

**TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
BỘ MÔN CÔNG NGHỆ PHẦN MỀM**



**NIÊN LUẬN CƠ SỞ NGÀNH
KỸ THUẬT PHẦN MỀM**

**ĐỀ TÀI
GAME SUDOKU**

Giáo viên hướng dẫn

Ths. Phan Huy Cường
MSCB: 001586

Sinh viên thực hiện

Trần Tuấn Đạt
MSSV: B1805749
Lớp: DI1896A3

Học kỳ 1, năm học 2021-2022

LỜI CẢM ƠN

Trước hết tôi xin gửi tới quý thầy cô khoa Công Nghệ Thông Tin và Truyền Thông, trường Đại Học Cần Thơ lời chúc sức khỏe và lời biết ơn sâu sắc. Được sự quan tâm, dạy dỗ và chỉ bảo chân tình, chu đáo của quý thầy cô, đã giúp chúng tôi có được những kiến thức vô cùng quý giá, giúp tôi hiểu được giá trị của việc học và vận dụng kiến thức vào áp dụng trong thực tiễn. Thổi lên niềm đam mê khám phá và nghiên cứu khoa học trong tôi.

Để báo cáo này đạt kết quả tốt đẹp, tôi đã nhận được sự góp ý và hỗ trợ chân thành từ những anh chị, bạn bè cùng khoa. Với những tình cảm tốt đẹp, cho phép tôi được bày tỏ lòng biết chân thành đến tất cả các anh chị, bạn bè đã tạo điều kiện giúp đỡ tôi trong quá trình học tập cũng như trong quá trình nghiên cứu đề tài.

Đặc biệt tôi muốn gửi lời cảm ơn chân thành nhất tới giảng viên Phan Huy Cường đã tạo điều kiện tốt nhất để tôi có thể hoàn thành tốt đề tài này trong thời gian qua.

Với điều kiện hạn chế và những ảnh hưởng phân nào của dịch Covid-19 trong thời gian qua, cũng như kinh nghiệm còn hạn chế của sinh viên, báo cáo này không thể tránh được những thiếu sót. Tôi rất mong nhận được sự chỉ bảo, đóng góp ý kiến của quý thầy cô để tôi có điều kiện bổ sung, nâng cao kiến thức của mình, phục vụ tốt hơn trong học tập cũng như công tác sau này.

Sinh viên thực hiện,

Trần Tuấn Đạt

ĐÁNH GIÁ KẾT QUẢ THỰC HIỆN NIÊN LUẬN CƠ SỞ NGÀNH KTPM

(Học kỳ: 01, năm học 2021 - 2022)

TÊN ĐỀ TÀI: Game Sudoku**GIÁO VIÊN HƯỚNG DẪN:**

STT	HỌ VÀ TÊN	MSCB
1	Phan Huy Cường	001586

SINH VIÊN THỰC HIỆN:

HỌ VÀ TÊN	MSSV	THƯỜNG (TỐI ĐA 1,0 ĐIỂM)	ĐIỂM (THANG ĐIỂM 10)
Trần Tuấn Đạt	B1805749		

I. HÌNH THỨC (0,5 điểm)	
Bìa (tối đa 0,25 điểm)	
❖ Đầy đủ các thông tin ❖ Đúng định dạng	
Bố cục (0,25 điểm)	
❖ Trang đánh giá kết quả thực hiện niên luận 1 ❖ Mục lục: cấu trúc chương, mục và tiểu mục ❖ Phụ lục (nếu có) ❖ Tài liệu tham khảo	
II. NỘI DUNG (3,5 điểm)	
Giới thiệu (tối đa 0,5 điểm)	
❖ Mô tả bài toán (0,25 điểm) ❖ Mục tiêu cần đạt, hướng giải quyết (0,25 điểm)	
Lý thuyết (tối đa 0,5 điểm)	
❖ Các khái niệm sử dụng trong chương trình (0,25 điểm) ❖ Kết quả vận dụng lý thuyết trong đề tài (0,25 điểm)	
Ứng dụng (2,0 điểm)	
❖ Phân tích yêu cầu, xây dựng các cấu trúc dữ liệu (0,5 điểm) ❖ Sơ đồ chức năng, lưu đồ giải thuật giải quyết vấn đề (1,0 điểm) ❖ Giới thiệu sử dụng chương trình (0,5 điểm)	
Kết luận (tối đa 0,5 điểm)	
❖ Nhận xét kết quả đạt được ❖ Hạn chế ❖ Hướng phát triển	
III. CHƯƠNG TRÌNH DEMO (5,0 điểm)	
Giao diện thân thiện với người dùng (1,0 điểm)	
Hướng dẫn sử dụng (0,5 điểm)	
Kết quả thực hiện đúng với kết quả của phần ứng dụng (tối đa 3,5 điểm)	
❖ Kết quả đúng (2,0 điểm) ❖ Cách thực hiện hợp lý (1,0 điểm) ❖ Chức năng bổ sung, sáng tạo (0,5 điểm)	

Cần Thơ, ngày.....tháng.....năm 2021.

GIÁO VIÊN CHẤM

Phan Huy Cường

MỤC LỤC

1. CHƯƠNG I: PHẦN GIỚI THIỆU	1
1.1. Đặt vấn đề.	1
1.2. Lịch sử giải quyết vấn đề.	1
1.3. Mục tiêu đề tài.....	2
1.4. Đối tượng, phạm vi nghiên cứu.	2
1.5. Phương pháp nghiên cứu.....	2
2. CHƯƠNG II: CƠ SỞ LÝ THUYẾT.....	3
2.1. Giải thuật quay lui.	3
2.2. Độ quy.	4
2.3. Mô tả bài toán.....	4
3. CHƯƠNG III: PHÂN TÍCH BÀI TOÁN.....	5
3.1. Xây dựng dữ liệu tổ chức ma trận.	5
3.2. Xây dựng thuật toán quay lui kết hợp tìm kiếm sâu.	5
4. CHƯƠNG IV: THIẾT KẾ - CÀI ĐẶT	7
4.1. Thiết kế (kèm hướng dẫn cách chơi game).	7
4.2. Cài đặt.	11
4.2.1. Khởi tạo game.....	11
4.2.2. Giải thuật tìm vị trí ô trống trong ma trận.	13
4.2.3. Giải thuật kiểm tra đúng/sai khi người chơi nhập vào một con số.....	13
4.2.4. Giải thuật quay lui-vết cạn để giải ma trận.....	14
5. CHƯƠNG V: TỔNG KẾT	16
PHỤ LỤC	17
TÀI LIỆU THAM KHẢO.....	18

