41**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588613)

[**Hình 55: Visualize thuật toán minimax trong cờ vua 42**](#_Toc75588614)

[**Hình 56: Hierachy của MiniMax class 43**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588615)

[**Hình 57: Flowchart của hàm execute để tạo ra move nhờ minimax 45**](#_Toc75588616)

[**Hình 58: Hàm tính giá trị bàn cờ của mỗi người chơi 45**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588617)

[**Hình 59: Hàm tính đánh giá giá trị của bàn cờ 45**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588618)

[**Hình 60: Hierachy của StandardBoardEvaluator class 46**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588619)

[**Hình 61: Lựa chọn chế độ và depth 47**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588620)

[**Hình 62: Nước đi của máy sẽ được highligh màu đỏ 47**](#_Toc75588621)

[**Hình 63: Thời gian chạy của mỗi nước đi 47**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588622)

[**Hình 64: Các test class 48**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588623)

[**Hình 65: Kết quả chạy thử tất cả test class 48**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588624)

[**Hình 66: Một test case trong các tập test 49**](file:///C:\Users\Asus\Desktop\JAVA\ChessGame\BaoCao.docx#_Toc75588625)

1. **Mở đầu**
   1. **Mục tiêu đề tài**

Đề tài này sẽ hướng đến việc tìm hiểu cách lập trình java căn bản và nâng cao từ việc lập trình OOP và GUI căn bản. Mục tiêu của đề tài là tìm hiểu về cách lập trình game cờ vua căn bản.

Để thực hiện được điều này, nội dung của báo cáo bao gồm:

* Tìm hiểu tổng quan về cách lập trình JAVA.
* Tìm hiểu luật chơi cơ bản của game.
* Tìm hiểu về thuật toán Minimax.
  1. **Nội dung báo cáo**

Nội dung báo cáo gồm x phần:

Phần 1: Giới thiệu tổng quan về đồ án.

Phần 2: Giới thiệu tổng quan về trò chơi cờ vua.

Phần 3: Xây dựng mô hình game cờ vua cơ bản.

Phần 4: Trình bày một số vấn đề đặc biệt của game cờ vua.

Phần 5: Trình bày về thuật toán Minimax để áp dụng cho AI của game cờ vua.

Phần 6: Kiểm thử

Phần 7: Hạn chế và kết luận.

1. **Tổng quan về game cờ vua**
   1. **Luật chơi căn bản**

Trò chơi này diễn ra trên một [bàn cờ](https://vi.wikipedia.org/w/index.php?title=B%C3%A0n_c%E1%BB%9D&action=edit&redlink=1) [hình vuông](https://vi.wikipedia.org/wiki/H%C3%ACnh_vu%C3%B4ng) gồm 8 hàng (đánh số từ 1 đến 8) và 8 cột (đánh các chữ cái từ a đến h), tạo ra 64 ô hình vuông với các màu đậm và nhạt xen kẽ nhau, với mỗi người chơi sẽ có ô màu nhạt ở hàng cuối cùng bên tay phải của mình khi ngồi vào bàn chơi cờ. Mỗi người sẽ bắt đầu ván cờ với 16 [quân cờ](https://vi.wikipedia.org/w/index.php?title=Qu%C3%A2n_c%E1%BB%9D&action=edit&redlink=1) và sẽ lần lượt đi các quân của mình sau khi đối phương đã đi xong một nước. Các quân cờ của mỗi bên bao gồm 8 [tốt](https://vi.wikipedia.org/wiki/T%E1%BB%91t_(c%E1%BB%9D_vua)), 2 [mã](https://vi.wikipedia.org/wiki/M%C3%A3_(c%E1%BB%9D_vua)), 2 [tượng](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua#Nguy%C3%AAn_l%C3%BD_ch%C6%A1i), 2 [xe](https://vi.wikipedia.org/wiki/Xe_(c%E1%BB%9D_vua)), 1 [hậu](https://vi.wikipedia.org/wiki/H%E1%BA%ADu_(c%E1%BB%9D_vua)) và 1 [vua](https://vi.wikipedia.org/wiki/Vua_(c%E1%BB%9D_vua)). Người cầm quân [trắng](https://vi.wikipedia.org/wiki/Tr%E1%BA%AFng) đi đầu tiên; người cầm quân [đen](https://vi.wikipedia.org/wiki/%C4%90en) đi sau [1].

**A picture containing object, chessman, indoor, several

Description automatically generated**

Hình 1: Bàn cờ vua lúc mới bắt đầu trận đấu [1]

Các quân cờ có nước đi khác nhau:

* **Xe** (ký hiệu quốc tế **R - Rook**) di chuyển theo các [đường thẳng](https://vi.wikipedia.org/wiki/%C4%90%C6%B0%E1%BB%9Dng_th%E1%BA%B3ng) 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](https://vi.wikipedia.org/wiki/Nh%E1%BA%ADp_th%C3%A0nh). Khi đó nó có thể nhảy qua quân vua của mình để đứng cạnh nó. Chỉ có xe mới có nước đi như thế.
* **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](https://vi.wikipedia.org/wiki/L) (hình chữ nhật 3×2 hay 2×3). Quân mã không bị cản như trong [cờ tướng](https://vi.wikipedia.org/wiki/C%E1%BB%9D_t%C6%B0%E1%BB%9Bng).
* **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 liền 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](https://vi.wikipedia.org/wiki/B%E1%BA%AFt_T%E1%BB%91t_qua_%C4%91%C6%B0%E1%BB%9Dng) (en passant). Tốt còn một đặc điểm nữa là khi nó di chuyển đến hàng cuối cùng thì người chơi có quyền [phong cấp](https://vi.wikipedia.org/wiki/Phong_c%E1%BA%A5p_(c%E1%BB%9D_vua)) 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](https://vi.wikipedia.org/wiki/Nh%E1%BA%ADp_th%C3%A0nh). 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). [1]
  1. **Nhập thành**

