MỤC LỤC

[*DANH MỤC HÌNH ẢNH 3*](#_Toc26390764)

[*LỜI NÓI ĐẦU 4*](#_Toc26390765)

[*1. GIỚI THIỆU 5*](#_Toc26390766)

[*1.1. Giới thiệu đề tài* 5](#_Toc26390767)

[*1.2. Phương pháp nghiên cứu* 6](#_Toc26390769)

[*1.3. Mục tiêu nghiên cứu* 6](#_Toc26390770)

[*2. CƠ SỞ LÝ THUYẾT 7*](#_Toc26390771)

[*2.1. Nghiên cứu lý thuyết* 7](#_Toc26390772)

[*2.2. Một số khái niệm* 7](#_Toc26390773)

[*2.3. Mô tả bài toán* 8](#_Toc26390774)

[2.3.1. Xác định thông tin vào (Input) : 8](#_Toc26390775)

[2.3.2. Xác định thông tin ra (Output) : 8](#_Toc26390776)

[2.3.3. Xác định vị trí di chuyển từ 2 đến n\*n của quân mã 8](#_Toc26390777)

[*2.4. Xây dựng cấu trúc dữ liệu cho bài toán* 9](#_Toc26390779)

[*2.3. Giải pháp – Thuật toán đã có (Đệ quy)* 10](#_Toc26390780)

[*2.4. Giải pháp – Thuật toán đề xuất (Quay lui dùng Stack)* 11](#_Toc26390781)

[*3. XÂY DỰNG CHƯƠNG TRÌNH “MÃ ĐI TUẦN” 12*](#_Toc26390782)

[*3.1. Giới thiệu chương trình* 12](#_Toc26390783)

[*3.2. Chạy thử chương trình* 13](#_Toc26390784)

[*3.3. Một vài vấn đề nhỏ của chương trình* 14](#_Toc26390786)

[*3.4. Kết quả đạt được* 15](#_Toc26390787)

[3.4.1. Khó khăn 15](#_Toc26390788)

[3.4.2. Cách khắc phục khó khăn 15](#_Toc26390789)

[3.4.3. Tự đánh giá kết quả 16](#_Toc26390790)

[*3.5. Code toàn bộ chương trình “Mã đi tuần” trên C++* 17](#_Toc26390791)

[*4. LỜI KẾT 22*](#_Toc26390792)

[*TÀI LIỆU THAM KHẢO 23*](#_Toc26390793)

# DANH MỤC HÌNH ẢNH

[Hình 1. 1. Hình ảnh mô tả “Hành trình đóng” của con mã 5](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615882)

[Hình 1. 2. Hình ảnh mô tả các nước đi của quân mã trên bàn cờ 8 \* 8 5](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615883)

[Hình 2. 1. Cách di chuyển của quân mã trên bàn cờ vua 10](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615888)

[Hình 2. 2. Những vị trí một quân mã có thể đi đến được 10](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615889)

[Hình 3. 1. Chương trình khi mới khởi động lên 13](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615893)

[Hình 3. 2. Nhập vị trí bắt đầu cho quân mã 14](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615894)

[Hình 3. 3. Quân mã bắt đầu đi tuần 15](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615895)

[Hình 3. 4. Chương trình báo lỗi (Not Responding) 15](file:///C:\Users\acer\Desktop\ĐỒ%20ÁN\Báo%20cáo\BAO%20CAO.docx#_Toc26615896)

# LỜI NÓI ĐẦU

---------------------------------oOo----------------------------------

***Ngày nay***, Tin học ngày càng phát triển nhanh chóng và được ứng dụng rộng rãi trong mọi lĩnh vực của đời sống xã hội, việc học và nắm bắt công nghệ mới đặc biệt là công nghệ thông tin ngày càng trở nên hết sức cần thiết. Đối với sinh viên trong ngành càng phải tích cực học tập, nắm vững mọi kiến thức về công nghệ thông tin, trong đó môn học **cấu trúc dữ liệu và giải thuật** được xem là cơ sở, nền tảng đầu tiên, môn học giúp ta rèn luyện tư duy qua đó giúp ta hiểu sâu hơn về các ngôn ngữ, cách thức lập trình.

**Cấu trúc dữ liệu** giúp cho sinh viên hiểu được tầm quan trọng của giải thuật và cách tổ chức cấu trúc dữ liệu để giải quyết những bài toán cụ thể.

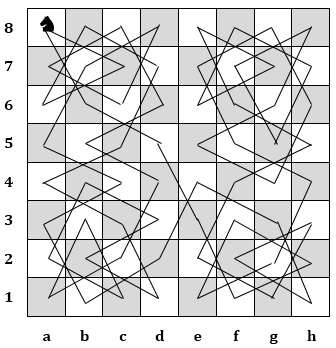
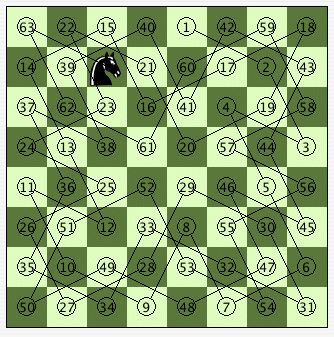
Sau một thời gian học tập và nghiên cứu ngôn ngữ lập trình C (*C Programming Language*) và C#, môn Cấu trúc dữ liệu, phân tích thiết kế thuật toán, để nắm bắt những kiến thức đã học một cách tốt hơn, nhóm tác giả đã thực hiện đề tài: “**Mã Đi Tuần**” (*Knight’s Tour*).