TÓM TẮT

Niên luận cơ sở ngành là một môn học quan trọng trong đối với sinh viên ngành Kỹ thuật phần mềm. Với mục đích ứng dụng các kiến thức về giải thuật để giải quyết những vấn đề trong cuộc sống. Trong học phần này, tôi được giao đề tài theo số thứ tự là “**Giải trò chơi Sudoku**”. Mục tiêu của đề tài là dựa vào gợi ý là các số cho sẵn trong trò chơi Sudoku, ta tiến hành điền giá trị từ 1 đến 9 vào các ô trống còn lại, sao cho ở mỗi dòng, mỗi cột, mỗi vùng 3 x 3 không có số nào bị lặp lại.

1. CHƯƠNG I: PHẦN GIỚI THIỆU

1.1. Đặt vấn đề.

Ngày nay tin học ngày càng gắn liền với cuộc sống và công việc của con người. Đặc biệt trong lĩnh vực Công nghệ thông tin, đã thật sự bùng nổ về công nghệ và hỗ trợ ngày càng tối ưu cho các công việc của con người trong nhiều lĩnh vực giúp con người giải đáp các vấn đề trong cuộc sống thực tế. Ứng dụng rất nhiều như nhận dạng vân tay khuôn mặt, chữ viết, tiếng nói, và cả chẩn đoán bệnh trong y học.

Nhằm áp dụng những kiến thức đã được học, tôi chọn đề tài Bài toán Sudoku – với hy vọng đề tài này sẽ là một minh chứng khẳng định Công nghệ thông tin là một phần rất quan trọng và đồng thời nó cũng có rất nhiều ứng dụng thiết thực cho mọi lĩnh vực trong thế giới hiện tại của chúng ta.

1.2. Lịch sử giải quyết vấn đề.

Sudoku từng đi qua các nền văn hóa cổ, và có lẽ nó bắt nguồn từ Trung Hoa, theo như một tài liệu của Ả Rập vào thế kỷ thứ 9. Năm 990, một danh sách những “Ô số kỳ ảo” đã xuất hiện và tỏ ra không khác mấy so với bản Sudoku xuất hiện trong Từ Điển Bách Khoa Ikhwān al-salfā của các học giả người Ả Rập. Trong từ điển này, họ gọi nó là wafa.

Abraham Ben ibn Ezra - một nhà triết học kiêm chiêm tinh học người Hispanic (Tây Ban Nha - Bồ Đào Nha) gốc Do thái - bắt đầu quảng bá khối vuông buduh ở châu Âu. Ông đi khắp Tây Ban Nha, Ý và các nước khác ở châu Âu để giới thiệu với công chúng về “những ô số kỳ ảo”.

Sudoku có thêm một bước tiến hóa mới vào năm 1776 khi một nhà toán học kiêm vật lý học người Thụy Sĩ tên Leonhard Euler bắt đầu nghiên cứu và phát triển các luật chơi mà ngày nay ta gọi là luật chơi Sudoku.

Đi xuyên qua một cuộc hành trình dài lâu và kiên trì, Sudoku lần đầu tiên được xuất bản vào cuối thập niên 1970 trong một tờ tạp chí ở New York. Tờ tạp chí này đã giới thiệu về các ô số kỳ ảo và khuôn nó lại trong một lưới 9x9, tạo thành từ các khối 3x3. Và như thế, Sudoku đã ra đời.

Những “con nghiện Sudoku” chỉ bó hẹp trong khuôn khổ xứ sở anh đào trong suốt hơn 20 năm cho đến khi một thẩm phán người Hồng Kông gốc New Zealand tên là Wayne Gould tình cờ phát hiện một cuốn sudoku trong một hiệu sách Nhật Bản. Ông đam mê nghiên cứu trò chơi số cổ xưa này.

Năm 2004, niềm đam mê Sudoku đã đưa Wayne Gould đến với London (Anh). Nhân một chuyến thăm ngẫu nhiên báo The Times, Gould đã thuyết phục tổng biên tập của báo này cho đăng Sudoku bên cạnh các ô chữ. Độc giả lập tức bị cuốn hút và yêu cầu đăng thêm nữa. Chỉ trong vài tuần lễ, các tờ báo trên khắp nước Anh đã thi nhau đăng Sudoku. Từ đó, Sudoku bắt đầu lan rộng sang Mỹ, Canada, Úc, Pháp, Nam Phi và nhiều quốc gia khác.

Sudoku là trò puzzle (đoán số hay chữ) phát triển nhanh nhất trên thế giới. Nó hiện có hàng triệu tín đồ và con nghiện. Nhiều nhân vật nổi tiếng ủng hộ nó. Và nó đã có được một nhà vô địch thế giới. Chính quyền nhiều nước đã khuyến cáo Sudoku như một công cụ rèn luyện trí lực và hạn chế sự phát triển của bệnh Alzheimer.

1.3. Mục tiêu đề tài.

Ứng dụng bài toán thỏa mãn ràng buộc để xây dựng được trò chơi SUDOKU trên bảng vuông có 9 dòng 9 cột.

1.4. Đối tượng, phạm vi nghiên cứu.

Đối tượng nghiên cứu:

- Thuật toán (quay lui - vét cạn) thỏa mãn ràng buộc.
- Trò chơi Sudoku.
- Ngôn ngữ lập trình Java.

Phạm vi nghiên cứu: đề tài xây dựng 1 trò chơi minh họa cho bài toán Sudoku với các kiến thức được cung cấp và các hàm được hỗ trợ trong ngôn ngữ Java.

