**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

****

BÀI TẬP LỚN MÔN HỌC

**CÔNG NGHỆ JAVA**

**Giảng viên hướng dẫn:** Vũ Huấn

**Sinh viên thực hiện:** Trương Trường An – 211203053

Dương Quốc Anh – 211203702

**Lớp**: Công nghệ thông tin 6 – K62 (N08)

**Đề tài**: Lập trình trò chơi Sudoku

**Mục lục**

[I. Giới thiệu cơ bản 3](#_Toc133985801)

[1. Giới thiệu về trò chơi 3](#_Toc133985802)

[2. Cách chơi Sudoku 3](#_Toc133985803)

[3. Mẹo chơi Sudoku 3](#_Toc133985804)

[4. Các tiện ích của trò chơi 3](#_Toc133985805)

[II. Thiết kế trò chơi 4](#_Toc133985806)

[1. Ý tưởng thiết kế: 4](#_Toc133985807)

[2. Thiết kế 4](#_Toc133985808)

[III. Hướng dẫn cài đặt: 8](#_Toc133985809)

**Lời cảm ơn**

Lời đầu tiên cho phép em xin cảm ơn thầy Vũ Huấn đã giảng dạy để bổ trợ cho chúng em kiến thức về công nghệ java, cũng như những góp ý của các lần review bài tập lớn để chúng em rút được kinh nghiệm và cải thiện bài tập lớn hoàn thiện hơn. Nhóm 09 chúng em xin cảm ơn thầy đã cải thiện kiến thức và chúc thầy luôn mạnh khỏe, hạnh phúc để tiếp tục con đường giảng dạy những lớp sinh viên tiếp theo.

# I. Giới thiệu cơ bản

## 1. Giới thiệu về trò chơi

Sudoku là một trò chơi nổi tiếng của Nhật Bản, Sudoku dựa trên việc sắp xếp các con số theo logic. Là một trò chơi logic, Sudoku không đòi hỏi bất kì phép tính hay kĩ năng đặc biệt nào; tất cả những gì cần thiết là trĩ não và sự tập trung.

## 2. Cách chơi Sudoku

Mục tiêu của Sudoku là điền các chữ số vào một ma trận 9x9 ô kẻ sao cho mỗi cột, mỗi hàng và mỗi phần 3x3 ô kẻ đều chữa các chữ số từ 1 đến 9. Đầu tiên, lưới 9x9 sẽ có một số ô điền sẵn. Nhiệm vụ của người chơi là sử dụng logic để điền các chữ số còn thiếu và hoàn thành ma trận. Một lựa chọn được coi là sai nếu:

* Có bất kì hàng nào chứa nhiều hơn một số trùng lại trong các chữ số từ 1 đến 9.
* Có bất kì cột nào chứa nhiều hơn một số trùng lại trong các chữ số từ 1 đến 9.
* Bất kì ma trận 3x3 ô nào chứa nhiều hơn một số trùng lại trong các chữ số từ 1 đến 9.

## 3. Mẹo chơi Sudoku

Sudoku là một trò chơi giải đố thú vị khi bạn đã quen với nó. Đồng thời, học chơi Sudoku có thể hơi đáng sợ cho người mới băt đầu. Vì vậy, nếu bạn là một người chơi mới hoàn toàn, dưới đây là một vài mẹo chơi Sudoku mà bạn có thể sử dụng để cải thiện kĩ năng chơi Sudoku của mình

* Mẹo 1: Tìm các hàng, cột của các phần 3x3 ô có chứa 5 số hoặc nhiều hơn. Giải các ô trống còn lại, thử các con số chưa được sử dụng. Trong nhiều trường hợp, bạn sẽ nhìn thấy các số chỉ có thể nằm ở một vị trí khi xem xét các số khác đã có trong hàng, cột và phần 3x3 ô của nó.
* Mẹo 2: Chia bảng theo trực quan thành 3 cột và 3 hàng. Mỗi cột lớn sẽ có 3 ma trận 3x3 ô và mỗi hàng sẽ có 3 ma trận 3x3 ô. Bây giờ, hãy tìm các cột hoặc ma trận có 2 số giống nhau. Về mặt logic, phải có một số giống nhau thứ 3 nằm trong phần 9 ô còn lại duy nhất đó. Hãy nhìn vào từng vị trí còn lại trong 9 vị trí và xem liệu bạn có thể tìm thấy vị trí của con số còn thiếu hay không.

## 4. Các tiện ích của trò chơi

Trò chơi Sudoku khá đơn giản, nó chỉ phụ thuộc vào tư duy logic nên các nút tiện ích của trò chơi cũng không hề phức tạp.