Trong quá trình thực hiện đề tài, mặc dù đã ra sức nghiên cứu và cố gắng nhưng chắc chắn khó tránh khỏi những thiếu sót, nhóm tác giả rất mong nhận được những sự chỉ dẫn, ý kiến đóng góp của quý thầy cô để nhóm ngày càng hoàn thiện hơn. Đồng thời nhóm tác giả xin gởi lời cảm ơn chân thành đến thầy **Trần Công Tú** đã giúp nhóm hoàn thành đề tài này.

**Nhóm sinh viên thực hiện**

# GIỚI THIỆU

## Giới thiệu đề tài

* **“Mã đi tuần”** (*Knight’s Tour*) là một bài toán liên quan đến một quân mã trên một bàn cờ vua. Quân mã được đặt tại một vị trí trống bất kỳ trên bàn cờ và di chuyển theo luật của cờ vua. Nó nhảy qua mỗi ô trên bàn cờ chính xác một lần. Hành trình của quân mã được gọi là “**Hành trình đóng**” (*Closed Tour*) nếu vị trí kết thúc của quân mã tấn công được vị trí xuất phát của nó (để quân mã có thể đi tuần tiếp một vòng nữa ngay lập tức). Nếu không thì được gọi là “**Hành trình mở**”
* Lập trình để giải quyết bài toán là một vấn đề phổ biến của các sinh viên thuộc các ngành Khoa học Máy tính. Những biến thể của bài toán “**Mã đi tuần**” bao gồm việc giải quyết bài toán trên các kiểu bàn cờ khác kiểu 8\*8 (*n\*n, hình chữ nhật,…*). Ở phạm vi đồ án này, nhóm tác giả chỉ phát triển “**hành trình mở**”của quân mã trên bàn cờ 8\*8.

Hình 1. 1. Hình ảnh mô tả “Hành trình đóng” của con mã

Hình 1. 2. Hình ảnh mô tả các nước đi của quân mã trên bàn cờ 8 \* 8

## Phương pháp nghiên cứu

* *Chọn đề tài nghiên cứu:* Việc nghiên cứu bài toán Mã Đi Tuần vừa giúp nhóm tác giả có thể áp dụng những kiến thức đã học trong môn học đề giải quyết bài toán. Cụ thể là sử dụng Stack, ngoài ra còn hiểu biết thêm về cách thức lập trình giao diện trên Window Form.
* *Lên kế hoạch thực hiện:* Để thực hiện tốt đề tài đưa ra thì nhóm tác giả đã lên kế hoạch cụ thể cho từng công việc ngay từ khi bắt đầu lên ý tưởng. Lên kế hoạch để giúp nhóm tác giả có một hướng nhìn cụ thể hơn về những gì nhóm chuẩn bị làm giúp tránh mất thời gian và công việc được hiệu quả hơn.
* *Tìm kiếm tài liệu:* Luôn chủ động tìm kiếm tài liệu, thông tin trên các diễn đàn, trang mạng xã hội, sách báo, bạn bè và thậm chí là giảnh viên hướng dẫn đề tài cho nhóm. Không nên dựa dẫm, phụ thuộc vào những gì người khác cho mình sẵn mà nhóm phải chủ động tìm tòi, học hỏi thì mới có được một kết quả như mong muốn.
* *Chọn lọc và sắp xếp tài liệu:* Sau khi đã có được một lượng tài liệu tương đối, nhóm cần đọc để chọn lọc lại. Cần đọc tất cả các bài đã có. Đánh dấu những ý quan trọng. Ghi chú, tóm tắt một cách có hệ thống. Sắp xếp theo một trật tự phù hợp với thói quen hoặc ý đồ trình bày của nhóm.

## Mục tiêu nghiên cứu

* Nhóm tác giả muốn tạo ra một chương trình có thể giải được bài toán Mã Đi Tuần một cách nhanh chóng. Vẫn có thể dựa trên lý thuyết đã được học trong môn học mà vẫn sử dụng những thuật toán cũ để tạo ra một sự kết hợp mới và hiệu quả hơn.
* Ngoài ra còn muốn phát triển chương trình thành một trò chơi có thể chơi một hoặc hai người. Giúp đưa bài toàn cổ này đến gần gũi với mọi người hơn so với việc chỉ những ai học Công Nghệ Thông Tin mới biết bài toán hay này.