1.5. Phương pháp nghiên cứu.

Tìm hiểu các thuật toán quay lui, đệ quy, tìm kiếm sâu để giải quyết bài toán thỏa mãn ràng buộc đã học trên lớp để áp dụng vào bài toán. Bằng cách xây dựng những hàm chức năng để xử lý quá trình tìm kiếm lời giải trong phạm vi đề tài.

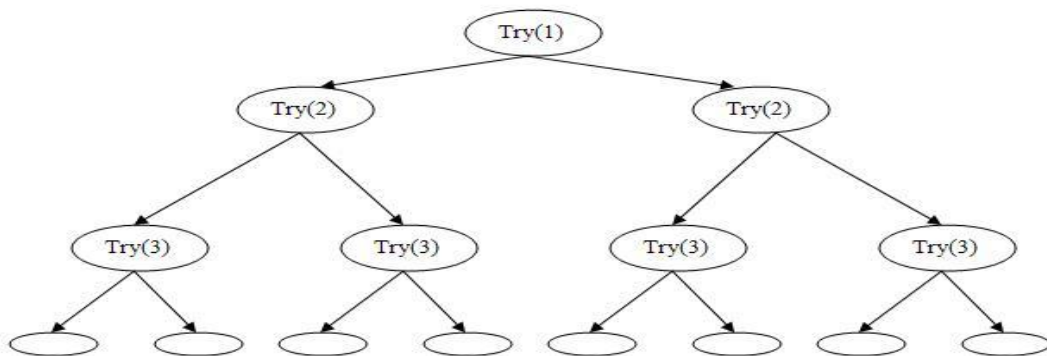
2. CHƯƠNG II: CƠ SỞ LÝ THUYẾT

2.1. Giải thuật quay lui.

Suy diễn lùi là một chiến lược tìm kiếm lời giải cho các bài toán thỏa mãn ràng buộc. Các bài toán thỏa mãn ràng buộc là các bài toán có một lời giải đầy đủ, trong đó thứ tự của các phần tử không quan trọng. Các bài toán này bao gồm một tập các biến mà mỗi biến cần được gán một giá trị tùy theo các ràng buộc cụ thể của bài toán. Việc quay lui là để thử tất cả các tổ hợp để tìm được một lời giải. Thế mạnh của phương pháp này là nhiều cài đặt tránh được việc phải thử nhiều tổ hợp chưa hoàn chỉnh, và nhờ đó giảm thời gian chạy, tìm được nhiều đáp án cho những bài có nhiều cách giải.

Đó là một quá trình tìm kiếm theo độ sâu trong một tập hợp các lời giải. Trong quá trình tìm kiếm, nếu ta gặp một hướng lựa chọn không thỏa mãn, ta quay lui về điểm lựa chọn nơi có các hướng khác và thử hướng lựa chọn tiếp theo. Khi đã thử hết các lựa chọn xuất phát từ điểm lựa chọn đó, ta quay lại điểm lựa chọn trước đó và thử hướng lựa chọn tiếp theo tại đó. Quá trình tìm kiếm thất bại khi không còn điểm lựa chọn nào nữa.

Chiến lược quay lui tương tự với tìm kiếm theo độ sâu nhưng sử dụng ít không gian bộ nhớ hơn, nó chỉ lưu giữ trạng thái của một lời giải hiện tại và cập nhật nó.



Hình 1: Mô tả giải thuật quay lui.

- Ở một bài toán hiện tại (mỗi nút), ta đi tìm lời giải cho bài toán đó. Ứng với lời giải, ta đi giải bài toán kế tiếp cho đến khi lúc bài toán gốc trở nên đầy đủ.
- Lời giải của bài toán gốc thường là một lối đi từ gốc đến nút cuối cùng (không có nút con).
- Người ta thường sử dụng một số phương pháp heuristic để tăng tốc cho quá trình quay lui.

2.2. Độ quy.

Độ quy (Recursion) là một trong những giải thuật khá quen thuộc trong lập trình, mở rộng ra là trong toán học (thường được gọi với tên khác là “quy nạp”). Một hàm được gọi là đệ quy nếu trong bản thân hàm đó có chứa lệnh gọi lại chính nó. Có một số bài toán, buộc phải sử dụng đệ quy mới giải quyết được, chẳng hạn như trong trò chơi sudoku.

2.3. Mô tả bài toán.

Trò chơi Sudoku thường có các dạng 4x4, 6x6, 9x9, 12x12... trong vùng chơi sẽ có ma trận với các số cho trước.

Với phiên bản Sudoku 9x9 người chơi có nhiệm vụ điền các con số từ 1 đến 9 vào những ô trống sao cho:

- Mỗi cột dọc
- Mỗi hàng ngang
- Mỗi phân vùng nhỏ (ô 3x3)

Điều có đủ các số từ 1 đến 9 mà không được lặp lại với nhau (đối với dạng 9x9).

Trò chơi kết thúc khi các ô trống đã được điền và thỏa mãn điều kiện mà trò chơi đặt ra.

			2		9	6		
		9			3			
6	3							
	4					2		
					2			
			6	4	8	5		
		2	9		6			
			8					

1	5	4	2	7	9	6	3	8
2	8	9	1	6	3	4	5	7
6	3	7	5	8	4	9	1	2
3	4	6	7	9	5	2	8	1
5	9	8	3	1	2	7	4	6
7	2	1	6	4	8	5	9	3
4	1	2	9	3	6	8	7	5
9	6	3	8	5	7	1	2	4
8	7	5	4	2	1	3	6	9

Hình 2: Trạng thái ban đầu (trái) và trạng thái goal (phải).

3. CHƯƠNG III: PHÂN TÍCH BÀI TOÁN

3.1. Xây dựng dữ liệu tổ chức ma trận.

Theo luật chơi, mỗi ô vuông sẽ có các giá trị từ 1 đến 9. Vì thế, ta có thể sử dụng mảng 2 chiều để lưu giá trị từ 1 đến 9, đồng thời tọa độ mảng cũng chính là tọa độ của ô mà ta cần điền số.

Chỉ số nằm ngang là vị trí số cột của ma trận, và chỉ số dọc chính là vị trí số dòng của ma trận trong trò chơi.

