A picture containing antenna, swing, lamp, riding





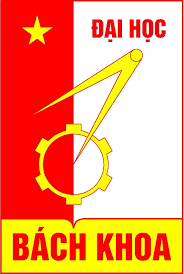




Description automatically generatedTRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────



**BÀI TẬP LỚN**

MÔN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

**XÂY DỰNG TRÒ CHƠI SUDOKU**

**THEO MÔ HÌNH MVC**

**Giáo viên hướng dẫn:** ThS.Nguyễn Mạnh Tuấn

**Lớp:** 117504

**Sinh viên thực hiện:**

|  |  |
| --- | --- |
| **Họ và Tên** | **MSSV** |
| Nguyễn Hoài Nam | 20173277 |
| Đặng Quang Anh | 20172942 |
| Bo Sayon | 20170286 |
| Hoàng Trung Đức | 20173027 |

**MỤC LỤC**

[LỜI NÓI ĐẦU 3](#_Toc49474035)

[PHÂN CÔNG THÀNH VIÊN TRONG NHÓM 4](#_Toc49474036)

[CHƯƠNG 1. KHẢO SÁT, ĐẶC TẢ YÊU CẦU BÀI TOÁN 5](#_Toc49474037)

[1.1. Mô tả yêu cầu bài toán 5](#_Toc49474038)

[1.2. Biểu đồ use case 6](#_Toc49474039)

[1.2.1 Biểu đồ use case tổng quan 6](#_Toc49474040)

[1.2.2 Biểu đồ use case phân rã mức 2 7](#_Toc49474041)

[CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ BÀI TOÁN 10](#_Toc49474042)

[2.1. Thiết kế cơ sở dữ liệu cho chương trình 10](#_Toc49474043)

[2.2 Biểu đồ trình tự 11](#_Toc49474044)

[2.2.1 Biểu đồ trình tự chơi 11](#_Toc49474045)

[2.2.2 Biểu đồ trình tự Undo 11](#_Toc49474046)

[2.3 Biểu đồ lớp 12](#_Toc49474047)

[2.4 Thiết kế chi tiết lớp 13](#_Toc49474048)

[2.4.1 Model 13](#_Toc49474049)

[2.4.2 View 14](#_Toc49474050)

[2.4.3 Controller 15](#_Toc49474051)

[CHƯƠNG 3.CÔNG NGHỆ VÀ THUẬT TOÁN SỬ DỤNG 16](#_Toc49474052)

[3.1 Công nghệ sử dụng 16](#_Toc49474053)

[3.1.1 Sử dụng ngôn ngữ Java 16](#_Toc49474054)

[3.1.2. Sử dụng mô hình MVC 16](#_Toc49474055)

[3.2 Thuật toán sử dụng 16](#_Toc49474056)

[3.2.1. Thuật toán kiểm tra Sudoku 9x9: 16](#_Toc49474057)

[3.2.2. Thuật toán sinh Sudoku 9x9: 17](#_Toc49474058)

[3.2.3. Thuật toán tự động giải Sudoku 9x9: 17](#_Toc49474059)

[CHƯƠNG 4. XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA 19](#_Toc49474060)

[4.1 Kết quả chương trình minh họa 19](#_Toc49474061)

[4.2 Giao diện chương trình 20](#_Toc49474062)

[4.3 Kiểm thử các chức năng đã thực hiện 20](#_Toc49474063)

[Kiểm thử cho chức năng easy 21](#_Toc49474064)

[Kiểm thử cho chức năng medium 22](#_Toc49474065)

[Kiểm thử cho chức năng hard 22](#_Toc49474066)

[Kiểm thử cho chức năng undo 24](#_Toc49474067)

[Kiểm thử cho chức năng redo 25](#_Toc49474068)

[Kiểm thử cho chức năng save 26](#_Toc49474069)

[Kiểm thử cho chức năng load 27](#_Toc49474070)

[Kiểm thử cho chức năng solve 27](#_Toc49474071)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 28](#_Toc49474072)

[TÀI LIỆU THAM KHẢO 28](#_Toc49474073)

[PHỤ LỤC 28](#_Toc49474074)

LỜI NÓI ĐẦU

Với môn học lập trình hướng đối tượng giúp sinh viên rèn luyện được kỹ năng làm việc nhóm, kỹ năng phân tích thiết kế chương trình, kỹ năng lập trình hướng đối tượng, kỹ năng tìm hiểu các tài liệu để có thể hoàn thành yêu cầu đề ra. Với đề tài là viết chương trình trò chơi Sudoku giúp nhóm chúng em hoàn thiện thêm được các kỹ năng kể trên. Do mới làm quen với lập trình hướng đối tượng và kì hè rút ngắn nên chương trình viết ra còn nhiều hạn chế, chưa được hoàn thiện tốt. Chúng em mong thầy sẽ đánh giá tận tình để chúng em rút ra những bài học kinh nghiệm để chúng em có thể hoàn thiện thêm kiến thức và kỹ năng của mình. Chúng em xin chân thành cảm ơn thầy!