# CƠ SỞ LÝ THUYẾT

## Nghiên cứu lý thuyết

* Các tài liệu tham khảo sớm nhất để giải quyết bài toán xuất hiện từ thế kỷ thứ IX. Nhưng giải thuật đầu tiên đầy đủ cho bài toán về hành trình của quân mã là “**Giải thuật Warnsdorff**” (*Warnsdorff’s algorithm*), công bố lần đầu năm 1823 bởi **H. C. Warnsdorff.** Thuật toán **Warnsdorff** được biết đến như là phương pháp **Heuristic** của “**Mã đi tuần**”.
* Trên bàn cờ cỡ 8 x 8 có đúng **26,534,728,821,064** lời giải. Và số cách giải là **9.862** đối với bàn cờ cỡ 6 \* 6. Không có lời giải hành trình đóng cho các bàn cờ có cỡ nhỏ hơn.

## Một số khái niệm

* *Quay lui (Backtracking):* là một chiến lược tìm kiếm lời giải cho các bài toán thỏa mãn ràng buộc. Người đầu tiên đề ra thuật ngữ này là nhà toán học người Mỹ D. H. Lehmer vào những năm 1950. 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 củaphương pháp này là nhiều cài đặt tránh được việc phải thử nhiều tổ hợp chưa hoàn chỉnh, và nhờ đó giảm thời gian chạy. Phương pháp quay lui có quan hệ chặt chẽ với tìm kiếm tổ hợp.
* *Stack:* là một cấu trúc dữ liệu trừu tượng hoạt động theo nguyên lý "vào sau ra trước" (Last In First Out (LIFO)). Một ngăn xếp là một cấu trúc dữ liệu dạng thùng chứa (container) của các phần tử (thường gọi là các nút (node)) và có hai phép toán cơ bản là push and pop. Push bổ sung một phần tử vào đỉnh (top) của ngăn xếp, nghĩa là sau các phần tử đã có trong ngăn xếp. Pop giải phóng và trả về phần tử đang đứng ở đỉnh của ngăn xếp. Trong stack, các đối tượng có thể được thêm vào stack bất kỳ lúc nào nhưng chỉ có đối tượng thêm vào sau cùng mới được phép lấy ra khỏi stack. Ngoài ra, stack cũng hỗ trợ một số thao tác khác:
* isEmpty(): Kiểm tra xem stack có rỗng không.
* Top(): Trả về giá trị của phần tử nằm ở đầu stack mà không hủy nó khỏi stack.

## Mô tả bài toán

### Xác định thông tin vào (Input) :

* Một bảng vuông (bàn cờ) kích thước n\*n.
* Tọa độ xuất phát của quân mã A(x,y).

### Xác định thông tin ra (Output) :

* Một bảng vuông (bàn cờ) có đánh dấu vị trí theo thứ tự từ 1 (vị trí xuất phát) đến n\*n (vị trí kết thúc) của quân mã.
* Từ vị trí thứ k sang vị trí thứ k + 1 phải theo đúng luật di chuyển quân mã trong cờ vua.

### Xác định vị trí di chuyển từ 2 đến n\*n của quân mã

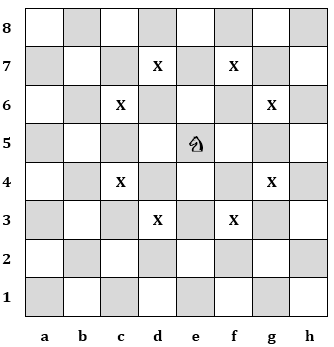
* Di chuyển đúng luật (Tới 1 ô trống và nằm trong phạm vi bàn cờ).
* Mỗi ô chỉ được đi qua 1 lần.
* Vị trí cuối cùng (n\*n) có thể nhảy tới vị trí xuất phát.

## cavalier

Hình 2. 1. Cách di chuyển của quân mã trên bàn cờ vua

## Xây dựng cấu trúc dữ liệu cho bài toán

* Biểu diễn bàn cờ bằng một ma trận K[n][n].
* Giá trị của mỗi ô K[i][j] (0 <= i,j < n) trên bàn cờ như sau :
  + K[i][j] = 0 : Ô (i,j) chưa được quân mã đi qua.
  + K[i][j] = sobuocdichuyen : Ô (i, j) được quân mã đi đến ở nước thứ sobuocdichuyen (1 <= t <= n\*n).
* Cấu trúc dữ liệu cho mỗi bước đi của quân mã :
  + Nước đi có thể của con mã được phân tích như sau:



Hình 2. 2. Những vị trí một quân mã có thể đi đến được

* + Từ 1 vị trí (i, j) trên bàn cờ thì có tối đa 8 vị trí (u, v) khác nhau cho quân mã nhảy tới. Việc thu được vị trí (u,v) từ (i,j) có thể được giải quyết bằng phương pháp cộng thêm độ chênh lệch về tọa độ chứa sẵn trong 2 mảng phụ hằng sau :

X**[ ] =** { 1,2,2,1,-1,-2,-2,-1 };

**Y[ ] =** { 2,1,-1,-2,-2,-1,1,2 };

Ví dụ : Vị trí (xx, yy) = (x + X[1], y + Y[1]) ,….