* Nút Exit: Thoát khỏi trò chơi.
* Nút Delete: Xóa số ở ô vừa chọn.
* Lỗi sai: Nếu người chơi điền sai số vào ô trống thì số lỗi sẽ tang lên, tối đa là 3 lỗi, nếu vượt quá 3 lỗi, trò chơi sẽ thông báo người chơi đã thua và đưa người chơi về ván đấu đó qua nút “OK”.
* Gợi ý: Mỗi ván Sudoku sẽ có 1 gợi ý, người chơi ấn vào ô trắng sau đó ấn “Gợi ý”, đáp án của ô trắng sẽ hiện ra và nếu bấm “Gợi ý” lần thứ 2, trò chơi sẽ thông báo “Bạn đã hết gợi ý” , người chơi ấn “OK” để chơi tiếp ván đấu.
* Bộ đếm thời gian: Trò chơi sẽ cho người chơi trong một khoảng thời gian nhất định để hoàn thành bàn chơi, nếu không hoàn thành trò chơi sẽ thông báo đến người chơi “Bạn đã thua” và quay lại ván chơi đó. Khi hoàn thành trò chơi sẽ thông báo “Bạn đã chiến thắng” và tạo màn chơi mới cho người chơi.
* New game: Nút ấn new game để thực hiện câu lệnh tạo trò chơi mới với các mức “Dễ” (41 ô trắng”), “Trung bình” (46 ô trắng), “Khó” (51 ô trắng), “Chuyên gia” (56 ô trắng).

# II. Thiết kế trò chơi

## 1. Ý tưởng thiết kế:

Tạo ra một ma trận 9x9, sau đó dung 2 file để lưu 2 ma trận (1 ma trân lưu kết quả của 81 ô, 1 ma trận lưu các số và các ô trống thể hiện bằng số 0). Thuật toán sẽ đọc file ma trận sau đó random hàng đầu và dùng backtracking để giải ma trận.

## 2. Thiết kế

* Tạo 1 java project có tên “BTL”, trong project tạo thêm 3 package: “actionview”, “backend”, “view”. Trong package “actionview” có class “listener”, trong package “backend” có class “back211” và “newgame”, trong package “view” có class “Collection”.
* Lớp “listener” lớp lắng nghe các câu lệnh để thực hiện

Hàm “actionPerformed” là hàm hiển thị các nút tiện ích trên thanh công cụ. Khi người chơi nhấn vào “Exit” thì lập tức chương trình thoát ra và in ra sửa sổ Console thông báo “đã nghe” để thông báo thoát thành công trò chơi. Nếu ấn vào “Dễ”, “Trung bình”, “Khó”, “Chuyên gia” thì trò chơi khởi tạo với lần lượt 41, 46, 51, 56, 61 ô trắng; hàm “actionPerformed” cho phép ấn “Gợi ý” để hiện lên ô duy nhất cần được gợi ý.

Hàm “isNumeric” cho phép kiểm tra chuỗi vào ma trận là chữ hay số.

* Lớp “back211”

Hàm “back211” dùng để tạo đối tượng mới của lớp “back211” và khởi tạo một danh sách liên kết để thực hiện thuật toán backtracking. Linklist dùng để lưu trữ lịch sử các lần di chuyển trước đó, sau đó có thể sử dụng để hoàn tác hoặc quay lại các bước trước đó của trò chơi.

Các phương thức “getD”, “setD”, “getC”, “setC” “getB”, “setB” để dùng để truy xuất và thay đổi giá trị của các biến thành viên “d”, “c”, “b”. Chúng cho phép các đối tượng khác có thể đọc và ghi giá trị của các biến thành viên thông qua phương thức getter và setter tương ứng(dùng để hiện text cho các button).

Hàm “exit” có nhiệm vụ thực hiện 1 số biến đổi khi người dùng thao tác. Trong hàm này, đầu vào là 1 chuỗi a được thêm vào danh sách lệnh (command). Sau đó, chuỗi a được chuyển đổi thành một số nguyên và kiểm tra giá trị của nó, nếu giá trị lớn hơn 9 sẽ được gán vào biến d. Nếu số lượng lệnh trong danh sách là 2, nó sẽ lấy 2 phần tử đầu tiên trong danh sách và kiểm tra nó có đúng định dạng hay không. Nếu có, biến b và c sẽ được gán bằng giá trị của các phần tử đó; nếu không, các phần tử sẽ lại được thêm vào danh sách. Nói chung hàm này thực hiện một số kiểm tra lỗi đơn giản trên các giá trị đầu vào của người dùng.

Hàm “rasdd” có tác dụng gán giá trị 100 cho biến b trong đối tượng hiện tại, do đó khi gọi phương thức này, giá trị của biến b sẽ bị thay đổi thành 100.