Chart

Description automatically generated**Nhập thành**là một nước đi đặc biệt trong [trò chơi](https://vi.wikipedia.org/wiki/Tr%C3%B2_ch%C6%A1i) [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua), trong đó [vua](https://vi.wikipedia.org/wiki/Vua_(c%E1%BB%9D_vua)) và một trong hai quân [xe](https://vi.wikipedia.org/wiki/Xe_(c%E1%BB%9D_vua)) tham gia vào việc di chuyển. Khi nhập thành vua di chuyển 2 qua 2 ô về phía quân xe tham gia nhập thành, và sau đó di chuyển quân xe tới ô mà quân vua vừa di chuyển qua sao cho nó nằm ngay bên cạnh quân vua.

Hình 2: Nhập thành gần: 0-0

Chart

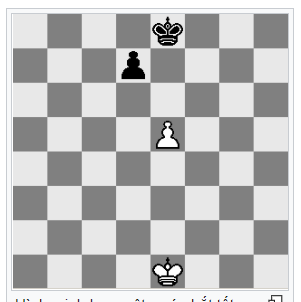
Description automatically generatedViệc nhập thành chỉ có thể được phép nếu tất cả các điều kiện sau được đảm bảo vào thời điểm thực hiện việc nhập thành:

Hình 3: Nhập thành xa 0-0-0

1. Quân vua chưa bao giờ bị di chuyển;
2. Quân xe tham gia vào nhập thành cũng chưa bao giờ bị di chuyển;
3. Không có quân nào nằm giữa vua và xe đó;
4. Vua không bị [chiếu](https://vi.wikipedia.org/wiki/Chi%E1%BA%BFu_(c%E1%BB%9D)), các ô mà vua sẽ di chuyển qua không nằm dưới sự kiểm soát (ô hay đường nằm trong tầm chiếu) của quân đối phương, cũng như việc nhập thành không làm cho vua đứng vào ô bị chiếu. (Phần cuối cùng của quy tắc này là một phần của quy tắc chung là không bao giờ đặt vua vào ô bị chiếu).

Một số người còn cho là các yêu cầu cho việc nhập thành còn phức tạp hơn những quy tắc trên đây, nhưng điều đó là sai lầm. Để làm rõ vấn đề này, cần khẳng định là nhập thành vẫn có thể thực hiện khi:

1. Vua có thể đã bị chiếu trước đây nhưng miễn là nó có quân hộ giá để cản nước chiếu và không bị chiếu vào thời điểm nhập thành.
2. Xe tham gia nhập thành có thể đang bị tấn công.
3. Xe tham gia nhập thành có thể di chuyển qua hoặc đứng vào các ô bị kiểm soát bởi quân đối phương.
   1. **En passant**

**Bắt Tốt qua đường** (hay thường gặp trong một số tài liệu, [tiếng Pháp](https://vi.wikipedia.org/wiki/Ti%E1%BA%BFng_Ph%C3%A1p): **en passant**) là một nước đi trong [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua). Đây là nước bắt quân đặc biệt mà người chơi thực hiện sau khi [Tốt](https://vi.wikipedia.org/wiki/T%E1%BB%91t_(qu%C3%A2n_c%E1%BB%9D)) của đối phương di chuyển qua ô kiểm soát của Tốt bên phía mình.



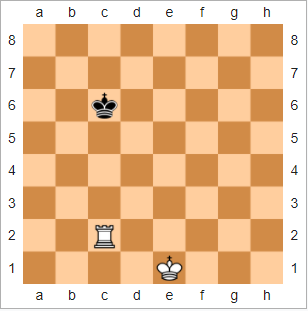




Hình 4: Minh hoạ một nước bắt tốt qua đường

* 1. **Kết thúc trò chơi**

Trong [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua), **chiếu** là một tình huống mà [Vua](https://vi.wikipedia.org/wiki/Vua_(c%E1%BB%9D_vua)) của một bên đang không bị đe dọa bắt mất trong lượt tiếp theo của đối thủ. Vua ở trong tình trạng như vậy gọi là bị chiếu. Người chơi cần phải tìm cách khong hóa giải nước chiếu nếu có thể, bằng cách đặt một quân khác vào giữa Vua của mình và quân đang chiếu của đối phương trên đường chiếu, ăn quân đang chiếu của địch, hoặc di chuyển Vua sang ô an toàn khác. Nếu người chơi không thể hóa giải được nước chiếu, ván cờ sẽ kết thúc trong thế [chiếu hết](https://vi.wikipedia.org/w/index.php?title=Chi%E1%BA%BFu_h%E1%BA%BFt&action=edit&redlink=1) (checkmate) và người chơi sẽ thua.