	0	1	2	3	4	5	6	7	8
0	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
1	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
2	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
3	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
4	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
5	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
6	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
7	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9
8	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9	1..9

Hình 3: Bảng tọa độ, giá trị.

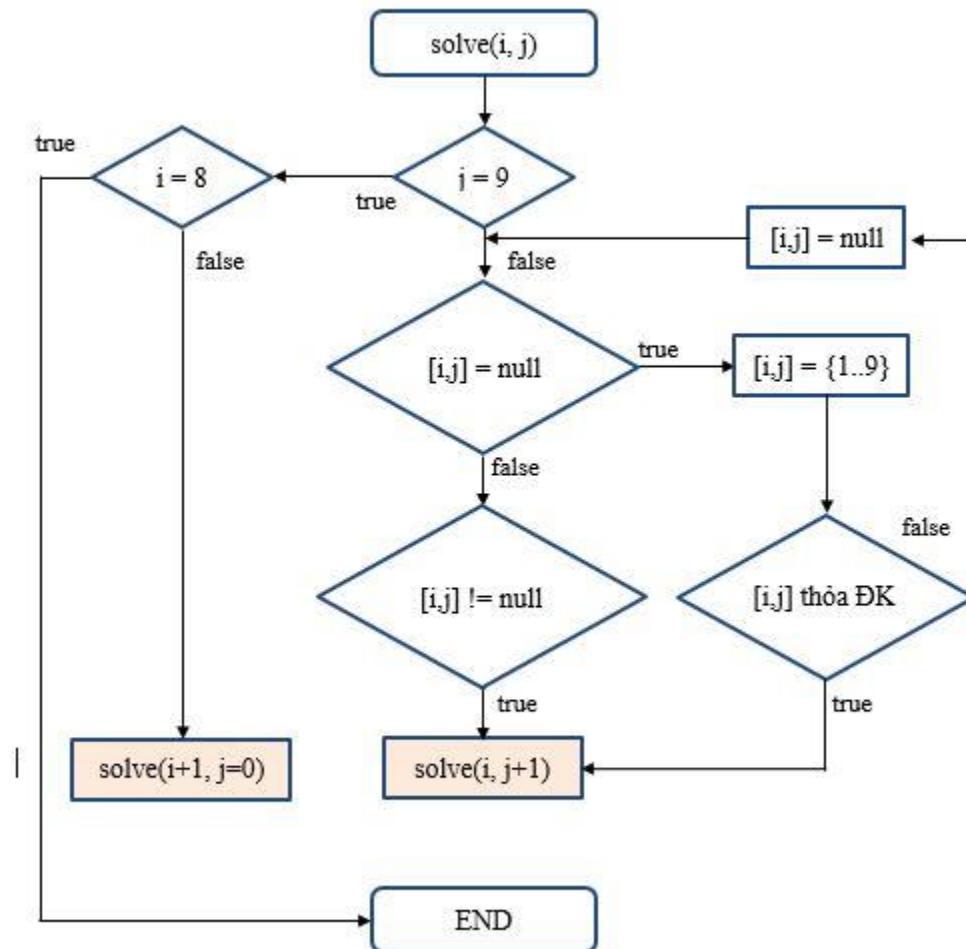
3.2. Xây dựng thuật toán quay lui kết hợp tìm kiếm sâu.

Việc tìm kiếm giá trị cho vị trí trống được thiết kế bằng giải thuật đệ quy và tìm kiếm sâu kết hợp với quay lui, bằng cách đặt thử từng giá trị vào ô trống.

Chúng ta cùng xét việc thử đặt giá trị ở cột thứ i và dòng thứ j :

- Điều kiện dừng đệ quy: Nếu dòng $j = 9$, cột $i = 8$ thì dừng đệ quy, nếu như cột $i < 8$ thì gán cột $i = i + 1$, và biến dòng $j = 0$.
- Nếu ô đang xét tại vị trí hàng j cột i mà trống thì ta bắt đầu điền giá trị với các số từ 1 - 9. Và sau đó kiểm tra sau mỗi lần thử từng số, nếu số đó thỏa điều kiện thì tiếp tục đệ quy ở vị trí tiếp theo là cột i và dòng $j + 1$. Ngược lại nếu các số từ 1 đến 9 mà không điền được thì xóa giá trị hiện tại và quay lui về vị trí dòng trước đó tăng giá trị lên 1 đơn vị và tiếp tục đệ quy.

- Nếu ô đang xét tại vị trí hàng j cột i mà đã có giá trị thì ta bỏ qua, và đệ quy tiếp tục ở vị trí cột i và dòng $j + 1$.



Hình 4: Lưu đồ giải thuật quay lui.

4. CHƯƠNG IV: THIẾT KẾ - CÀI ĐẶT

4.1. Thiết kế (kèm hướng dẫn cách chơi game)

Giao diện chương trình gồm có 81 ô (9x9), và 9 nút lệnh (Hình 5.1):

- **Nút “VÁN MỚI”**: Khi click vào nút này, chương trình sẽ hiển thị ma trận – ván cờ mới với level đang chọn tương ứng thay thế cho ma trận cũ (Hình 5.2).

- **Nút “CHƠI LẠI”**: Khi click vào nút này, tất cả các giá trị do người dùng đã nhập vào ở các ô trống sẽ bị xóa hết, và quay lại ma trận hiện tại.

- **Nút “KIỂM TRA”**: Khi click vào nút này, màu nền của các ô cờ tại vị trí người dùng đã nhập vào sẽ có sự thay đổi: màu xanh lá – con số người dùng nhập vào tại ô đó đúng, màu đỏ - con số người dùng nhập vào sai. Nếu người dùng đã giải đúng, chương trình sẽ hiện thông báo người dùng đã thắng, người dùng có thể lưu lại ma trận Sudoku này ra file .txt (Hình 5.3).

