

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN TOÁN TỔ HỢP VÀ ĐỒ THỊ
SUDOKU SOLUTION**

Người hướng dẫn: **NGUYỄN CHÍ THIỆN**

Người thực hiện: **TRỊNH VÂN THƯƠNG – 51800128**

PHAN XUÂN SƠN – 51801018

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN TOÁN TỔ HỢP VÀ ĐỒ THỊ
SUDOKU SOLUTION**

Người hướng dẫn: **NGUYỄN CHÍ THIỆN**

Người thực hiện: **TRỊNH VÂN THƯƠNG – 51800128**

PHAN XUÂN SƠN – 51801018

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022

LỜI CẢM ƠN

Cảm ơn thầy Nguyễn Chí Thiện đã tận tình chỉ dạy và truyền đạt những kiến thức quý báu cho em trong suốt thời gian học tập vừa qua. Trong thời gian tham gia lớp học Toán tổ hợp và đồ thị của thầy, chúng em đã có thêm cho mình nhiều kiến thức bổ ích, tinh thần học tập hiệu quả, nghiêm túc. Đây chắc chắn sẽ là những kiến thức quý báu, là hành trang để chúng em có thể vững bước sau này. Trong bài báo cáo còn nhiều sai sót chúng em mong thầy tận tình chỉ bảo để có thể hoàn thiện hơn nữa. Chúng em xin chân thành cảm ơn.

ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là sản phẩm tiểu luận của riêng tôi và được sự hướng dẫn của thầy Nguyễn Chí Thiện. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung tiểu luận của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 10 tháng 04 năm 2022

Tác giả

Trịnh Vân Thương

Phan Xuân Sơn

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

Tp. Hồ Chí Minh, ngày tháng năm 2022
(kí và ghi họ tên)

Mục lục

LỜI CẢM ƠN.....	3
PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN.....	5
I. PHÁT BIỂU BÀI TOÁN:.....	7
II. CÁC BƯỚC THỰC HIỆN:.....	7
a) Bước 1: Kiểm tra tham số đầu vào:.....	7
b) Bước 2: Tạo ra các hoán vị của Latin Squares.....	7
c) Bước 3: Tạo lập sudokuBase3 và tiến hành tạo lập sudokuBase10:.....	9
d) Bước 4: Đổi các dòng 2 & 4, 3 & 7, 6 & 8.....	10
e) Bước 5: Tiến hành đực lỗ:.....	11
f) Bước 6: Ghi ra file:.....	11
g) Testcase:.....	11
i. Testcase 1:.....	11
ii. Testcase 2:.....	11
iii. Testcase 3:.....	12
iv. Testcase 4:.....	12
v. Testcase 5:.....	12
III. TỰ ĐÁNH GIÁ:.....	13
IV. TÀI LIỆU THAM KHẢO:.....	15
HẾT.....	15

I. PHÁT BIỂU BÀI TOÁN:

Mục đích thực hiện bài tiểu luận để hiện thực việc tạo ra 1 Sudoku puzzle 9x9 bằng cách sử dụng hoán vị, trong đó bao gồm 9 khối 3x3 nhỏ hơn. Các thành phần trong puzzle phải thỏa mãn một số luật trong trò chơi như:

- Các con số trong 1 hàng không được lặp lại.
- Các con số trong 1 cột không được lặp lại.
- Các con số trong 1 khối 3x3 không được lặp lại.

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

Hình 1. Sudoku Puzzle

II. CÁC BƯỚC THỰC HIỆN:

a) Bước 1: Kiểm tra tham số đầu vào:

Function:

checkArgument(inputAgr)

Bước đầu tiên của bài toán là kiểm tra tham số đầu vào của người dùng nhập từ command line. Trong bài toán này có 2 tham số đầu vào đó là: số lượng lỗi cần đọc và tên output file.

Nhiệm vụ của function này đó là thông báo cho người dùng nếu số lỗi nhập vào không hợp lệ.

Số lỗi hợp lệ nằm trong khoảng $[0, 81]$ và phải chia hết cho 9.

b) Bước 2: Tạo ra các hoán vị của Latin Squares

Function:

latinSquareGenerate(booleanMatrix)

checkAndCreate(booleanMatrix, element, prevElement, latinSquarePermutations, i)

Đầu tiên cần tạo ra 1 booleanMatrix (binary matrix, logical matrix) bao gồm các phần tử 0, 1, Trong booleanMatrix các khối 3x3 đối xứng nhau đồng thời các phần tử trong khối cũng đối xứng.