Hình 5: Vua đen đang bị chiếu bởi xe trắng

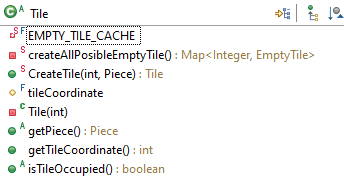
1. **Mô hình cơ bản game cờ vua**
   1. **Mô hình phân cấp class**

**Diagram

Description automatically generated**

Hình 6: Sơ đồ phân cấp class cơ bản của game cờ vua

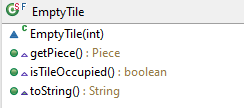
* 1. **Tile class**

Tile class dùng để biểu thị ô vuông trên bàn cờ.

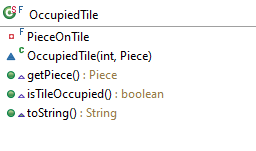
Hình 7: Hierachy của Tile class

Trong Tile class sẽ gồm 2 inner class con là EmptyTile và OccupiedTile biểu thị lần lượt cho 2 loại ô là ô trống và ô đã bị chiếm.

* **EmptyTile**

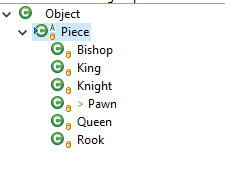


Hình 8: Hierachy của EmptyTile class

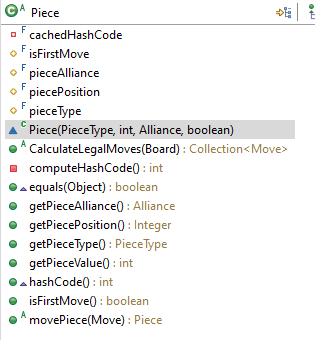
* **OccupiedTile**

Hình 9: Hierachy của OccupiedTile class

* 1. **Piece class**

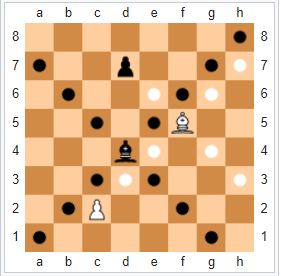
Piece class dùng để biểu thị cho mỗi quân cờ, là class cha cho mỗi loại quân cờ.

Hình 10: Hierachy class của Piece class



Hình 11: Hierachy của Piece class

* 1. **Class con của Piece Class**
     1. **Bishop**

Quân tướng di chuyển theo các đường chéo.

Hình 12: Cách di chuyển của quân tướng

Đối với quân tướng, sẽ có các offset là { -9, -7, 7, 9} và các ngoại lệ ở cột thứ nhất và cột thứ tám.

Thuật toán sẽ cộng vị trí hiện tại với offset để ra nước đi tiếp theo, nếu không có vật cản và hợp lệ thì sẽ tiếp tục cộng với offset rồi xử lý lại, lúc đầu.

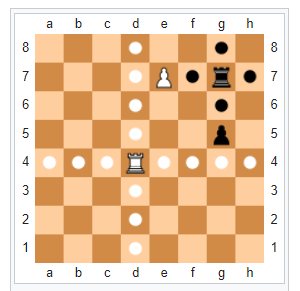
Diagram

Description automatically generated

Hình 13: Thuật toán tìm các nước đi của quân tướng

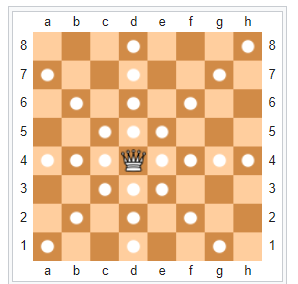
* + 1. **Rook**

Đối với quân xe, thuật toán sử dụng tương tượng với quân tướng, chỉ có khác ở các offset là {-8, -1, 1, 8}



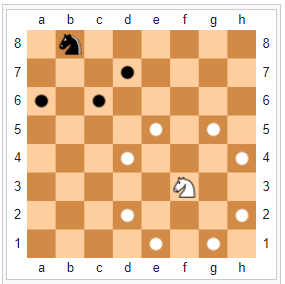
Hình 14: Cách di chuyển của quân xe

* + 1. **Queen**

Cách di chuyển của quân hậu giống với quân xe và quân tướng kết hợp, nên thuật toán sẽ tương tự và offset sẽ là cả 2 offset của quân xe và quân mà {-9, -8, -7, -1, 1, 7, 8, 9}.

Hình 15: Cách di chuyển của quân hậu

* + 1. **Knight**

Đối với quân mã, thuật toán sử dụng sẽ có một khác biệt nhỏ so với các quân trước, offset là {-17, -15, -10, -6. 6, 10, 15 ,17}.

Hình 16: Cách di chuyển của quân mã

Diagram

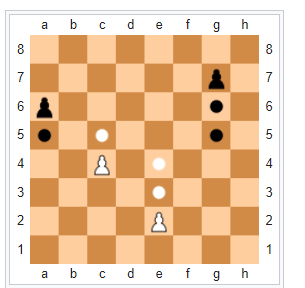
Description automatically generated

Hình 17: Thuật toán tính toán cách di chuyển của quân mã

* + 1. **Pawn**

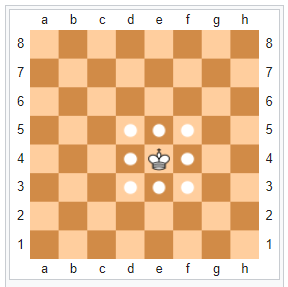
Thuật toán di chuyển của quân tốt sẽ giống với quân mã nhưng khác offset {8, 16, 7 ,9}.

Bên cạnh đó còn 1 số điều kiện về promotion và en passant sẽ được đề cập ở phần sau.