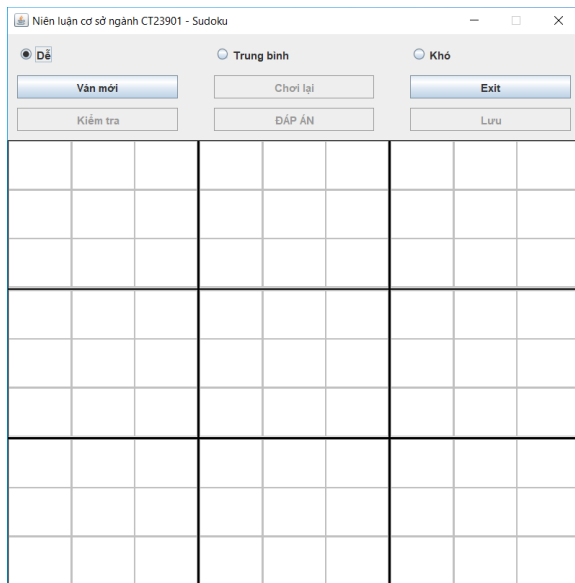
- **Nút “ĐÁP ÁN”**: Khi click vào nút này, đáp án cho ma trận Sudoku sẽ hiện ra, các ô trống sẽ được điền vào, nếu ô người dùng đã nhập sai sẽ bị thay thế bằng kết quả đúng và người dùng có thể click chọn Lưu để lưu ma trận này (Hình 5.4).

- **Nút “LƯU”**: Khi click vào nút này, chương trình sẽ lưu lại ma trận gồm ma trận chưa giải và đáp án, thời gian hiện tại và level người dùng đang chọn ra tập tin .txt vào thư mục \Log. Hệ thống hiển thị cửa sổ thông báo lưu thành công/thất bại (Hình 5.5). Người dùng có thể xem lại ma trận này trong thư mục \Log (Hình 5.7 5.8).

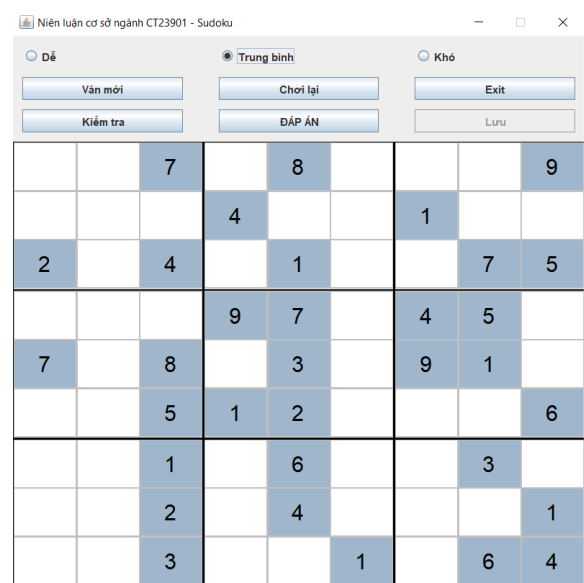
- **Nút “THOÁT”**: Khi click vào nút này, chương trình sẽ hiện thông báo lựa chọn YES/NO người dùng có chắc chắn muốn thoát (Hình 5.6).

- **Nút “Dễ”, “Trung bình”, “Khó”**: Khi click vào nút này, ma trận mới sẽ hiện ra tương ứng với level của nút mà người dùng chọn. Độ khó của các ma trận Sudoku này sẽ tăng dần thông qua việc giảm các ô số đề cho ban đầu và tăng số ô trống. Người dùng sẽ mất thời gian hơn, tư duy nhiều hơn để có thể hoàn thành trò chơi ở level cao hơn. Việc tạo ra ma trận sẽ được random một cách ngẫu nhiên các con số và vị trí xuất hiện của chúng. Do đó, trong cùng một mức level, độ khó giữa các ván game cũng có sự chênh lệch khác nhau.

- Khi khởi chạy chương trình, bảng 9x9 chưa có ma trận, các nút Chơi lại, Kiểm tra, Đáp án, Lưu sẽ bị khóa. Người dùng có thể click chọn các radiobutton level ở trên hoặc chọn nút Ván mới để có thể bắt đầu chơi. Trong quá trình giải, người dùng có thể chọn Kiểm tra để xem các giá trị đã nhập ở các ô trước là đúng hay chưa, hoặc sau khi nhập xong cả bảng Sudoku. Nếu người dùng giải đúng toàn bộ, chương trình sẽ hiện thông báo và tùy chọn người dùng có thể chơi ván mới hoặc lưu ma trận này ra file text.