PHÂN CÔNG THÀNH VIÊN TRONG NHÓM

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Công việc thực hiện** |
| Nguyễn Hoài Nam | 20173277 | Tìm hiểu giải thuật,viết chương trình sơ lược, làm báo cáo |
| Đặng Quang Anh | 20172942 | Tìm hiểu giải thuật phân tích chương trình theo mô hình MVC |
| Bo Sayon | 20170286 | Tìm hiểu giải thuật, kiểm thử |
| Hoàng Trung Đức | 20173027 | Hoàn thành code theo mô hình MVC |

# KHẢO SÁT, ĐẶC TẢ YÊU CẦU BÀI TOÁN

## Mô tả yêu cầu bài toán

Sử dụng mô hình MVC. Hệ thống phải thuần túy là hướng đối tượng, không chấp nhận các cách tiếp cận khác.

Hoàn thành các thao tác sau:

- Trò chơi cung cấp cho người chơi các chức năng cơ bản:

+ Tự động sinh ô số sudoku với các mức độ khó/dễ khác nhau: mức độ dễ

nhất 35/81 ô có sẵn giá trị. Mức độ trung bình: 27/81 ô có sẵn giá trị. Chỉ

rõ phương pháp đánh giá mức độ khó dễ sử dụng trong chương trình (phụ

thuộc nhiều tham số: số lượng các số có sẵn, phân bố các ô, ...)

+ Tổ chức lưu trữ thông tin của các ô số cho chức năng tạm dừng / tiếp tục (save/load)

+ Xây dựng chức năng undo/redo trong quá trình chơi.

+ Trong quá trình người chơi giải ô số chương trình có chức năng hỗ trợ:

loại bỏ các số không phù hợp trong ô và thay đổi sau mỗi bước chơi, kiểm

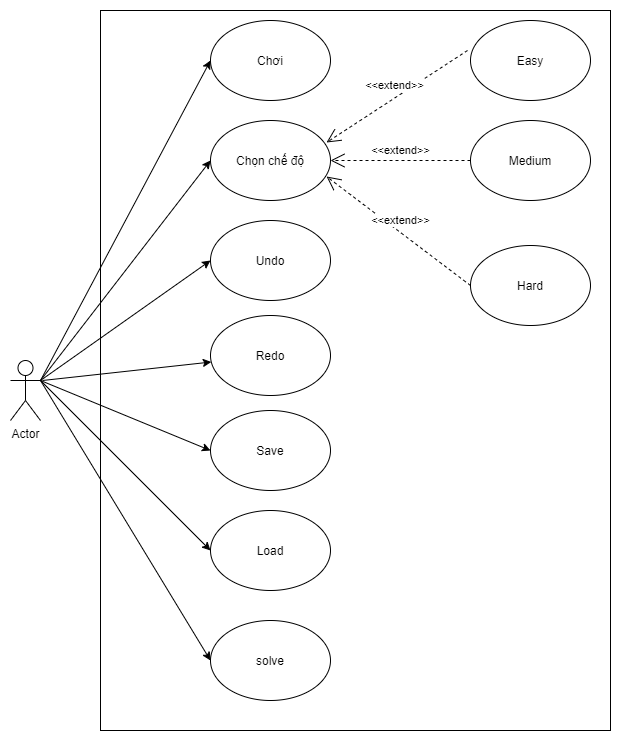
tra xem một ô người dùng nhập có chính xác,...

+ Xây dựng chức năng cho người dùng tự nhập một ô số và chương trình sẽ tính toán để đưa ra tất cả các đáp án có thể.

+ Thiết kế giao diện đồ họa cho trò chơi.

## Biểu đồ use case

### 1.2.1 Biểu đồ use case tổng quan



### 1.2.2 Biểu đồ use case phân rã mức 2

Ảnh có chứa văn bản

Mô tả được tạo tự động

**1.2.3 Đặc tả use case**

|  |  |
| --- | --- |
| **Tên use case** | **Chơi** |
| Mô tả | Người chơi chọn các con số để điền vào những ô còn trống. |
| Tác nhân | Người chơi. |
| Sự kiện kích hoạt | Chạy chương trình. |
| Usecase liên quan | Không có. |
| Tiền điều kiện | Không có. |
| Hậu điều kiện | Hệ thống sẽ hiển thị số mà người chơi vửa chọn ở ô người chơi vừa chọn. |
| Luồng sự kiện chính | 1. Người chơi chọn con số thích hợp vào ô còn trống. 2. Hệ thống sẽ hiển thị số mà người chơi vừa chọn ở ô người chơi vừa chọn. |
| Luồng sự kiện phụ | Không có. |