****

Hình 18: Cách di chuyển của quân tốt

* + 1. **King**

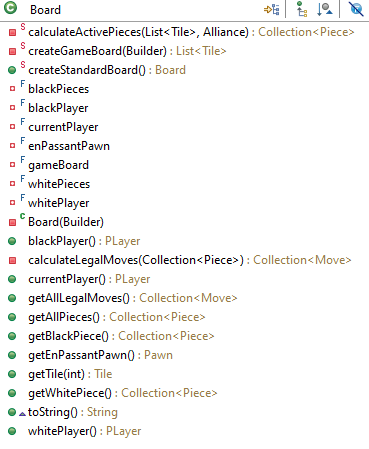
Thuật toán di chuyển của quân vua sẽ giống với quân mã nhưng khác offset {-9, -8, -7, -1, 1, 7, 8, 9}.

Hình 19: Cách di chuyển của quân vua

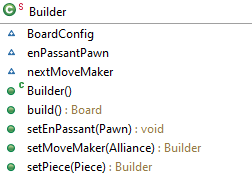
* 1. **Board class**

Board class được dùng để quản lý bàn cờ bao gồm một tập các Tile của bàn cờ, một tập piece quân cờ của người chơi trắng và người chơi đen, và 2 người chơi trắng và đen ( WhitePlayer class và BlackPlayer class)

Trong board class sẽ có 1 class khác là builder class được dùng để xây dựng các quân cờ trên board và tính toán người chơi tiếp theo.



Hình 20: Hierachy của board class

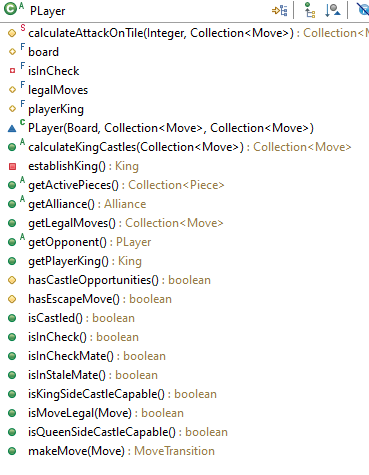


Hình 21: Khởi tạo bàn cờ

Hình 22: Hierachy của builder class

* 1. **Player class**

Player class sẽ chứa bàn cờ, quân vua của mỗi player, một collection các legalMove của người chơi đó và biến isInCheck để kiểm tra xem vua bị chiếu hay không.

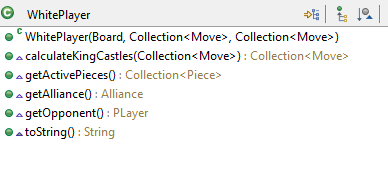


Hình 23: Hierachy của player class

Để có thể tận dụng được tối đa lợi ích của việc lập trình OOP, chúng ta sẽ phân player class thành riêng 2 class khác là: BlackPlayer và WhitePlayer.

2 lớp này sẽ tương tự lớp player và không có thêm biến nào nhưng sẽ override lại hàm toString, và hàm calculateKingCastles (dùng để tính toán các move nhập thành sẽ được đề cập trong phần sau).

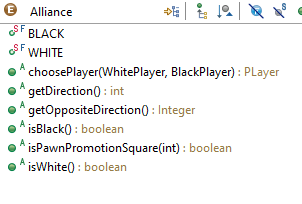
Hình 24: Hierachy của BlackPlayer class



Hình 25: Hierachy của WhitePlayer class

* 1. **Alliance Enum**

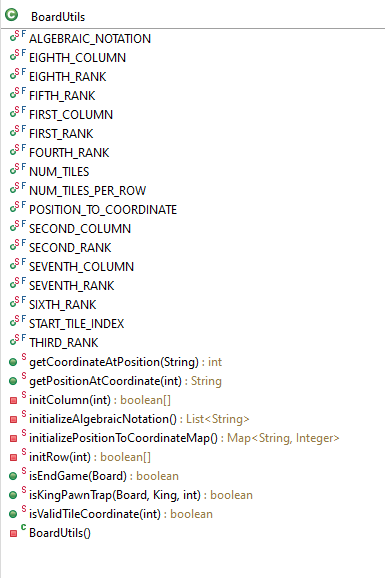
Alliance là enum gồm 2 biến là WHITE và BLACK dùng để biểu thị cho bên đen và bên trắng.



Hình 26: Hierachy của Alliance enum

* 1. **BoardUtils class**

BoardUtils class là class tiện ích bao gồm một số config và một số biến final được dùng để hỗ trợ các class khác.

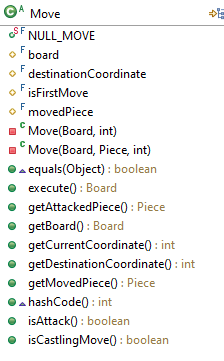


Hình 27: Hierachy của BoardUtils class

* 1. **Move class**

Move class dùng để di chuyển các quân cờ chứa các thông tin gồm bàn cờ (board class), quân cờ cần di chuyển (Piece), vị trí cần đến (int) và biến bool kiểm tra đây phải nước di chuyển đầu tiên không.

Move class sẽ không được gọi đến mà nó sẽ được sử dụng như một class ảo, chỉ gọi đến một số class con của nó.