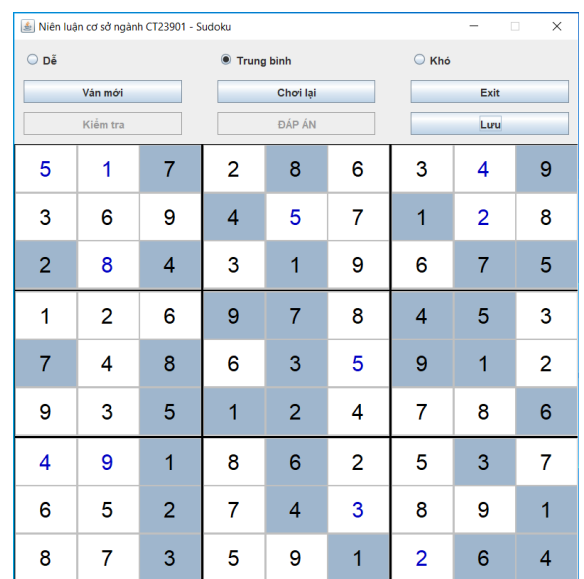
Hình 5.1



Hình 5.2



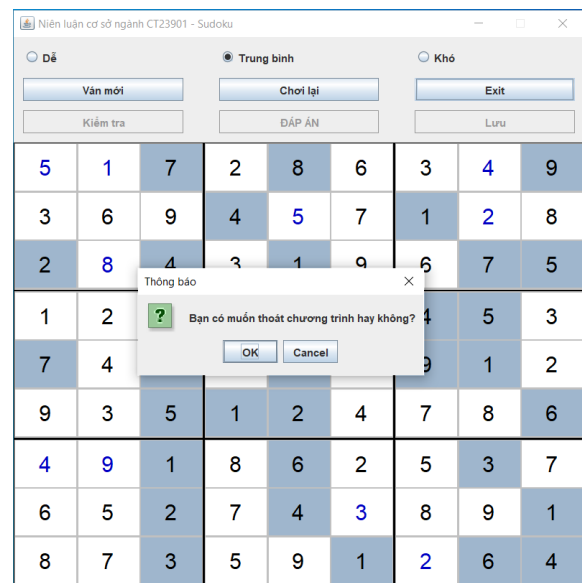
Hình 5.3



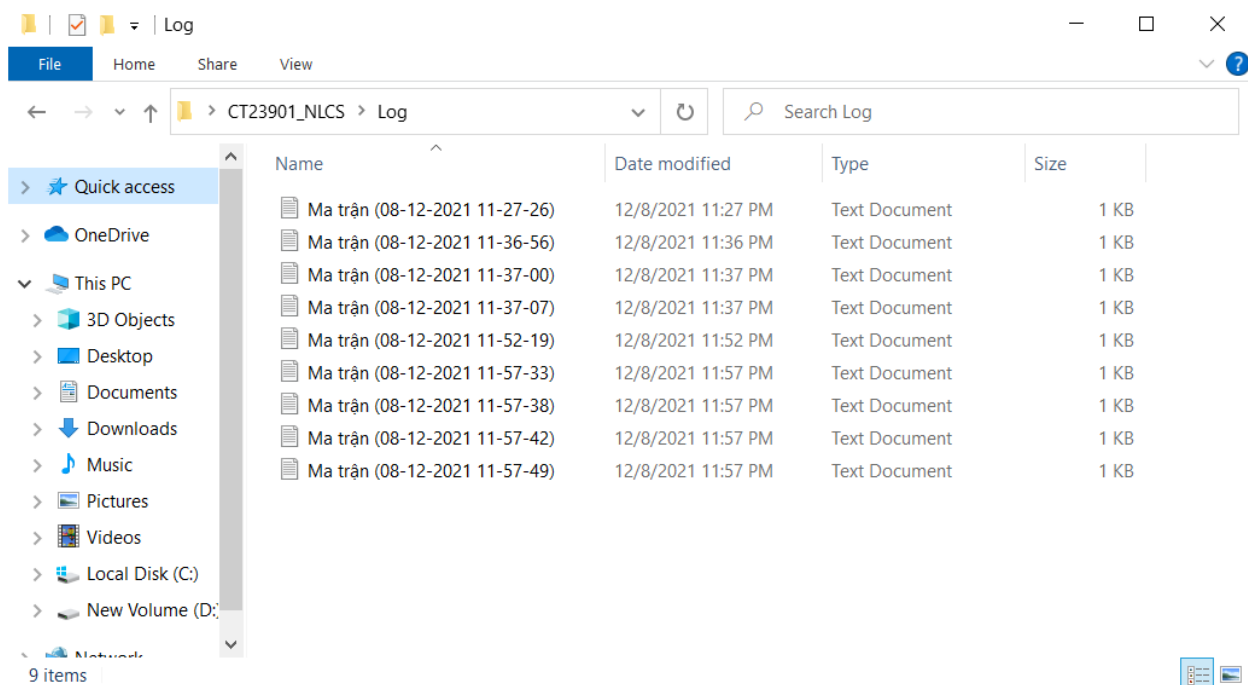
Hình 5.4



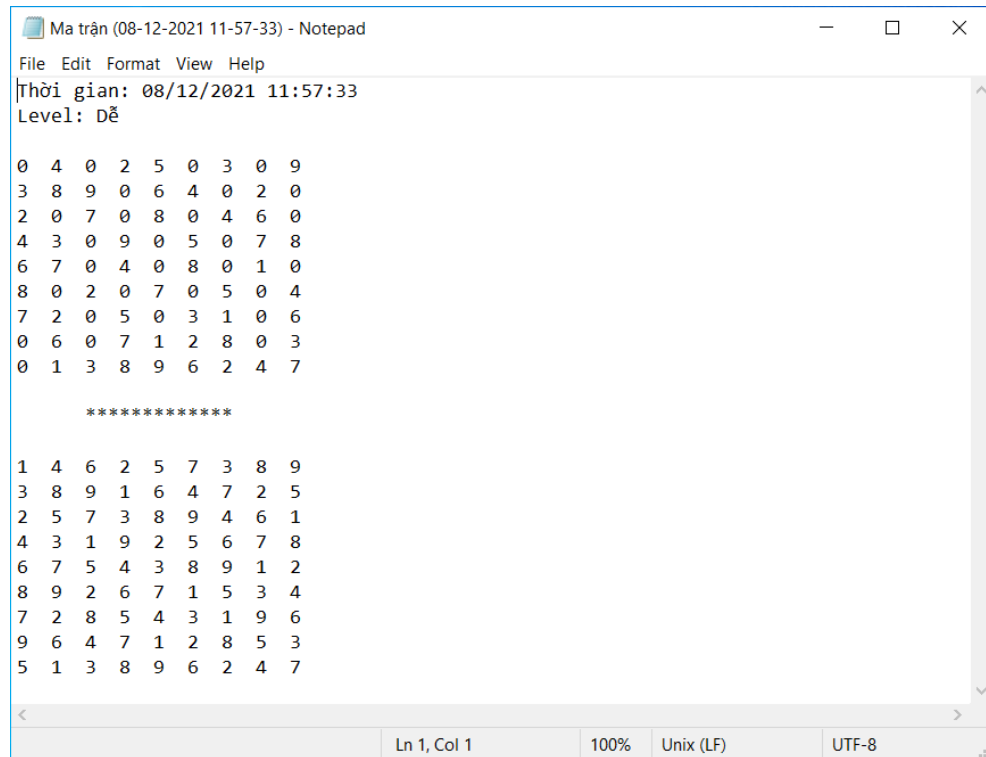
Hình 5.5



Hình 5.6



Hình 5.7



The screenshot shows a Notepad window titled "Ma trận (08-12-2021 11-57-33) - Notepad". The menu bar includes File, Edit, Format, View, and Help. The text content is as follows:

```

Thời gian: 08/12/2021 11:57:33
Level: Dễ

0 4 0 2 5 0 3 0 9
3 8 9 0 6 4 0 2 0
2 0 7 0 8 0 4 6 0
4 3 0 9 0 5 0 7 8
6 7 0 4 0 8 0 1 0
8 0 2 0 7 0 5 0 4
7 2 0 5 0 3 1 0 6
0 6 0 7 1 2 8 0 3
0 1 3 8 9 6 2 4 7

*****

1 4 6 2 5 7 3 8 9
3 8 9 1 6 4 7 2 5
2 5 7 3 8 9 4 6 1
4 3 1 9 2 5 6 7 8
6 7 5 4 3 8 9 1 2
8 9 2 6 7 1 5 3 4
7 2 8 5 4 3 1 9 6
9 6 4 7 1 2 8 5 3
5 1 3 8 9 6 2 4 7
    
```