* Lớp “newgame” để khởi tạo trò chơi mới

Hàm “generateRandomMatrix” dùng để tạo ra một ma trận 9x9 với các phần tử là các số ngẫu nhiên từ 1 đến 9, đảm bảo rằng mỗi số chỉ xuất hiện 1 lần trong mỗi hàng. Nó sử dụng 1 vector để lưu trữ các số ngẫu nhiên được sinh ra và sau đó gán chúng cho hàng đầu tiên của ma trận.

Hàm “newgame” là một constructure để khởi tạo một trò chơi mới với kích thước và độ khó được xác định bởi tham số a. Trong hàm này, ma trận Sudoku được khởi tạo với kích thươc 9x9 bằng cách sử dụng hàm “generateMatrix” để tạo ra một ma trận ngẫu nhiên đáp ứng các yêu cầu của trò chơi Sudoku, sau đó một thread mới được tạo để chạy trò chơi. Ngoài ra, hàm này có thể ném một ngoại lệ IOException nếu có lỗi xảy ra trong quá trình thực hiện.

Hàm run() là phương thức run() của một thread. Nó thực hiện việc giải một game sudoku bằng cách gọi phương thức sodoku() với ma trận sudoku ban đầu và vị trí bắt đầu là (0,0). Nếu xảy ra lỗi IOException hoặc InterruptedException, nó sẽ in ra thông báo lỗi tương ứng.

Hàm “sodoku” được sử dụng để giải một trò chơi Sudoku bằng cách sử dụng đệ quy. Cụ thể, hàm này nhận vào một ma trận 9x9 chứa các giá trị của trò chơi Sudoku và sử dụng thuật toán quay lui để điền các giá trị khuyết sao cho ma trận thỏa mãn các quy tắc của trò chơi này. Nếu hàm tìm được một giải pháp đúng, nó sẽ ghi lại giải pháp này vào một tập tin và kết thúc quá trình giải. Hàm này cũng có chức năng kiểm tra trạng thái interrupt của thread và ném ra ngoại lệ “InterruptedException” để dừng các tác vụ đang chạy 1 cách an toàn.

Hàm “printSolution” có tác dụng in ra một giải pháp (solution) của Sudoku. Game được lưu trong một ma trận 9x9 và lưu giải pháp đó vào một file có đường dẫn filename. Cụ thể, hàm này sẽ duyệt qua từn phần tử của ma trận, in giá trị của mỗi phần tử cùng với một khoảng trắng vào file, và chèn ký tự xuống dòng (“\n”) để xuống dòng mới sau khi in xong 1 hàng của ma trận. Sau khi in xong toàn bộ ma trận, hàm sẽ đóng file.

Hàm “generateSudoku” có tác dụng tạo ra một bảng Sudoku và khoảng trắng bằng cách xóa ngẫu nhiên một số phần tử của bảng cho đến khi lỗ hổng bằng tham số a, sau đó ghi bảng Sudoku với lỗ hổng vào một file được chỉ đinh bởi tham số filename.

Hàm “check” được sử dụng trong thuật toán giải Sudoku để kiểm tra xem một số nguyên k có hợp lệ tại vị trí (x,y) trong ma trận c hay không. Để kiểm tra tính hợp lệ, hàm sẽ kiểm tra xem số k đã xuất hiện trong hàng và cột chứa (x,y) chưa. Sau đó, hàm sẽ kiểm tra xem số k đã xuất hiện trong vùng 3x3 chứa ô (x,y) hay chưa. Nếu k đã xuất hiện trong bất kỳ hàng, cột hoặc vùng 3x3 nào, hàm sẽ trả về “false” để thể hiện rằng k không hợp lệ tại vị trí (x,y) của ma trận Sudoku. Ngược lại, hàm sẽ trả về “true” để thể hiện rằng k hợp lệ tại vị trí (x,y) của ma trận Sudoku.

* Lớp “Collection”

Hàm public Collection() throws IOException là constructor (hàm khởi tạo) của lớp Collection. Trong hàm này, đối tượng JFrame được tạo ra với tên "Nine By Nine Board" và các biến instance của lớp được khởi tạo. Cụ thể, trong hàm này: this.matrix1 và this.matrix được khởi tạo là các ma trận số nguyên kích thước 9x9; this.bk được khởi tạo là một đối tượng của lớp back211; một ActionListener được tạo và gán cho biến ac để lắng nghe các sự kiện trên giao diện; hàm this.view() được gọi để hiển thị giao diện chính của ứng dụng; các biến instance khác như error, dem, và goiy cũng được khởi tạo.