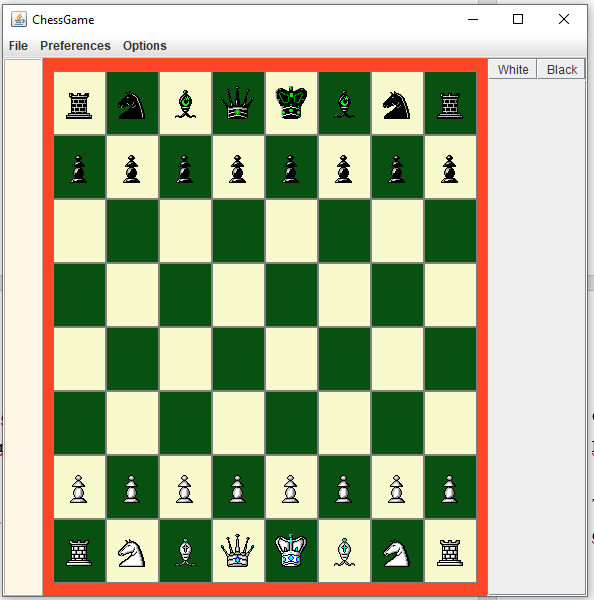
Hình 28: Hierachy của Move class

Hình 29: Hierachy class của Move class

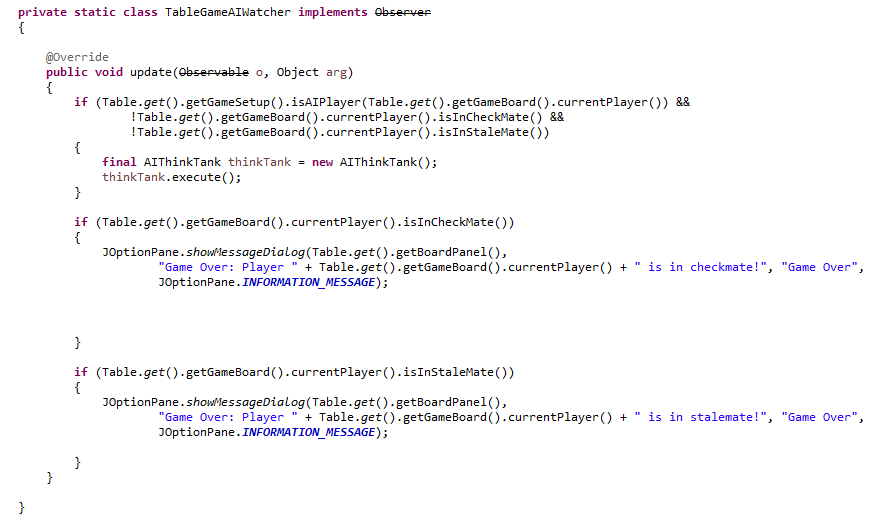
* 1. **GUI**

GUI bao gồm một số class như sau: Table class, GameHistoryPanel, GameSetUp, TakenPiecesPanel, TilePanel.

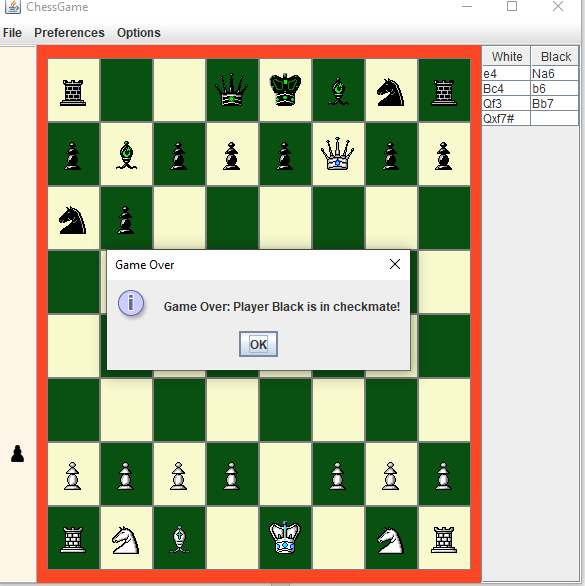
* + 1. **Table class**

Table class sẽ chứa frame chính của trò chơi bao gồm bàn cờ, các cửa sổ menu, bảng các quân cờ bị mất và bảng lịch sử các nước đi.

Hình 30: Giao diện của trò chơi

Table class sẽ được extends từ Observable class dùng kết hợp với interface Observer để có thể theo dõi trò chơi và thông báo game kết thúc.

Hình 31: Observer interface



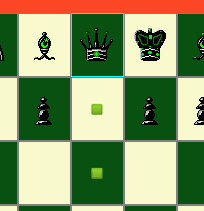
Hình 32: Thông báo trò chơi kết thúc

* + 1. **TilePanel class**

TilePanel class extends từ JPanel class quản lý 1 ô trên bàn cờ .

Hình 33: 1 instance của TilePanel class

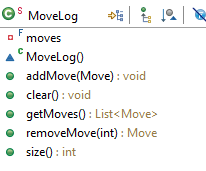
TilePanel class sẽ có thêm mouseListener để cho người dùng có thể chọn quân cờ và chọn các ô di chuyển. Bên cạnh đó còn có 1 số chức năng như high light ô được chọn và high light nước đi của quân cờ đó.



Hình 34: Quân cờ được chọn và nước đi có thể đi của nó

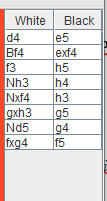
* + 1. **MoveLog class**

MoveLog class là một class chứa tất cả các move đã được thực thi của bàn cờ.