The status bar at the bottom indicates "Ln 1, Col 1", "100%", "Unix (LF)", and "UTF-8".

Hình 5.8

4.2. Cài đặt

Sau đây là các hàm, các giải thuật cốt lõi để hình thành nên Game Sudoku:

4.2.1. Khởi tạo game

- Giải thuật khởi tạo ma trận (81 ô số).

```
public Board() {
    initComponents();
    //Cài đặt giao diện của panel chính
    panMain = new JPanel();
    panMain.setLayout(new GridLayout(3, 3));
    panMain.setBackground(Color.BLACK);
    setLayout(new BorderLayout());
    add(panMain);
    o = new JTextField[9][9];
    paneles = new JPanel[3][3];

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            paneles[i][j] = new JPanel();
            paneles[i][j].setBorder(BorderFactory.createLineBorder(Color.black));
            paneles[i][j].setLayout(new GridLayout(3, 3));
            panMain.add(paneles[i][j]);
        }
    }
    //Thêm ô textfield vào các khối 3x3
    for (int n = 0; n < 9; n++) {
        for (int i = 0; i < 9; i++) {
            o[n][i] = newtextfield();
            int fm = (n + 1) / 3;
            if ((n + 1) % 3 > 0)
                fm++;
            int cm = (i + 1) / 3;
            if ((i + 1) % 3 > 0)
                cm++;
            paneles[fm - 1][cm - 1].add(o[n][i]);
        }
    }
}
```


- Giải thuật tạo ma trận mới:

```
//Hàm tạo ma trận mới cho ván game mới
public static void newGame() {
    int k = 0;
    ArrayList<Integer> randomnumber = getRandomNum();

    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            grid[i][j] = 0;
            if ((j + 2) % 2 == 0 && (i + 2) % 2 == 0) {
                grid[i][j] = randomnumber.get(k);
                k++;
                if (k == 9) {
                    k = 0;
                }
            }
        }
    }
    search(grid);
    int rann = ran.nextInt(level);
    int c = 0;
    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            temp[i][j] = 0;
            if (c < rann) {
                c++;
                continue;
            } else {
                rann = ran.nextInt(level);
                c = 0;
                temp[i][j] = grid[i][j];
            }
        }
    }
    b.setarray(grid, temp);
    b.setTextLable();
}
```

4.2.2. Giải thuật tìm vị trí ô trống trong ma trận.

Giải thuật bên dưới sẽ tìm và trả về danh sách các ô còn trống mà nó tìm được. Danh sách này sẽ được dùng để chạy giải thuật giải ma trận Sudoku.

```
//Hàm lấy về danh sách các ô trống
public static int[][] getList_otrong(int[][] grid) {
    int slotrong = 0; //Biên đếm số ô trống
    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            if (grid[i][j] == 0) {
                slotrong++;
            }
        }
    }

    int[][] list_otrong = new int[slotrong][2]; //Mảng lưu danh sách ô trống
    int count = 0;
    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            if (grid[i][j] == 0) {
                list_otrong[count][0] = i;
                list_otrong[count][1] = j;
                count++;
            }
        }
    }
    return list_otrong;
}
```

4.2.3. Giải thuật kiểm tra đúng/sai khi người chơi nhập vào một giá trị

Giải thuật sẽ kiểm tra các con số người dùng nhập vào có phạm quy tắc hay không, nếu phát hiện một số nào đó không hợp lệ thì hàm sẽ trả về False (ngược lại sẽ trả về True).

```
//Hàm kiểm tra giá trị nhập vào có hợp lệ hay không
public static boolean valid(int i, int j, int[][] grid) {
    // Kiểm tra hàng
    for (int column = 0; column < 9; column++) {
        if (column != j && grid[i][column] == grid[i][j]) {
            return false;
        }
    }

    // Kiểm tra cột
    for (int row = 0; row < 9; row++) {
        if (row != i && grid[row][j] == grid[i][j]) {
            return false;
        }
    }

    // Kiểm tra khối 3 x 3
    for (int row = (i / 3) * 3; row < (i / 3) * 3 + 3; row++) {
        for (int col = (j / 3) * 3; col < (j / 3) * 3 + 3; col++) {
            if (row != i && col != j && grid[row][col] == grid[i][j]) {
                return false;
            }
        }
    }
    return true;
}
```

4.2.4. Giải thuật quay lui-vết cạn để giải ma trận:

Giải thuật sẽ quay lui-vết cạn để giải ma trận (với tham số truyền vào là ma trận cần giải). Nếu ma trận truyền vào có thể giải ra kết quả sẽ trả về True (ngược lại sẽ trả về False).

```
//Hàm giải Sudoku theo thuật toán Quay lui vết cạn.
public static boolean search(int[][] grid) {
    int[][] list_otrong = getList_otrong(grid);
    int k = 0;
    boolean found = false;
    while(!found) {
        int i = list_otrong[k][0];
        int j = list_otrong[k][1];
        if (grid[i][j] == 0) {
            grid[i][j] = 1;
        }
        if (valid(i, j, grid)) {
            if (k + 1 == list_otrong.length) {
                found = true;
            } else {
                k++;
            }
        }
        else if (grid[i][j] < 9) {
            grid[i][j] = grid[i][j] + 1;
        }
        else {
            while (grid[i][j] == 9) {
                grid[i][j] = 0;
                if (k == 0) {
                    return false;
                }
                k--;
                i = list_otrong[k][0];
                j = list_otrong[k][1];
            }
            grid[i][j] = grid[i][j] + 1;
        }
    }
    return true;
}
```