|  |  |
| --- | --- |
| **Tên use case** | **Chọn chế độ** |
| Mô tả | Người chơi chọn một trong các chế độ để chơi. |
| Tác nhân | Người chơi. |
| Sự kiện kích hoạt | Chạy chương trình. |
| Usecase liên quan | Không có. |
| Tiền điều kiện | Không có. |
| Hậu điều kiện | Hệ thống chuyển sang chế độ mà người chơi đã chọn. |
| Luồng sự kiện chính | 1. Người chơi chọn một trong các chế độ của trò chơi. 2. Hệ thống chuyển sang chế độ mà người chơi đã chọn. |
| Luồng sự kiện phụ | Không có |

|  |  |
| --- | --- |
| **Tên use case** | **Undo** |
| Mô tả | Quay lại bước ngay trước đó |
| Tác nhân | Người chơi. |
| Sự kiện kích hoạt | Chọn vào Undo. |
| Usecase liên quan | Chơi. |
| Tiền điều kiện | Người chơi đã chơi ít nhất một ô. |
| Hậu điều kiện | Bỏ chọn số ô vừa chơi gần nhất. |
| Luồng sự kiện chính | 1. Người chơi chọn Undo. 2. Hệ thống bỏ chọn số ô vừa chơi gần nhất. |
| Luồng sự kiện phụ | Không có. |

|  |  |
| --- | --- |
| **Tên use case** | **Redo** |
| Mô tả | Đảo ngược lại của Undo |
| Tác nhân | Người chơi. |
| Sự kiện kích hoạt | Chọn vào Redo. |
| Usecase liên quan | Undo. |
| Tiền điều kiện | Người chơi vừa chọn Undo. |
| Hậu điều kiện | Đảo ngược lại bước Undo. |
| Luồng sự kiện chính | 1. Người chơi chọn Redo. 2. Hệ thống đảo ngược lại bước Undo ngay trước đó. |
| Luồng sự kiện phụ | Không có. |

|  |  |
| --- | --- |
| **Tên use case** | **Save** |
| Mô tả | Lưu trạng thái hiện tại của hệ thống |
| Tác nhân | Người chơi |
| Sự kiện kích hoạt | Chọn vào Save |
| Usecase liên quan | Không có |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Trạng thái của hệ thống sẽ được lưu lại. |
| Luồng sự kiện chính | 1. Người chơi chọn Save 2. Hệ thống lưu lại trạng thái lúc đó |
| Luồng sự kiện phụ | Không có |

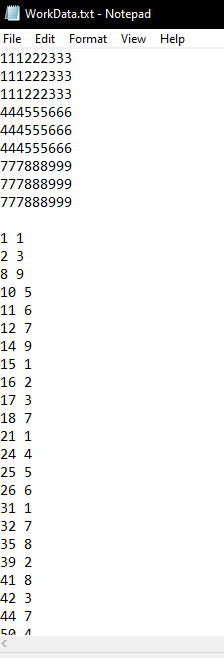
|  |  |
| --- | --- |
| **Tên use case** | **Load** |
| Mô tả | Quay lại trạng thái đã lưu trước đó |
| Tác nhân | Người chơi |
| Sự kiện kích hoạt | Chọn vào load |
| Usecase liên quan | Save |
| Tiền điều kiện | Đã lưu một trạng thái trước đó |
| Hậu điều kiện | Đưa đến trạng thái đã lưu |
| Luồng sự kiện chính | 1. Người chơi chọn Load. 2. Hệ thống đưa đến trạng thái đã lưu trước đó |
| Luồng sự kiện phụ | Không có |

|  |  |
| --- | --- |
| **Tên use case** | **Solve** |
| Mô tả | Chọn vào solve |
| Tác nhân | Người chơi |
| Sự kiện kích hoạt | Chọn vào solve |
| Usecase liên quan | Không có. |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Đưa ra kết quả cuối cùng |
| Luồng sự kiện chính | 1. Người chơi chọn solve 2. Hệ thống đưa ra kết quả cuối cùng |
| Luồng sự kiện phụ | Không có |

# CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ BÀI TOÁN

## 2.1. Thiết kế cơ sở dữ liệu cho chương trình

* Để lưu trữ dữ liệu của Sudoku ta sử dụng mảng hai chiều 9x9.
* Để sử dụng stack ta sử dụng ArrayList trong Java(giống với danh sách liên kết).
* Về SaveGame: lưu trữ shape(hình dạng ô) và giá trị của các Cell(1 trong số 81 ô) vào file txt như hình minh họa dưới đây:



* + Ma trận 9x9 đầu tiên lưu shape (hình dạng của 1 bảng trong 9 bảng). Với mỗi giá trị khác nhau tạo thành một hình có thể là bất cứ hình dạng nào. Ở đây chúng em sử dụng đơn giản nhất là hình vuông 3x3 Cell
  + Tiếp theo là lưu giá trị đã đặt của các Cell với cú pháp: [chỉ\_số\_hàng \* 9 + chỉ\_số\_cột] [giá\_trị\_của\_cell]. Nếu một Cell không có giá trị thì sẽ không lưu vào file.