Hình 35: Hierachy của MoveLog class

* + 1. **GameHistoryPanel class**

Class này được dùng để hiểu thị các nước đi đã đi, được extends từ JPanel class.

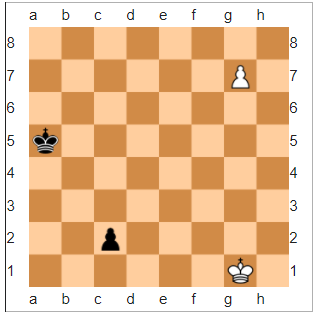
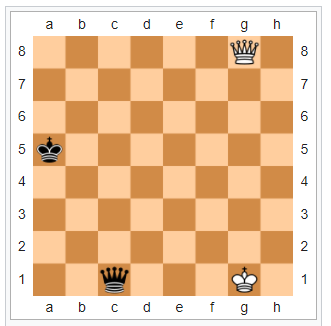
Hình 36: Bảng lịch sử các nước đi

* + 1. **TakenPiecesClass**

Class này được dùng để hiểu thị các quân cờ đã bị mấtm được extends từ JPanel class

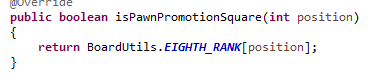
Hình 37: Bảng các quân cờ đã bị ăn mất

1. **Một số vấn đề đặc biệt trong cờ vua**
   1. **Promotion**

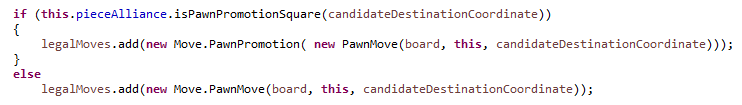
****Trong [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua), **phong cấp** là việc thay thế cho một quân [Tốt](https://vi.wikipedia.org/wiki/T%E1%BB%91t_(c%E1%BB%9D_vua)) bởi một ([Hậu](https://vi.wikipedia.org/wiki/H%E1%BA%ADu_(c%E1%BB%9D_vua)), khi quân Tốt đó đã đi đến hàng cuối phía bên kia của bàn cờ.[[1]](https://vi.wikipedia.org/wiki/Phong_c%E1%BA%A5p_(c%E1%BB%9D_vua)#cite_note-1) Việc phong cấp được thực hiện ngay ở ô mà quân Tốt tiến đến và có hiệu lực lập tức.

Hình 38: Trước khi phong cấp

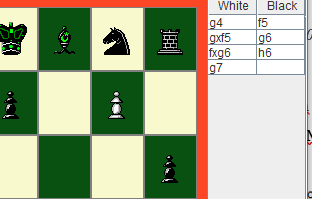
Hình 39: Sau khi phong cấp

Để có thể phong hậu, chúng ta cần kiểm tra liệu quân tốt có đủ điều kiện hay chưa bằng hàm isPawnPromotionSquare, bằng cách kiểm tra vị trí quân tốt đó có nằm ở hàng 1 đối với quân trắng và hang 8 đối với quân đen.

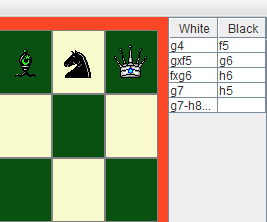
Hình 40: Hàm isPromotionSquare của quân trắng

Nếu kiểm tra đúng điều kiện thì tại class Pawn, chúng ta sẽ thêm move thuộc lớp PawnPromotion thay vì lớp PawnMove thông thường.

Hình 41: Xử lý promtion



Hình 42: Trước khi phong hậu

****

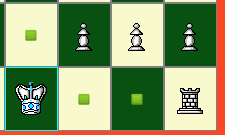
Hình 43: Sau khi phong hậu

* 1. **Castling**

**Nhập thành** là một nước đi đặc biệt trong [trò chơi](https://vi.wikipedia.org/wiki/Tr%C3%B2_ch%C6%A1i) [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua), trong đó [vua](https://vi.wikipedia.org/wiki/Vua_(c%E1%BB%9D_vua)) và một trong hai quân [xe](https://vi.wikipedia.org/wiki/Xe_(c%E1%BB%9D_vua)) tham gia vào việc di chuyển. Khi nhập thành vua di chuyển 2 qua 2 ô về phía quân xe tham gia nhập thành, và sau đó di chuyển quân xe tới ô mà quân vua vừa di chuyển qua sao cho nó nằm ngay bên cạnh quân vua.

Việc xử lý nhập thành khá đơn giản, thay vì đối với những move khác, ta sẽ duyệt qua tất cả các quân cờ rồi thêm vào 1 collection các legal move thì chúng ta sẽ tính toán riêng tại mỗi class con của Player (BlackPlayer và WhitePlayer) vì cách tính toán tại mỗi người chơi là khác nhau. Sau khi tính toán xong KingSideCastleMove và QueenSideCastleMove của mỗi player thì chúng ta sẽ concat các move đó với tập các legal move ban đầu.