* + Vị trí (xx ,yy) cần tìm bị ràng buộc bởi 2 điều kiện :

***Thứ 1***, (xx, yy) phải còn nằm trong bàn cờ, tức là (0 <= xx < n) và (0 <= yy < n).

***Thứ 2***, A[xx][yy] phải bằng 0, tức là vị trí này chưa được quân mã đi qua.

## Giải pháp – Thuật toán đã có (Đệ quy)

Thông thường, ta sẽ nghĩ đến thuật toán **đệ quy** để giải quyết bài toán “**Mã đi tuần**” (*phương pháp vét cạn*). Ta sẽ xét đến những hạn chế còn mắc phải của phương pháp này sau. Thuật toán đệ quy như sau :

* Bắt đầu từ vị trí xuất phát.
* Từ vị trí thứ (k) ta đi tìm vị trí thứ (k+1).
* Sử dụng 2 mảng phụ để tìm các khả năng của vị trí thứ (k + 1). Với mỗi khả năng (k +1) ta thực hiện :
  + Nếu là vị trí thứ n\*n thì kết thúc hành trình và có được kết quả.
  + Nếu chưa là vị trí thứ n\*n thì tiếp tục tìm vị trí thứ (k + 2).
  + Nếu hết đường đi tại vị trí thứ (k + 1), tức là cả 8 khả năng đều bất khả thi thì quay lại, tìm theo hướng (k + 1) khác.
* Những hạn chế còn mắc phải của thuật toán đệ quy:
* Vì sử dụng vét cạn nên chỉ áp dụng được với những bài toán với bàn cờ có kích cỡ nhỏ.
* Trường hợp xấu nhất độ phức tạp của bài toán có thể là O(n64)

## Giải pháp – Thuật toán đề xuất (Quay lui dùng Stack)

* Nhận thấy ở thuật toán ***đệ quy***, hạn chế là do ta không phân biệt độ ưu tiên của các hướng đi có thể có từ vị trí (i,j). Có tối đa 8 vị trí (u,v) mà từ (i,j) có thể nhảy tới, ta phải chọn ra hướng đi nào tốt nhất, dễ nhất để hướng cho quân mã.
* Giải pháp: ***Quay lui dùng Stack*** 
  + Đầu tiên ta tạo một mảng Banco[][], để lưu sobuocdichuyen (con cờ đã di chuyển đến bước này).
  + Tạo mảng một chiều a[] xem như là một **Stack**, dùng để lưu kiểu di chuyển của con cờ tạo điều kiện quay lui khi con cờ không thể đi tiếp.
  + Biến xx , yy dùng để thử nước đi vị trí tiếp theo. Biến x, y là vị trí hiện tại.
  + Mảng dùng để lưu các trường hợp di chuyển của con cờ.

X[] = { 1,2,2,1,-1,-2,-2,-1 };

Y[] = { 2,1,-1,-2,-2,-1,1,2 };

* Ta xây dựng phép thử như sau:

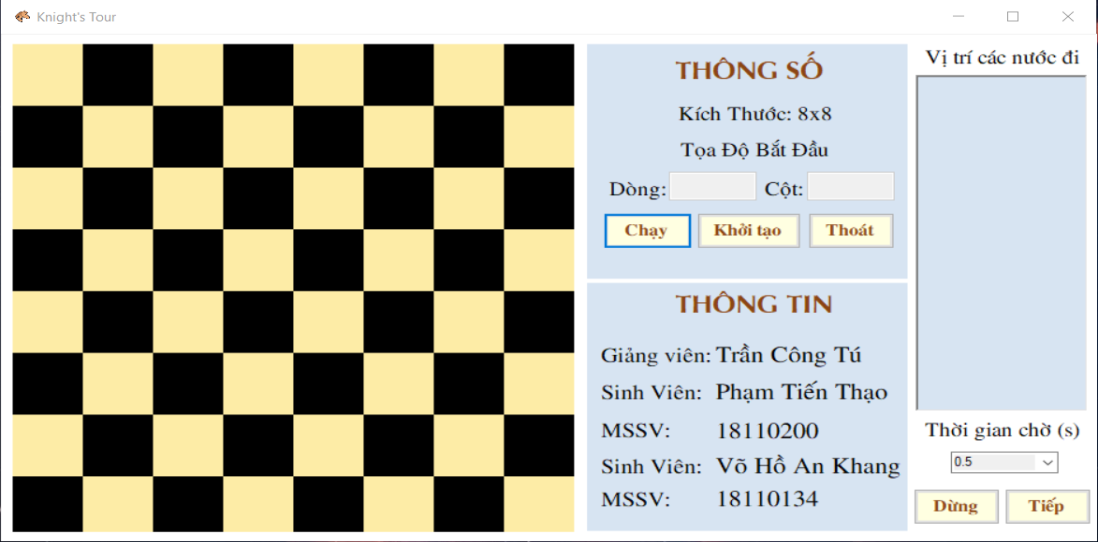
xx = x + X[num] (num chạy 0 -> 8)

yy = y + Y[num]