## 2.2 Biểu đồ trình tự

### 2.2.1 Biểu đồ trình tự chơi

Ảnh có chứa ảnh chụp màn hình

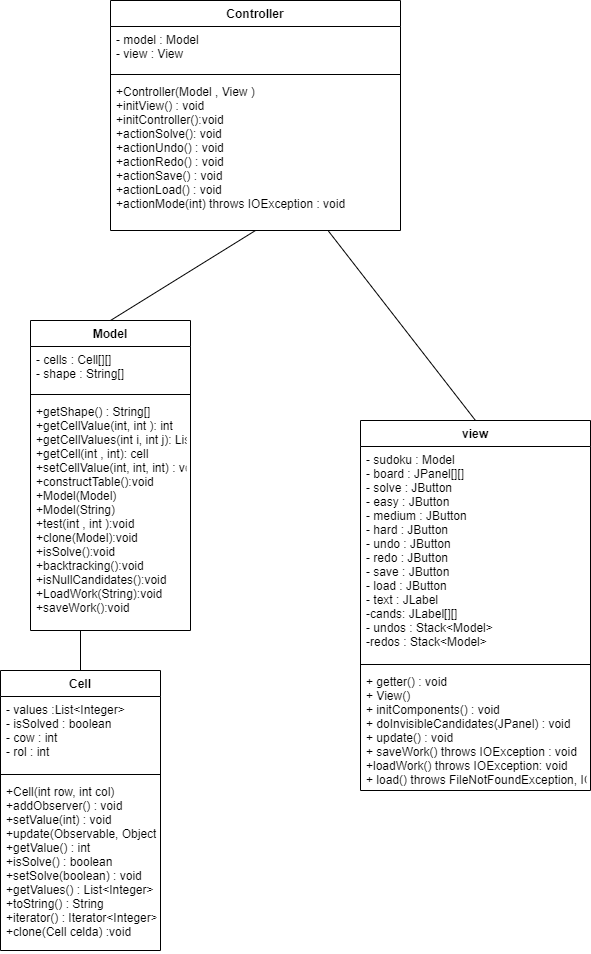
Mô tả được tạo tự động

### 2.2.2 Biểu đồ trình tự Undo

Ảnh có chứa ảnh chụp màn hình

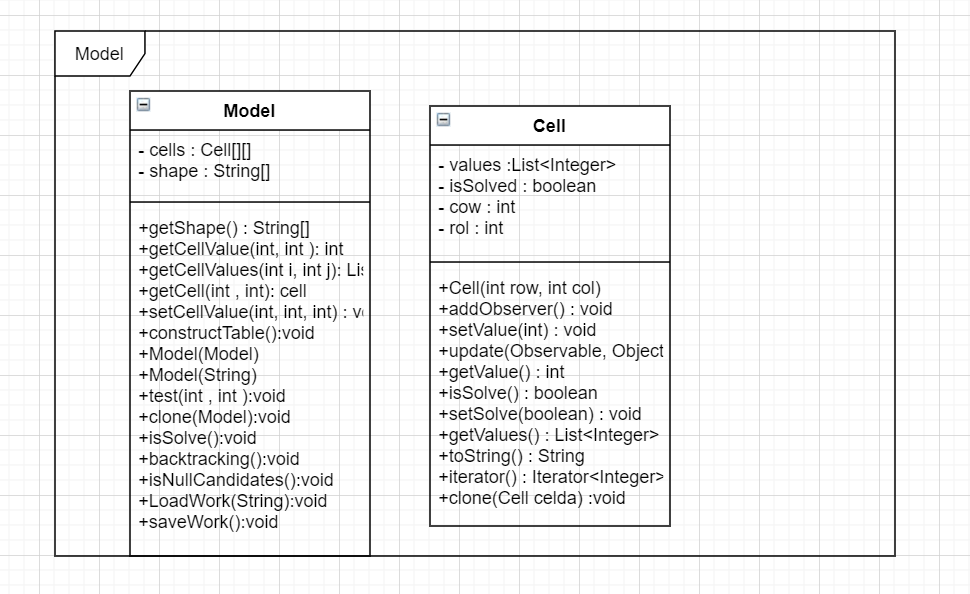
Mô tả được tạo tự động

## 2.3 Biểu đồ lớp

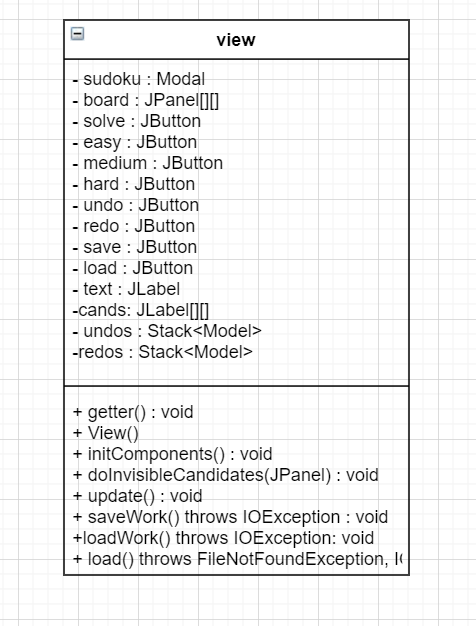


## 2.4 Thiết kế chi tiết lớp

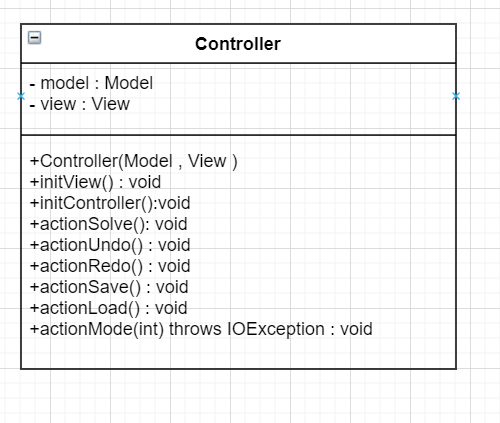
### 2.4.1 Model



### 2.4.2 View



### 2.4.3 Controller



# CHƯƠNG 3.CÔNG NGHỆ VÀ THUẬT TOÁN SỬ DỤNG

## Công nghệ sử dụng

### 3.1.1 Sử dụng ngôn ngữ Java

* Chúng em sử dụng ngôn ngữ Java để thực hiện bài tập lớn này. Những công nghệ bọn em sử dụng bao gồm :
* Sử dụng thư viện javax.swing, java.awt để xây dựng giao diện của trò chơi.
* Các kiến thức bọn em đã tìm hiểu quả để tạo lên ứng dụng đó là:
* Các kiến thức về lập trình hướng đối tượng: trừu tượng hóa, kết tập, kế thừa, đóng gói, đa hình…

### 3.1.2. Sử dụng mô hình MVC

* Model – View – Controller (MVC) là một mẫu thiết kế nhằm mục tiêu chia tách phần Giao diện và Code để dễ quản lý, phát triển và bảo trì. MVC chia ứng dụng phần mềm ra làm 3 phần có tương tác với nhau là Model(Dữ liệu), View(Giao diện), Controller( Code điều khiển tương tác giữa Model và View cũng như nghiệp vụ (Business)).

## 3.2 Thuật toán sử dụng

### 3.2.1. Thuật toán kiểm tra Sudoku 9x9:

* Một Sudoku 9x9 là được giải nếu ta kiểm tra các hàng, các cột, các khối chứa 3x3 ô đều được điền đầy các số từ 1 đến 9.
* Đầu vào của thuật toán là mảng Sudoku 9x9.
* Đầu ra là true hoặc false.
* Ý tưởng là duyệt qua 81 Cell bằng 2 vòng lặp for sau đó với mỗi Cell ta kiểm tra nếu:
  + Số giá trị có thể điền của Cell đó lớn hơn 1 hoặc bằng 0
  + Ô đó chưa được điền giá trị nào

🡪 trả về false

còn không thuật toán sẽ trả về true.