Hình 44: Gom tất cả các move thành legalMove trong lớp PLayer

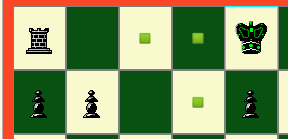
****

Hình 45: Ví dụ trước khi nhập thành

Hình 46: Xử lý castleMove của BlackPlayer class

****

Hình 47: Sau khi nhập thành bên vua quân trắng



Hình 48: Trước khi nhập thành bên quân hậu cờ đen



Hình 49: Sau khi nhập thành bên quân hậu cờ đen

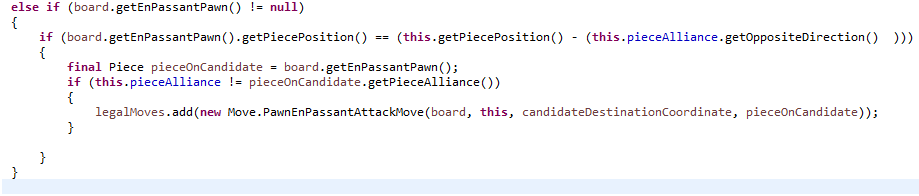
* 1. **En Passant**

**Bắt Tốt qua đường** (hay thường gặp trong một số tài liệu, [tiếng Pháp](https://vi.wikipedia.org/wiki/Ti%E1%BA%BFng_Ph%C3%A1p): **en passant**) là một nước đi trong [cờ vua](https://vi.wikipedia.org/wiki/C%E1%BB%9D_vua). Đây là nước bắt quân đặc biệt mà người chơi thực hiện sau khi [Tốt](https://vi.wikipedia.org/wiki/T%E1%BB%91t_(qu%C3%A2n_c%E1%BB%9D)) của đối phương di chuyển qua ô kiểm soát của Tốt bên phía mình.

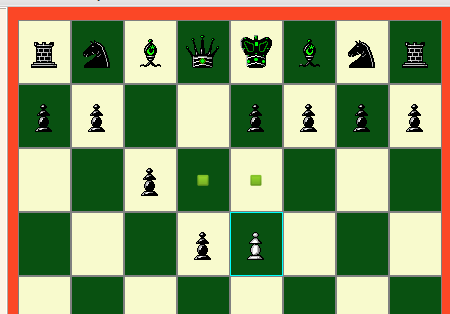
Để xử lý en passant, khi một con tốt thực hiện một bước nhảy (di chuyển 2 ô) thì nó sẽ được set thành enpassant pawn. Sau đó khi tính toán move cho những quân tốt, ta kiểm tra xem sát cạnh bên nó có tốt EnPassant nào không, nếu có thì thêm EnPassant move vào.



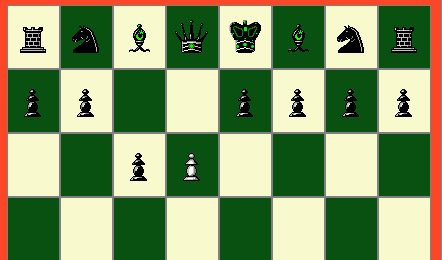
Hình 50: Set enpassant ở class PawnJump



Hình 51: Xử lý enpassant move tại Pawn class



Hình 52: Trước khi enpassant move



Hình 53: Thực thi enpassant move

1. **Thuật toán Minimax**
   1. **Sơ lược về thuật toán**

Graphical user interface, application, Teams

Description automatically generated**Minimax** (còn gọi là **minmax**) là một phương pháp trong [lý thuyết quyết định](https://vi.wikipedia.org/wiki/L%C3%BD_thuy%E1%BA%BFt_quy%E1%BA%BFt_%C4%91%E1%BB%8Bnh) có mục đích là tối thiểu hóa (*mini*mize) [tổn thất](https://vi.wikipedia.org/wiki/H%C3%A0m_t%E1%BB%95n_th%E1%BA%A5t) vốn được dự tính có thể là "tối đa" (*max*imize). Có thể hiểu ngược lại là, nó nhằm tối đa hóa lợi ích vốn được dự tính là *tối thiểu* (**maximin**). Nó bắt nguồn từ [trò chơi có tổng bằng không](https://vi.wikipedia.org/wiki/Tr%C3%B2_ch%C6%A1i_c%C3%B3_t%E1%BB%95ng_b%E1%BA%B1ng_kh%C3%B4ng). Nó cũng được mở rộng cho nhiều trò chơi phức tạp hơn và giúp đưa ra các quyết định chung khi có sự hiện diện của sự không chắc chắn. [2]

Hình 54: Mô hình cây của thuật toán minimax lớp depth bằng 4

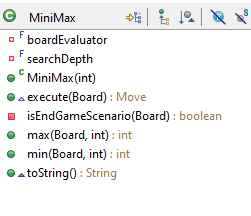
Diagram

Description automatically generated

Hình 55: Visualize thuật toán minimax trong cờ vua

* 1. **Cài đặt**
     1. **MiniMax class**

MiniMax class là class thực thi thuật toán minimax để đưa ra được move tốt nhất trong các move.



Hình 56: Hierachy của MiniMax class

**Diagram

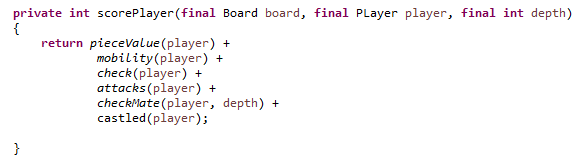
Description automatically generated**

Hình 57: Flowchart của hàm execute để tạo ra move nhờ minimax

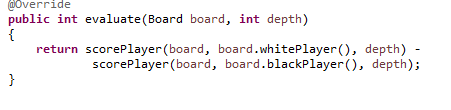
* + 1. **StandardBoardEvaluator class**

Class này dùng để evaluation giá trị của bàn cờ, để thực hiện giải thuật minimax.