* Nếu nước đi phù hợp (ô trống và vẫn trong phạm vi bàn cờ) thì biến sobuocdichuyen sẽ tăng 1 đơn vị, vị trí của con cờ sẽ được gán (a[top].x, a[top].y), kết thúc vòng lặp và đi đến ô tiếp theo. Nếu không phù hợp thì biến sobuocdichuyen sẽ giảm 1 đơn vị và xét lại ô phía trước, ta gọi đây là quay lui. Nếu quay lui thì sẽ bắt đầu một kiểu di chuyển khác (>a[top].num) của ô phía trước (a[top] – con cờ ở trên cùng của stack).
* Trong điều kiện để thử nước đi phù hợp, ta có dùng thêm kỹ thuật ***lính canh*.** Tức là ta sẽ tạo bàn cờ 12 x 12. Và bàn cờ 8 x 8 nằm ở trung tâm. Viền ngoài của bàn cờ được đánh dấu là ô -1 (Không trống). Ta kiểm tra bằng điều kiện (num < 8 && Banco[xx][yy] == 0) thay vì (num < 8 && xx >= 0 && xx < N && yy >= 0 && yy<N && Banco[xx][yy] == 0).
* Giải thuật chính được sử dụng trong đề xuất này là ***quay lui*** và cấu trúc chính được sử dụng là ***stack.*** Tuy nhiên, đây không phải là giải thuật nhanh nhất để giải bài toán này, theo như nhóm tác giả tìm hiểu thì là phương pháp **Heuristic** bởi vì nó có thể chọn lọc ra được thứ tự ưu tiên cho các vị trí tiếp theo, giúp rút ngắn thời gian chương trình phải làm việc lại và đưa ra kết quả nhanh nhất.
* Mặc dù ***quay lui***  không phải là thuật toán nhanh nhất để giải bài toán ***Mã đi tuần*** nhưng đây lại là thuật toán thích hợp nhất để có thể áp dụng ***stack*** vào việc giải bài toán này, phù hợp với nội dung được tìm hiểu ở môn học.