### 3.2.2. Thuật toán sinh Sudoku 9x9:

* Đầu vào :số nguyên Cell muốn điền sẵn vị trí.
* Đầu ra : mảng ô số Sudoku 9x9 được sinh ra với số Cell được điền sẵn vị trí cho trước.
* Thuật toán sẽ sinh ra một Sudoku với lời giải ngẫu nhiên, sau đó xóa đi giá trị của các Cell ngẫu nhiên sao cho số Cell có giá trị điền sẵn bằng với số nguyên cho trước. Cụ thể thuật toán gồm các bước sau:
  + Tạo một Sudoku mới với toàn bộ giá trị của Cell bằng 0.
  + Sinh ngẫu nhiên 3 số: row(chỉ số hàng của Cell), col(chỉ số cột của Cell), value(giá trị của Cell) sau đó đặt giá trị “value” + 1 cho Cell ở hàng “row” cột “col” rồi dung thuật toán Backtracking để giải Sudoku vừa tạo với một giá trị cho sẵn để đảm bảo sự ngẫu nhiên và ít bị lặp lại lời giải Sudoku.
  + Lấy 81 trừ đi tham số truyền vào để có được số Cell chưa điền sẵn.
  + Tạo vòng lặp for “n” lần với “n” là số Cell chưa điền sẵn:
    - Tiếp tục sinh ngẫu nhiên 2 số nguyên: “row” và “col” cho đến khi nào giá trị của Cell tại hàng “row” và cột ”col” không bằng 0
    - Sau khi kết thúc vòng lặp ta đã có tọa độ của Cell chưa bị xóa. Để xóa giá trị của Cell ta chỉ cần đặt giá trị của Cell đó bằng 0 và đặt trạng thái điền giá trị là false.
* Kết thúc vòng lặp ta được một Sudoku với số cho trước Cell được điền sẵn vị trí. Đặt Sudoku mới bằng Sudoku hiện tại, update lại giao diện là xong.

### 3.2.3. Thuật toán tự động giải Sudoku 9x9:

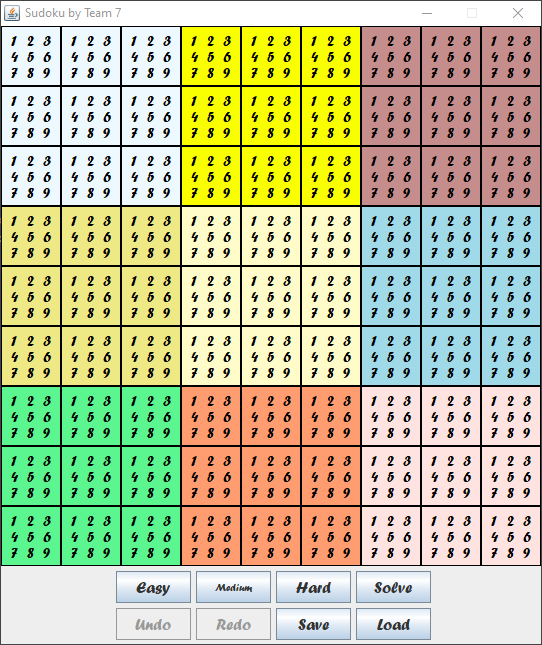
* Đầu vào : Sudoku chưa được giải.
* Đầu ra: Sudoku đã được giải.
* Thuật toán xây dựng dựa trên nguyên lý vét cạn và thuật toán quay lui.
* Mô tả thuật toán backtracking():
  + Duyệt qua 81 Cell bằng 2 vòng lặp for và kiểm tra với mỗi Cell:
    - Nếu Cell không có giá trị nào để điền (sai quy luật) thì trả về false
    - Nếu Cell chưa được điền giá trị (giá trị bằng 0)
      * Duyệt qua các giá trị có thể điền của Cell đó, tạo một Sudoku mới bằng với Sudoku hiện tại và lần lượt đặt thử giá trị có thể điền cho Cell. Với mỗi giá trị, nếu gọi lại backtracking() trả về true thì đặt Sudoku hiện tại bằng Sudoku mới tạo và trả về true.
      * Duyệt hết các giá trị mà không có đáp án nào đúng thì trả về false
  + Duyệt hết 81 Cell nếu toàn bộ đều điền đúng giá trị rồi thì trả về true, lúc này Sudoku hiện tại đã được đặt giá trị đúng cho toàn bộ các Cell.
* Đánh giá thuật toán: thuật toán nêu ra vì dựa vào nguyên lý vét cạn nên sẽ tốn khá nhiều thời gian xử lý các trường hợp nhưng đã được giảm bớt phần lớn do trong quá trình đặt giá trị cho một Cell, chương trình sẽ gọi hàm update() của lớp Cell và xóa bỏ đi các giá trị dự kiến của các cell cùng hàng/cột/shape khiến cho các trường hợp tiếp theo phải xét ít hơn rất nhiều.

# CHƯƠNG 4. XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA

## Kết quả chương trình minh họa

|  |  |
| --- | --- |
| **Yêu cầu** | **Đánh giá mức độ hoàn thành(%)** |
| Tạo game Sudoku mới với 2 mức độ dễ, trung bình. | 100% |
| Tổ chức lưu trữ thông tin ô số cho các chức năng tạm dừng (tiếp tục), save(load) | 70% |
| Xây dựng chức năng undo và redo | 100% |
| Xây dựng chức năng tự động loại bỏ các ô số cho chương trình | 100% |
| Xây dựng chức năng gợi ý cho người chơi | 100% |
| Xây dựng chức năng tự động giải Sudoku | 100% |
| Chương trình có giao diện đồ họa | 100% |

## Giao diện chương trình



* Nút easy, medium, hard: chế độ của game.
* Nút undo, redo: quay lại lượt trước và quay về lượt sau.
* Nút save: lưu lại.
* Nút load: tải lại.
* Nút solve: tìm kiếm lời giải.