Hàm view() khởi tạo giao diện người dùng đồ họa trò chơi. Hàm đặt kích thước cửa sổ thành 600x400 và tạo một mảng JButton để giữ 81 nút trong trò chơi. Một JMenuBar được tạo để chứa các mục menu của trò chơi bao gồm: “Thoát”, “Xóa”, “Lỗi”, “Newgame” và “Thời gian”. Chức năng thiết lập ActionListeners cho mỗi mục menu để các mục menu thực hiện các hành động chính xác khi được chọn. Chức năng thiết lập đồng hồ bấm giờ để theo dõi thời gian người chơi dành để giải trò chơi. Bộ hẹn giờ cập nhật thời gian trong JMenuBar mỗi giây và nếu thời gian về 0, người chơi sẽ thua trò chơi. Hàm đọc hai tệp, test\_file1.txt và test\_file.txt, để nhận các câu đố Sudoku ban đầu và đã giải tương ứng. Chức năng tạo một JPanel để giữ 81 nút và thêm chúng vào bảng điều khiển. Hàm đặt màu nền của các nút thành màu trắng và nếu nút tương ứng với một ô được điền sẵn trong câu đố Sudoku, thì hàm sẽ đặt văn bản của nút thành số tương ứng. Cuối cùng, hàm thêm JPanel và JMenuBar vào cửa sổ bằng cách sử dụng BorderLayout.

Hàm name() nhận đầu vào là một chuỗi a và một số nguyên b, và ném ra ngoại lệ IOException. Trong hàm, giá trị của chuỗi a được chuyển đổi thành một số nguyên bằng cách sử dụng phương thức Integer.parseInt(). Nếu giá trị của b bằng 0, hàm sẽ gọi phương thức exit() của một đối tượng bk. Nếu getC() của bk khác 0 và getB() của bk khác 0 và khác 100, thì hàm sẽ thực hiện một số thao tác giao diện người dùng. Cụ thể, nút được xác định bởi getC() được đặt văn bản trở thành giá trị của getB() và được đặt màu nền xám. Sau đó, vị trí của nút được tính bằng cách sử dụng phép toán số học và các thao tác được thực hiện trên ma trận matrix1. Nếu giá trị của a là một số nhỏ hơn 10, biến dem được tăng lên. Nếu tổng của dem và dem1 là 81 (kích thước của bảng), một hộp thoại thông báo sẽ hiển thị để thông báo người dùng đã chiến thắng. Sau đó, một cửa sổ mới sẽ hiển thị để chơi ván khác, bằng cách tạo một đối tượng newGameWindow, tắt cửa sổ hiện tại, và mở cửa sổ mới. Nếu giá trị của matrix1[m][n] khác với getB() của bk và giá trị của a nhỏ hơn 10, biến error được tăng lên và một thông báo lỗi sẽ được hiển thị trên màn hình. Nếu error bằng 3, bộ hẹn giờ timer sẽ bị hủy và một hộp thoại sẽ hiển thị để thông báo cho người dùng đã thua. Sau đó, một cửa sổ mới sẽ được hiển thị để chơi lại ván, bằng cách tạo một đối tượng play\_back, tắt cửa sổ hiện tại và mở cửa sổ mới.

Hàm new\_game() được sử dụng dể bắt đầu một trò chơi mới với độ khó được đưa vào thông qua tham số b. Cụ thể, hàm này tao ra một cửa sổ mới cho trò chơi với độ khó tương ứng, sau đó tắt cửa sổ trò chơi hiện tại và hiển thị cửa sổ mới. nó giúp người dùng chơi lại trò chơi mới với mức độ khó khác nhau mà không cần thoát khỏi ứng dụng và khởi động lại từ đầu.

Hàm delete1 được sử dụng để xóa số hiển thị trên một ô cờ bàn và thiết lập lại màu nền của ô đó về trắng. Cụ thể, hàm thực hiện các công việc sau: Gọi phương thức rasdd của đối tượng bk để xóa giá trị của ô cờ bàn tại vị trí hiện tại; thiết lập giá trị của ô đó thành rỗng và màu nền của ô đó về trắng bằng cách sử dụng các phương thức setText và setBackground trên đối tượng jButtons với chỉ số tương ứng với vị trí của ô cờ bàn trên giao diện người dùng.

Hàm play\_back() dùng để thực hiện chức năng "chơi lại" (play back) bước đi trước đó của người chơi. Nó cho phép người chơi quay lại bước đi trước đó bằng cách hiển thị lại giá trị của ô đó và đặt lại màu nền của ô đó. Nếu người chơi sử dụng hết số lần gợi ý thì khi bấm "chơi lại" sẽ hiển thị một thông báo cho biết đã hết số lần gợi ý.

Hàm main là hàm để chạy chương trình.

III. Hướng dẫn cài đặt:

Tạo 1 java project có tên “BTL”, trong project tạo thêm 3 package: “actionview”, “backend”, “view”. Trong package “actionview” có class “listener”, trong package “backend” có class “back211” và “newgame”, trong package “view” có class “Collection” rồi sau đó vào class “Collection” để chạy chương trình.

**Tài liệu tham khảo**

1. Slide bài giảng của Giảng viên bộ môn.

2. https://sudoku.com/vi.