# XÂY DỰNG CHƯƠNG TRÌNH “MÃ ĐI TUẦN”

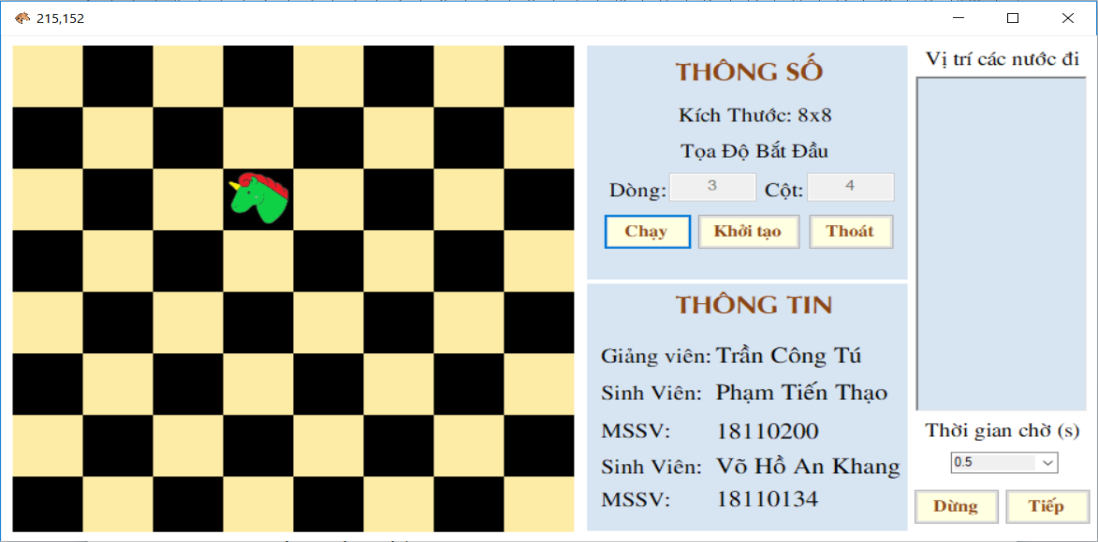
## Giới thiệu chương trình



Hình 3. 1. Chương trình khi mới khởi động lên

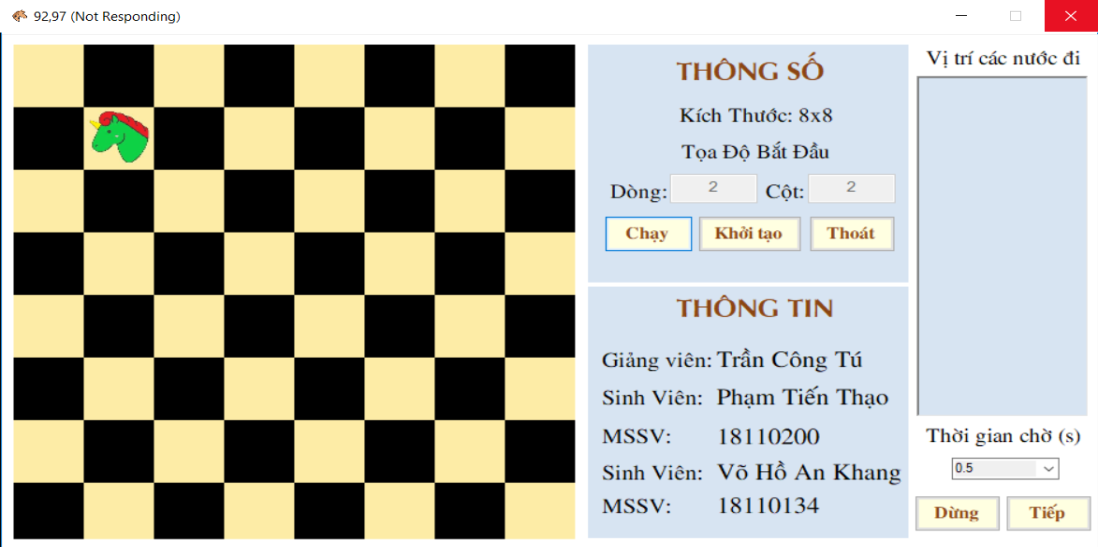
* Tên chương trình: ***Knight’s Tour***
* Trong chương trình có ghi các thông tin của sinh viên thực hiện và giảng viên hướng dẫn.
* Có các nút nhấn:
* **Chạy**: Bắt đầu chương trình.
* **Khởi tạo**: Để khởi tạo lại chương trình lúc chưa nhập toạ độ cho quân mã.
* **Thoát**: Để thoát chương trình.
* **Dừng**: Trong khi quân mã đang chạy ta nhấn **Dừng** quân mã sẽ dừng lại đúng vị trí nó đang đứng.
* **Tiếp**: Tiếp tục vị trí đang đứng khi ta nhấn nút **Dừng** trước đó.

## Chạy thử chương trình



Hình 3. 2. Nhập vị trí bắt đầu cho quân mã

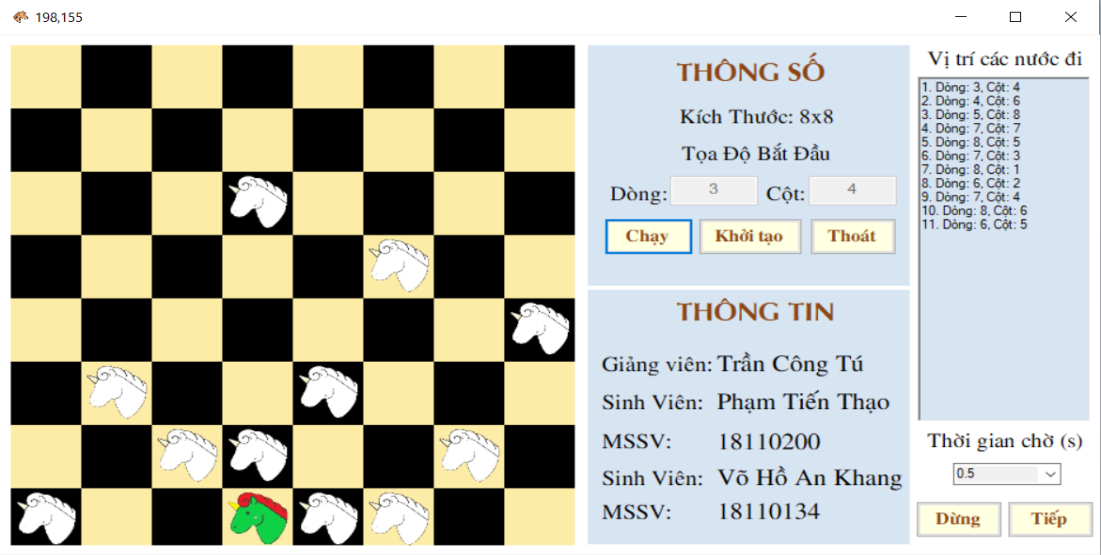
* Để bắt đầu chương trình ta click vào một ô bất kì trên bàn cờ, khi đó tọa độ ở vị trí **Dòng, Cột** sẽ được cập nhật tương ứng với tọa độ ta vừa chọn. (Cách làm này để giảm bớt việc nhập sai, nhập thiếu của người dùng)
* Sau đó nhấn nút **Chạy** thì con mã trên bàn cờ sẽ bắt đầu di chuyển theo đúng quy luật di chuyển của con mã trên bàn cờ vua, đồng thời bên ô **Vị trí các nước đi** cũng sẽ cập nhật vị trí mà con mã đang đi qua, cứ thế cho đến hết bàn cờ.