## Kiểm thử các chức năng đã thực hiện

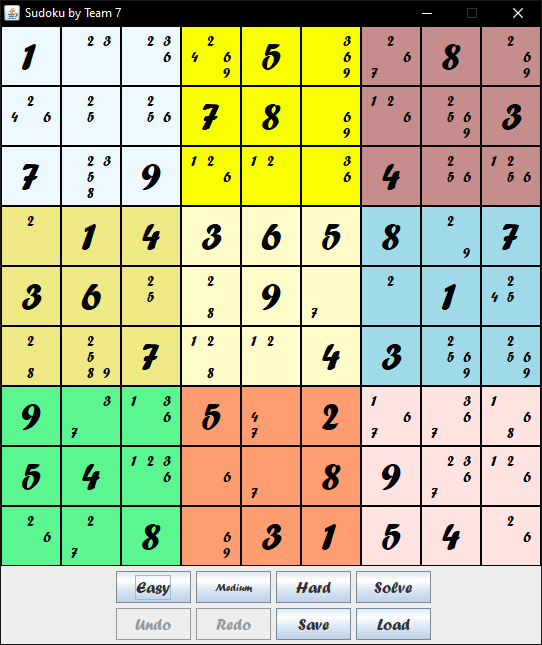
Các chức năng trong chương trình:

* easy
* medium
* hard
* solve
* undo
* redo
* save
* load

### Kiểm thử cho chức năng easy

easy:

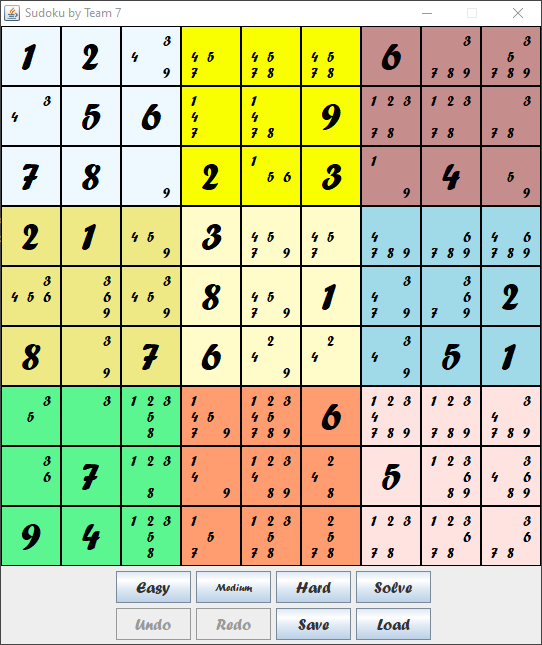
* Khi click vào nút easy ta được giao diện:



### Kiểm thử cho chức năng medium

medium:

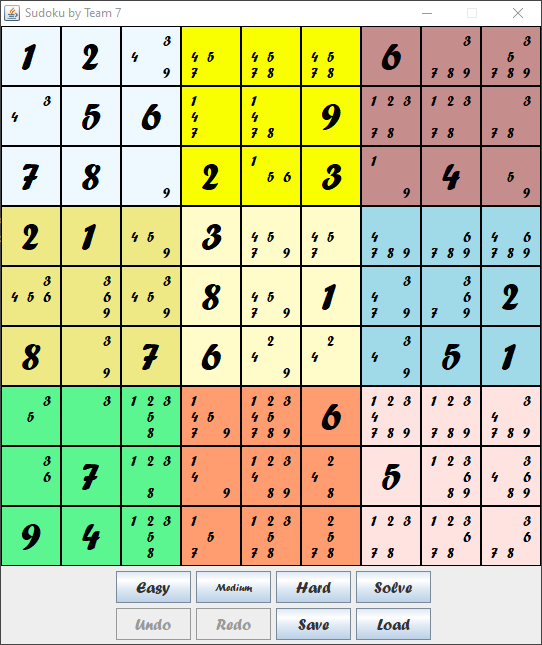
* Khi click vào nút medium sẽ hiện ra giao diện:



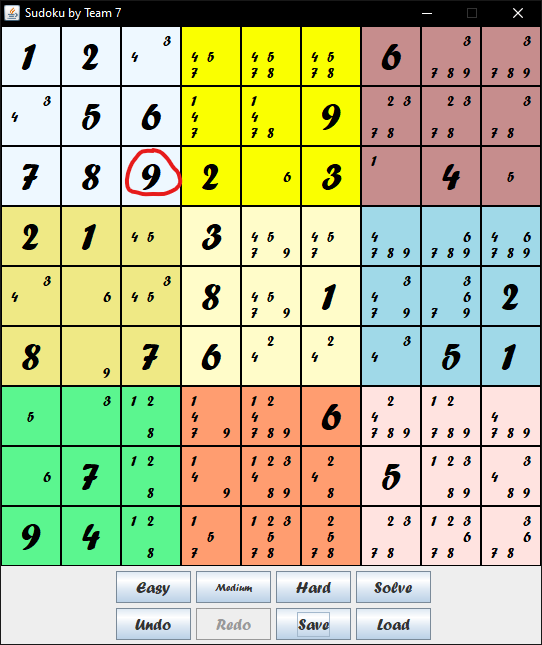
### Kiểm thử cho chức năng hard

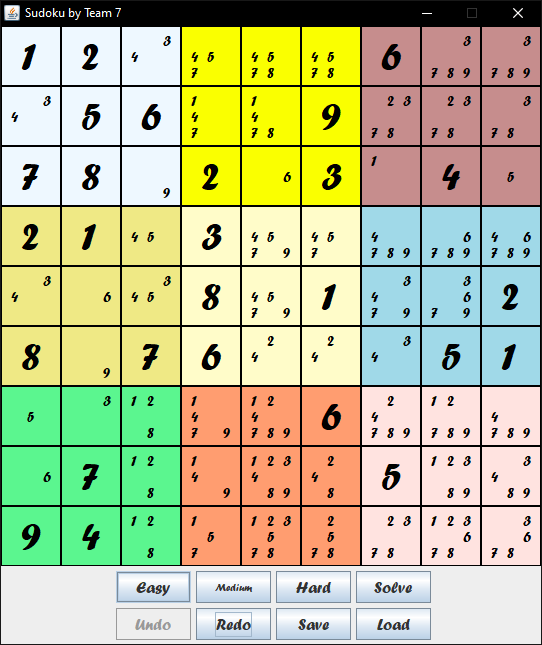
Hard:

* Khi click vào nút hard xuất hiện giao diện:



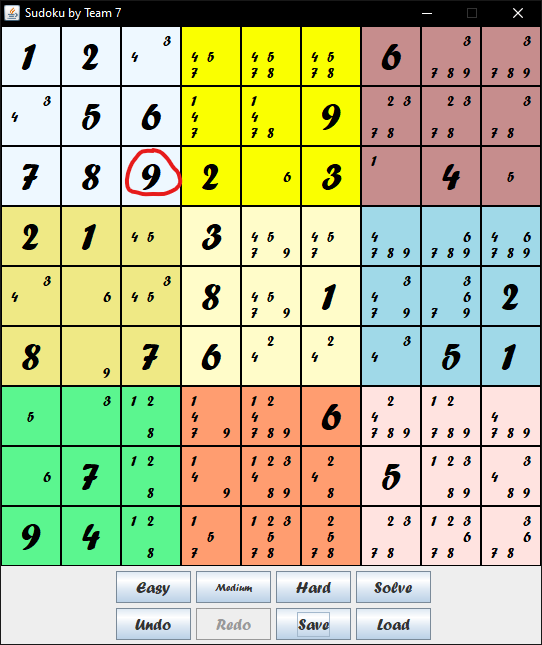
### Kiểm thử cho chức năng undo



* Sau khi nhấn undo :
* 

### Kiểm thử cho chức năng redo

Khi nhấn vào nút redo, yêu cầu phải nhấn nút undo trước



### Kiểm thử cho chức năng save

(Đã kiểm thử)

### Kiểm thử cho chức năng load

(Đã kiểm thử)

### Kiểm thử cho chức năng solve

Khi click vào nút solve

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

* Sau bài tập này, điều mà chúng em thu được là kinh nghiệm lập trình trong Java và sử dụng mô hình MVC.
* Hiểu được tầm quan trọng của việc tự tìm và đọc tài liệu và kĩ năng làm việc nhóm.
* Chưa tối ưu được các thuật toán trong chương trình.
* Giao diện chương trình còn đơn giản.
* Để khắc phục các nhược điểm trên chúng em cần tìm hiểu và học hỏi nhiều hơn để có thể hoàn thành tốt hơn.

# TÀI LIỆU THAM KHẢO

* Programing Sudoku – WeiMengLee
* Java Swing 2nd – O.Relly Thinking in Java.
* Tài liệu trên mạng internet.

# PHỤ LỤC

* Link hithub: <https://github.com/ducht2699/SudokuMVCJavaSwing>