4.2.5. Giải thuật lưu ma trận thành file Text:

Giải thuật sẽ lưu ma trận hiện tại ra file text với nội dung gồm: thời gian lưu file, level của ma trận, ma trận chưa giải và đáp án ma trận.

```
//Hàm xuất ma trận ra file txt
private void in(String dinh dang) {
    //Tạo thư mục để lưu file outPut.
    String link = "C:\\Users\\Dell\\Desktop\\CT23901_NLCS\\Log\\";
    File vitriluu = new File(link);

    //Kiểm tra thư mục lưu file đã tồn tại hay chưa, nếu chưa có thì tạo thư mục tương ứng với đường dẫn
    if(!vitriluu.exists())
        vitriluu.mkdir();

    try {
        Date dt = new Date();
        SimpleDateFormat d = new SimpleDateFormat("dd-MM-yyyy hh-mm-ss"); //Định dạng thời gian để lấy làm tên tệp phân biệt
        String time = d.format(dt);
        String tenfile = "Ma trận" + " (" + time + ").trim(); //Biến để lưu tên tệp
        System.out.println(tenfile);
        File fileDir = new File(link + tenfile + ".txt");
        try (Writer out = new BufferedWriter(new OutputStreamWriter(
            new FileOutputStream(fileDir), "UTF8"))) {
            out.append(dinh dang);
            out.flush();
            out.close();
        }
        JOptionPane.showMessageDialog(null, tenfile + ".txt đã được lưu tại \n" + link + "\nBạn có thể xem lại!");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, "Lỗi đường dẫn thư mục lưu tập tin, cài đặt lại!");
        System.out.println(e.getMessage());
    }
}

//Định dạng cho nội dung file txt lưu ma trận được in ra
private String dinh dang() {
    Date dt = new Date();
    SimpleDateFormat d = new SimpleDateFormat("dd/MM/yyyy hh:mm:ss");
    String time = d.format(dt);

    String matran = "Thời gian: ";
    matran += time + "\n";
    matran += "Level: " + level + "\n\n";

    for(int i=0; i<9; i++) {
        for (int j = 0; j < 9; j++) {
            matran+= temp[i][j] + " ";
        }
        matran += "\n";
    }

    matran += "\n          ***** \n";

    for(int i=0; i<9; i++) {
        matran += "\n";
        for (int j = 0; j < 9; j++) {
            matran+= grid[i][j];
            matran += " ";
        }
    }
    return matran;
}
```

5. CHƯƠNG V: TỔNG KẾT

Sau khoản thời gian tìm hiểu và thực hiện đề tài, thì trò chơi đã cơ bản được hoàn thành và đáp ứng được mục tiêu đã đề ra. Tuy đã hoàn thành được đề tài, nhưng với kiến thức và trình độ chưa cao, nên vẫn còn một số ưu khuyết điểm như sau:

- Ưu điểm:
 - Cơ bản đã cài đặt được thuật toán.
 - Chương trình đạt được các yêu cầu chức năng đã đặt ra.
 - Xây dựng được giao diện, dễ sử dụng.
- Nhược điểm:
 - Cài đặt thuật toán chưa được tối ưu.
 - Chỉ có thể giải Sudoku dạng chuẩn 9x9.
 - Chưa áp dụng heuristic để tối ưu hóa khả năng tính toán.

Hướng phát triển:

- Cần cải thiện giao diện hiển thị đẹp mắt, hợp lý.
- Cải thiện thuật toán nhằm giải tối ưu hơn.
- Thêm nhiều dạng Sudoku cao cấp hơn (12 x 12, 15 x 15, Sudoku X...) thay vì chỉ có dạng cơ bản là 9 x 9.

PHỤ LỤC

- Hướng dẫn sử dụng demo chi tiết: (đã có hướng dẫn sơ bộ ở mục 4.1)
- + Khi cài đặt chương trình, người dùng cần đặt lại đường dẫn vị trí lưu tập tin ma trận Sudoku trong mã nguồn chương trình.
- + Người chơi giải ma trận khá đơn giản bằng cách điền các con số từ 1 đến 9 vào 81 ô số, sao cho không vi phạm luật chơi thì người chơi sẽ dành chiến thắng.
- + Khi người chơi muốn xem đáp án của ván cờ thì click vào nút ĐÁP ÁN để xem kết quả của 81 ô cờ.
- + Di chuyển giữa các ô số bằng cách click chuột vào các ô, hoặc sử dụng các phím mũi tên trên bàn phím (hoặc phím Tab) để di chuyển giữa các ô số.

- Các biểu mẫu, chứng từ, công thức được sử dụng để thực hiện đề tài: Thuật toán Quay lui – Vết cạn để giải Game Sudoku.

TÀI LIỆU THAM KHẢO

- [1] Sudoku, Website tham khảo nguồn gốc lịch sử, Bách khoa toàn thư mở:
<https://vi.wikipedia.org/wiki/Sudoku>
- [2] B. Bernhardsson, Explicit Solutions to the N-queens Problem for All N, SIGART Bull. 2(2)(1991).
- [3] Đệ quy, Website Bách khoa toàn thư mở:
https://vi.wikipedia.org/wiki/%C4%90%E1%BB%87_quy
- [4] Giải thuật minimax, Website Giải thuật lập trình:
<http://www.giaithuatlaptrinh.com/?tag=sudoku>
- [5] Giải thuật quay lui quét cạn, Website Giải thuật lập trình:
<https://viblo.asia/p/thuat-toan-quay-lui-backtracking-bJzKmLbD59N>
- [6] Tìm kiếm theo chiều sâu, Website Blog Lập Trình: :
<https://bloglaptrinh2016.wordpress.com/2016/06/09/dfs-thuat-toan-tim-kiem-theo-chieu-sau/>
- [7] Website tra cứu mã màu CSS dùng trong chương trình:
<https://sites.google.com/site/wwwcaotongvn/css>