Hình 3. 3. Quân mã bắt đầu đi tuần

* Ngoài ra, còn 1 công cụ nữa giúp ta có thể lựa chọn được thời gian sau mỗi bước đi của con mã đó là **Thời gian chờ (s)**. Mặc định là 0.5s tức là sau mỗi 0.5s quân mã sẽ nhảy tới vị trí tiếp theo, ở đây ta có thể lựa chọn các giá trị **0.5, 1, 2, 3, 4, 5** để phù hợp với nhu cầu sử dụng của mỗi người.

## Một vài vấn đề nhỏ của chương trình



Hình 3. 4. Chương trình báo lỗi (Not Responding)

* Tuy nhiên, chương trình không thể chạy hết được tất cả các vị trí mà chỉ chạy được một số vị trí tiêu biểu như: ***(1,1), (1,5), (1,8), (3,4), (4,1), (4,8), (5,8), (6,5), (7,3), (8,1), (8,8),…***trong đó vị trí ***(3,4)***là đưa ra kết quả nhanh nhất.
* Những vị trí khác khi chạy thì chương trình sẽ báo ***(Not Responding)****,* đây không phải là thông báo của thuật toán sai mà do chương trình chờ quá lâu nhưng không thấy kết quả trả về nên đã thông báo như vậy. Như đã nói ở trên, đây không phải là thuật toán nhanh nhất để giải bài toán này nên một số vị trí rất lâu để có thể trả về kết quả. Nhóm tác giả đã test trên màn hình ***Console*** những vị trí đó vẫn ra kết quả nhưng thường là khoảng 5 – 10 phút, vị trí lâu nhất là vị trí *(2,2)* mất khoảng 56 phút để trả về kết quả.

## Kết quả đạt được

### Khó khăn

* Phải tìm hiểu kiến thức về Window Form (Trình làm giao diện trên C#), đây chính là trở ngại lớn nhất của đồ án. Vì nhóm tác giả chưa bao giờ được học qua việc làm giao diện, bắt tay vào làm đồ án là phải tìm hiểu từ những thứ đơn giản nhất của Window Form, điều đó làm cho tiến độ thực hiện đồ án theo kế hoạch bị giảm đi rất nhiều.
* Không có nhiều thời gian để làm việc chung với nhau, chỉ những lúc rảnh ở trường mới có thể làm việc nhóm được, tuy nhiên thời gian rảnh không nhiều do vậy công việc thực hiện không đạt được kết quả như mong muốn.
* Bất đồng quan điểm cá nhân, thời gian đầu cả 2 không ai chịu nghe ai, luôn cho rằng ý kiến của mình đúng cho nên chưa có tiếng nói chung, công việc cũng không được hiệu quả.

### Cách khắc phục khó khăn

* Phần tìm hiểu về lập trình giao diện trên C# cả 2 đã bỏ ra 1/3 thời gian trong suốt quá trình để tìm hiểu riêng về từng tính năng và cách sử dụng của từng hàm. Việc này tuy mất nhiều thời gian nhưng nó mang lại định hướng chính xác khi bắt tay vào viết thuật toán sao cho hợp lí.
* Ngay từ đầu không viết chung trên Window Form mà viết riêng từng phần. Viết code thuật toán trên C++ trước, test code kĩ càng làm sao cho thuật toán tối ưu nhất có thể, sau đó mới thiết kế giao diện và ghép hai phần riêng biệt đó lại với nhau. Để làm được điều đó thì ta phải hiểu rõ về giao diện trên Window Form như đã nói ở trên mới có thể làm được.
* Để tránh mất thời gian trong quá trình thực hiện đồ án và hiệu quả công việc cao thì nhóm tác giả sẽ trao đổi trên Internet trước, liệt kê những việc cần giải quyết vào buổi tiếp theo, khi làm như vậy thì những thời gian rảnh ở trường sẽ làm việc hiệu quả và đúng trọng tâm hơn, tránh được việc mất thời gian mà công việc không hiệu quả.
* Để không gây tranh cãi trong quá trình thực hiện đồ án thì cả hai đã phải ngồi lại và nói chuyện với nhau rõ ràng, mỗi người sẽ hạ bớt một phần cái tôi của mình xuống để trao đổi. Dẫu biết mỗi người đưa ra quan điểm riêng chỉ muốn sản phẩm cuối cùng được tốt hơn tuy nhiên làm việc nhóm thì cần phải nghe ý kiến từ thành viên của nhóm, cho nên cả hai sẽ chọn ra đâu là phương án tốt nhất cho đồ án mà phù hợp với ý kiến của cả hai người.

### Tự đánh giá kết quả

* Hoàn thành đồ án sớm hơn dự kiến, tuy nhiên có một vài bước nhỏ trong quá trình thực hiện đồ án lâu hơn kế hoạch đề ra.
* Chương trình chỉ chạy được một vài vị trí nhất định mà không chạy được tất cả các ô trong bàn cờ.
* Giao diện thân thiện với người dùng, dễ dàng sử dụng.
* Nhóm tác giả tự đánh giá bài làm của nhóm là 8/10.