Boolean matrix:

```
[0,1,1,1,0,0,1,0,0]
[1,0,1,0,1,0,0,1,0]
[1,1,0,0,0,1,0,0,1]
[1,0,0,0,1,1,1,0,0]
[0,1,0,1,0,1,0,1,0]
[0,0,1,1,1,0,0,0,1]
[1,0,0,1,0,0,0,1,1]
[0,1,0,0,1,0,1,0,1]
[0,0,1,0,0,1,1,1,0]
```

Element là 1 phần tử được khởi tạo rỗng có kích thước bằng 9:

```
[0,0,0,0,0,0,0,0,0]
```

prevElement chứa các phần tử trước đó trong tham chiếu đã có giá trị khác 0 đến boolean matrix.

latinSquarePermutations lưu trữ tất cả các hoán vị của latin square.

Lần lượt duyệt qua từng phần tử trong boolean matrix, nếu phần tử tại vị trí tham chiếu khác 0 tiến hành gán giá trị cho phần tử đó trong khoảng từ [1, 3] đối chiếu với previous element, ví dụ nếu phần tử tại booleanMatrix[1, 0] có giá trị là 0 ta tiến hành gán giá trị cho phần tử này, previous element đầu tiên mặc định là 0 nên giá trị được gán tại phần tử này lúc này sẽ là 1.

Chạy tay thuật toán với n = 1:

Element [1,0,0,0,0,0,0,0,0]

i = 1, j = 0, prev = []

booleanMatrix[1][0] = 0

Gán giá trị trong khoảng [1, 3]

k = 1 khác prev

Cập nhật giá trị cho Element[1] = 1

Element [1,0,0,0,0,0,0,0,0]

Tăng biến đếm i lên 1 và gọi đệ quy lại checkAndCreate(...)

Element [1,0,0,0,0,0,0,0,0]

i = 2, j = 0, prev = []

booleanMatrix[2][0] = 1 khác 0

Cập nhật giá trị cho prev = [1]

i = 2, j = 1, prev = [0]

Gán giá trị trong khoảng [1, 3]

k = 1 đã có trong prev

k = 2 khác prev

Cập nhật giá trị cho Element[2] = 2

Element [1,2,0,0,0,0,0,0,0]

Tăng biến đếm i lên 1 và gọi đệ quy lại checkAndCreate(...)

Tiếp tục như vậy cho đến khi tham chiếu hết booleanMatrix, ta được tất cả các hoán vị:

```
[1, 2, 3, 2, 3, 1, 3, 1, 2]
[1, 2, 3, 3, 1, 2, 2, 3, 1]
[1, 3, 2, 2, 1, 3, 3, 2, 1]
[1, 3, 2, 3, 2, 1, 2, 1, 3]
[2, 1, 3, 1, 3, 2, 3, 2, 1]
[2, 1, 3, 3, 2, 1, 1, 3, 2]
[2, 3, 1, 1, 2, 3, 3, 1, 2]
[2, 3, 1, 3, 1, 2, 1, 2, 3]
[3, 1, 2, 1, 2, 3, 2, 3, 1]
[3, 1, 2, 2, 3, 1, 1, 2, 3]
[3, 2, 1, 1, 3, 2, 2, 1, 3]
[3, 2, 1, 2, 1, 3, 1, 3, 2]
```

Sau đó giảm mỗi phần tử trong matrix vừa được để có được dạng base 3 như mong muốn (gồm 3 phần tử 0, 1, 2).

```
[0, 1, 2, 1, 2, 0, 2, 0, 1]
[0, 1, 2, 2, 0, 1, 1, 2, 0]
[0, 2, 1, 1, 0, 2, 2, 1, 0]
[0, 2, 1, 2, 1, 0, 1, 0, 2]
[1, 0, 2, 0, 2, 1, 2, 1, 0]
[1, 0, 2, 2, 1, 0, 0, 2, 1]
[1, 2, 0, 0, 1, 2, 2, 0, 1]
[1, 2, 0, 2, 0, 1, 0, 1, 2]
[2, 0, 1, 0, 1, 2, 1, 2, 0]
[2, 0, 1, 1, 2, 0, 0, 1, 2]
[2, 1, 0, 0, 2, 1, 1, 0, 2]
[2, 1, 0, 1, 0, 2, 0, 2, 1]
```

c) Bước 3: Tạo lập sudokuBase3 và tiến hành tạo lập sudokuBase10:

Function:

scaleToBase10(sudokuBase3, scalerMatrix)

sudokuBase3 bao gồm 9 trên tổng số 12 khối Latin squares:

```
[2, 0, 1, 1, 2, 0, 0, 1, 2]
[0, 2, 1, 2, 1, 0, 1, 0, 2]
[1, 2, 0, 0, 1, 2, 2, 0, 1]
[2, 1, 0, 0, 2, 1, 1, 0, 2]
[0, 1, 2, 1, 2, 0, 2, 0, 1]
[0, 1, 2, 2, 0, 1, 1, 2, 0]
[1, 2, 0, 0, 1, 2, 2, 0, 1]
[2, 0, 1, 0, 1, 2, 1, 2, 0]
[1, 0, 2, 0, 2, 1, 2, 1, 0]
```

scalerMatrix là 1 Latin square:

[1, 2, 0, 0, 1, 2, 2, 0, 1]

Tiến hành scale sudokuBase3 thành sudokuBase10:
Khối thứ 1:

$1 \times 3 + 2 + 1 = 6$
 $1 \times 3 + 0 + 1 = 4$
 $1 \times 3 + 1 + 1 = 5$
 $1 \times 3 + 1 + 1 = 5$
 $1 \times 3 + 2 + 1 = 6$
 $1 \times 3 + 0 + 1 = 4$
 $1 \times 3 + 0 + 1 = 4$
 $1 \times 3 + 1 + 1 = 5$
 $1 \times 3 + 2 + 1 = 6$

Sau khi scale tất cả các khối ta được sudokuBase10:

[6, 4, 5, 5, 6, 4, 4, 5, 6]
 [7, 9, 8, 9, 8, 7, 8, 7, 9]
 [2, 3, 1, 1, 2, 3, 3, 1, 2]
 [3, 2, 1, 1, 3, 2, 2, 1, 3]
 [4, 5, 6, 5, 6, 4, 6, 4, 5]
 [7, 8, 9, 9, 7, 8, 8, 9, 7]
 [8, 9, 7, 7, 8, 9, 9, 7, 8]
 [3, 1, 2, 1, 2, 3, 2, 3, 1]
 [5, 4, 6, 4, 6, 5, 6, 5, 4]

d) Bước 4: Đổi các dòng 2 & 4, 3 & 7, 6 & 8

Function:

rowSwapping(sudokuBase10, firstRow, secondRow)

Sau khi tạo thành sudokuBase10 thì trong giữa các dòng có sự lặp lại

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

Việc trao đổi các dòng được thực hiện trong hàm rowSwapping, dữ liệu của từng dòng sẽ được lưu vào biến tạm và trao đổi cho nhau. Kết quả trả về là sudokuBase10 với dữ liệu các dòng không còn trùng lặp.

e) Bước 5: Tiến hành đục lỗ:

Function: *diggingHoles(sudokuBase10, numberOfHoles)*

Số lỗ đục phải chia hết cho 9. Số lỗ đục ở mỗi khối được tính bằng `numberOfHoles` : 9. Ví dụ `numberOfHoles = 27` thì số lượng lỗ đục ở mỗi khối sẽ bằng 3.

Hàm `diggingHoles` sẽ duyệt tuần tự qua từng khối và ngẫu nhiên thay thế n phần tử trong khối thành giá trị 0.

f) Bước 6: Ghi ra file:

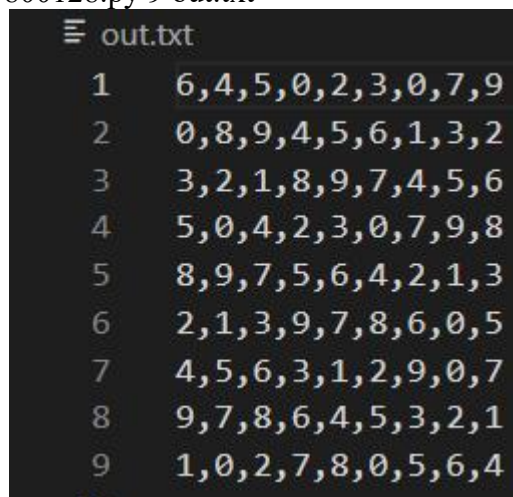
Function: *outputFile(sudoku, outputFileName)*

Tiến hành ghi giá trị của puzzle ra file với tên file là tham số thứ 2 mà người dùng nhập từ command line.

g) Testcase:

i. Testcase 1:

`python assignment_51800128.py 9 out.txt`



	out.txt
1	6,4,5,0,2,3,0,7,9
2	0,8,9,4,5,6,1,3,2
3	3,2,1,8,9,7,4,5,6
4	5,0,4,2,3,0,7,9,8
5	8,9,7,5,6,4,2,1,3
6	2,1,3,9,7,8,6,0,5
7	4,5,6,3,1,2,9,0,7
8	9,7,8,6,4,5,3,2,1
9	1,0,2,7,8,0,5,6,4

Hình 2. Kết quả testcase 1

ii. Testcase 2:

`python assignment_51800128.py 18 out.txt`

```

≡ out.txt
1  1,3,0,6,4,0,0,7,8
2  0,4,5,7,8,9,2,3,1
3  9,8,7,3,2,0,5,0,6
4  2,0,3,4,5,6,8,0,7
5  0,5,6,8,9,0,1,2,3
6  7,9,8,0,1,3,4,0,5
7  3,2,0,5,6,4,7,8,9
8  5,0,4,9,7,8,3,0,2
9  8,7,9,0,0,2,6,0,4