Trong thuật toán này, giá trị của board sẽ được đánh giá bằng tổng các giá trị mỗi quân cờ, độ linh động, số nước đi tấn công, có chiếu hay không, và có nhập thành hay không.

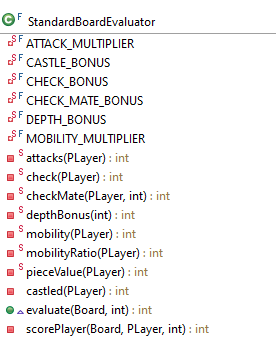


Hình 58: Hàm tính giá trị bàn cờ của mỗi người chơi

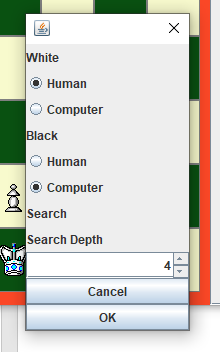


Hình 59: Hàm tính đánh giá giá trị của bàn cờ

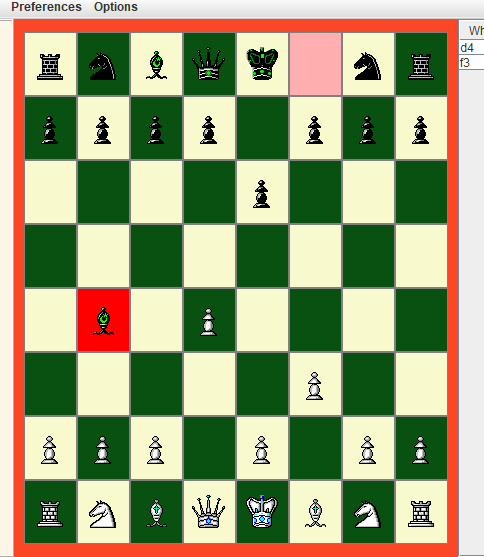
Có thể thấy được rằng, nếu giá trị bàn cờ càng dương, thì quân trắng đang thắng và nếu giá trị bàn cờ càng âm thì quân đen đang thắng.



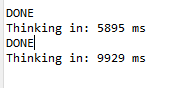
Hình 60: Hierachy của StandardBoardEvaluator class

* 1. **Chạy thử**

Hình 61: Lựa chọn chế độ và depth

****

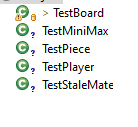
Hình 62: Nước đi của máy sẽ được highligh màu đỏ

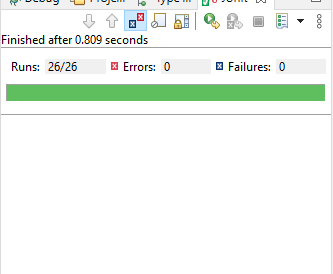


Hình 63: Thời gian chạy của mỗi nước đi

1. **Kiểm thử**

Kiểm thử khả năng hoạt động của game dựa vào framework JUnit

Tạo ra một số test case để kiểm tra game.

****

Hình 64: Các test class

Hình 65: Kết quả chạy thử tất cả test class

****

Hình 66: Một test case trong các tập test

1. **Tổng kết**
   1. **Kết quả đạt được**

* Hiểu và vận dụng cấc kiến thức được họp trên lớp về cách lập trình hướng đối tượng và GUI trong JAVA.
* Học được quy trình xây dựng một phần mềm cơ bản.
* Xây dựng được game thỏa mãn được một số yêu cầu.
  1. **Ưu điểm**
* **Về sản phẩm**
* Học được cách thiết kế một game, thuật toán AI cơ bản
* Áp dụng được Immutability Enhancement trong việc thiết kế theo hướng đối tượng
* Độ chính xác: trò chơi chạy linh hoạt, chính xác. Đáp ứng được yêu cầu người dùng.
* **Về việc tổ chức làm việc**
* Tất cả các công việc chỉ có một người làm, nên việc xây dựng chương trình đơn giản, không cần phải quản lý.
  1. **Hạn chế**
* **Về sản phẩm**
* Thuật toán AI ở những lớp đầu có thể chạy hiểu quả nhưng khi gia tăng depth thì độ phức tạp ngày càng lớn dẫn đến thời gian chạy rất lâu.
* GUI ở mức cơ bản, cần phải gia tăng độ mượt cũng như trải nghiệm người dùng.
* **Về việc tổ chức làm việc**
* Vì chỉ có một người làm nên công việc tương đối mặc dù cho khả năng làm việc nhanh hơn.
  1. **Hướng phát triển**
* Cải tiến thuât toán AI, tìm hiểu về FEN format.
* Áp dụng CSDL cho game.
* Cải tiến về GUI, đồng thời thêm một số tính năng mới cho game.

1. **Tài liệu tham khảo**

[1]Wikipedia contributors. (2021, June 18). Chess. In *Wikipedia, The Free Encyclopedia*. Retrieved 00:45, June 26, 2021, from <https://en.wikipedia.org/w/index.php?title=Chess&oldid=1029147167>

[2] Wikipedia contributors. (2021, May 2). Minimax. In *Wikipedia, The Free Encyclopedia*. Retrieved 00:50, June 26, 2021, from <https://en.wikipedia.org/w/index.php?title=Minimax&oldid=1021043424>

[3] M. Wahid and A. Almalaise, "JUnit framework: An interactive approach for basic unit testing learning in Software Engineering," 2011 3rd International Congress on Engineering Education (ICEED), 2011, pp. 159-164, doi: 10.1109/ICEED.2011.6235381.