```

Hình 3. Kết quả testcase 2

iii. Testcase 3:

python assignment_51800128.py 27 out.txt

```

≡ out.txt
1  0,1,3,6,0,0,0,8,0
2  0,0,8,2,1,3,6,4,5
3  5,4,6,0,8,7,2,1,0
4  3,2,1,5,4,6,9,0,8
5  0,0,9,1,0,2,4,0,6
6  0,5,4,0,7,0,1,3,0
7  1,3,2,4,6,5,8,9,0
8  0,0,0,0,0,0,5,0,4
9  4,6,5,7,9,8,3,0,1

```

Hình 4. Kết quả testcase 3

iv. Testcase 4:

python assignment_51800128.py 36 out.txt

```

≡ out.txt
1  0,2,3,0,7,0,0,5,6
2  9,0,0,0,5,6,0,1,3
3  6,4,0,0,2,1,9,0,0
4  0,0,0,9,8,7,0,4,5
5  8,0,7,5,6,0,3,2,1
6  4,5,6,0,0,0,0,0,0
7  2,0,0,7,0,8,5,0,0
8  7,0,9,0,4,0,1,3,0
9  0,6,4,2,0,3,7,0,8

```

Hình 5. Kết quả testcase 4

v. Testcase 5:

python assignment_51800128.py 45 out.txt

```

≡ out.txt
1  0,0,0,0,0,6,1,3,0
2  2,0,1,8,7,9,6,0,0
3  5,0,6,0,0,0,7,0,0
4  0,0,0,0,0,0,0,2,0
5  0,1,2,9,0,7,0,5,6
6  4,0,5,0,3,1,0,0,7
7  0,0,0,0,0,5,2,1,0
8  0,2,3,0,0,8,0,6,4
9  6,5,0,1,0,3,0,0,0

```

Hình 6. Kết quả testcase 5

III. TỰ ĐÁNH GIÁ:

Nội dung tiêu chí	Thang đánh giá	1	2	3	Tự đánh giá	Lý do
	Điểm /10	0 điểm	1/2 tổng điểm	Trọn điểm		
1/ Hình thức	4.0					
Bài báo cáo ngắn nội dung bài tiểu luận (docx).	3.0	Thiếu bài báo cáo	Có báo cáo, nhưng còn chưa rõ ràng, thiếu các mục cơ bản	Rõ ràng, logic (0.5đ), chứa đầy đủ các nội dung bao gồm phát biểu bài toán (0.5đ), mô tả các bước giải bài toán (0.5đ), code (0.5đ), 5 test case (0.5đ), bảng sinh viên tự đánh giá (0.5đ)	3.0	
Xuất nhập đúng định dạng yêu cầu.	1.0	Xuất nhập không theo định dạng		Xuất nhập theo định dạng	1.0	
2/ Nội dung	6.0					
Tạo được tất cả các	2.0	Không tạo được tất cả các		Tạo được tất cả các ô Latin	2.0	

Nội dung tiêu chí	Thang đánh giá	1	2	3	Tự đánh giá	Lý do
ô Latin Squares.		ô Latin Squares đánh giá bằng cách xuất ra được kết quả		Squares đánh giá bằng cách xuất ra được kết quả		
Thực hiện được phần Digging hole	1.0	Không thực hiện được phần Digging hole đánh giá bằng cách xuất ra được kết quả		Thực hiện được phần Digging hole đánh giá bằng cách xuất ra được kết quả	1.0	
Code	1.0	Code không rõ ràng, thiết kế không theo một programming paradigm nhất quán		Theo một programming paradigm nhất quán. Đầy đủ, rõ ràng, bao gồm các phần cơ bản như chức năng của hàm, mô tả đầu vào, đầu ra của mỗi hàm (kiểu dữ liệu, dùng để lưu trữ gì)	1.0	
Vấn đáp	2.0	Không trả lời được câu hỏi	Trả lời được một số các câu hỏi	Trả lời được tất cả các câu hỏi	2.0	Phần này mặc định là 2.0 cho đến khi vấn đáp với thầy(nếu có)
3/ Thái độ người học	0.0					
Nộp bài đúng hạn, chủ động trong việc hoàn thành tiểu luận	0.0	Nộp muộn (-1đ)		Nộp đúng hạn (0đ)		
Tổng điểm	10	Kết quả:			10	

IV. TÀI LIỆU THAM KHẢO:

1. <https://digitalcommons.library.umaine.edu/cgi/viewcontent.cgi?article=1393&context=honors>
2. <https://www.cut-the-knot.org/arithmetic/latin.shtml>
3. https://en.wikipedia.org/wiki/Mutually_orthogonal_Latin_squares
4. <https://www.sciencedirect.com/science/article/pii/S0166218X21001062>

HẾT