## Code toàn bộ chương trình “Mã đi tuần” trên C++

|  |
| --- |
| **// MADITUAN.cpp**  **#include"stdafx.h"**  **#include <iostream>**  **#include <stdio.h>**  **#include <conio.h>**  **#include <ctime>**  **using namespace std;**  **//Các biến toàn cục được sử dụng**  **struct vitri**  **{**  **int x, y, num;**  **};**  **int Banco[20][20];**  **int x, y, xx, yy, sobuocdichuyen, kieudichuyen, N =8;**  **int X[8] = { 1,2,2,1,-1,-2,-2,-1 };**  **int Y[8] = { 2,1,-1,-2,-2,-1,1,2 };**  **vitri a[100];**  **int top;**  **int num = 0;**  **void Inbanco(int N)**  **{**  **cout << endl << endl << endl;**  **for (int i = 2; i <10; i++)**  **{**  **for (int j = 2; j < 10; j++)**  **{**  **cout << "\t" << Banco[i][j];**  **}**  **cout << endl << endl;**  **}**  **}**  **void Khoitaobanco()**  **{**  **int i; int j;**  **// In 2 cột trái phải**  **for (i = 0; i < 12; i++)**  **for (j = 0; j < 2; j++)**  **{**  **Banco[i][j] = -1;**  **}**  **for (i = 0; i < 12; i++)**  **for (j = 10; j < 12; j++)**  **{**  **Banco[i][j] = -1;**  **}**  **// In 2 dòng trên dưới**  **for (i = 0; i < 2; i++)**  **for (j = 2; j < 10; j++)**  **{**  **Banco[i][j] = -1;**  **}**  **for (i = 10; i < 12; i++)**  **for (j = 2; j < 10; j++)**  **{**  **Banco[i][j] = -1;**  **}**  **}**  **int main()**  **{**  **double time;**  **cout << "Kich thuoc ban co NxN: 8x8" <<endl;**  **int STMAX = N\*N;**  **Khoitaobanco();**  **top = -1;**  **int xt; int yt;**  **do**  **{**  **if (x >= 10 || y >= 10)**  **{**  **cout << ">>>> Luu y: Nhap hang va cot luon nho hon " << N << "! <<<<" << endl;**  **cout << "Moi ban nhap lai: " << endl;**  **cout << "Nhap vi tri hang muon bat dau: ";**  **cin >> xt;**  **x = xt + 2;**  **cout << "Nhap vi tri cot muon bat dau: ";**  **cin >> yt;**  **y = yt + 2;**  **}**  **else**  **{**  **cout << "Nhap vi tri hang muon bat dau: ";**  **cin >> xt;**  **x = xt + 2;**  **cout << "Nhap vi tri cot muon bat dau: ";**  **cin >> yt;**  **y = yt + 2;**  **}**  **} while (x >= 10 || y >= 10);**  **Banco[x][y] = 1; //Điểm đầu tiên gán bằng 1**  **sobuocdichuyen = 1; //Số bước di chuyển**  **unsigned long t = clock();**  **while (sobuocdichuyen < N\*N)**  **{**  **for (int num = 0; num <=8; num++)**  **{**  **xx = x + X[num];**  **yy = y + Y[num];**  **if ((num < 8) && (Banco[xx][yy] == 0))**  **{**  **sobuocdichuyen++;**  **top++;**  **x = xx;**  **y = yy;**  **a[top].num = num;**  **a[top].x = x;**  **a[top].y = y;**  **break;**  **}**  **if (num == 8)**  **{**  **if (top != 0)**  **{**  **Banco[x][y] = 0;**  **sobuocdichuyen--;**  **top--;**  **x = a[top].x;**  **y = a[top].y;**  **num = a[top + 1].num;**  **}**  **else**  **{**  **cout << "Khong ton tai nuoc di phu hop!" << endl;**  **exit(-1);**  **}**  **}**  **}**  **Banco[x][y] = sobuocdichuyen;**  **}**  **Inbanco(N);**  **time = ((double)(clock() - t)) / CLOCKS\_PER\_SEC;**  **cout << endl << endl << "\t\t\t Thoi gian chay = " << time << " giay. " << endl;**  **\_getch();**  **return 0;**  **}** |

# LỜI KẾT

“**Mã đi tuần**” là một ví dụ điển hình về việc kết hợp giữa **Giải thuật và Cấu trúc dữ liệu** để giải quyết bài toán.

Phương pháp **Đệ quy** cho phép ta tìm được đường đi của quân mã chỉ bằng vài dòng lệnh, nhưng bên cạnh đó độ phức tạp của đệ quy là khá lớn. Nên đối với những cấu trúc có kích cỡ lớn, đồ sộ thì việc sử dụng Đệ quy tỏ ra chậm và chiếm dụng bộ nhớ. Việc kết hợp đệ quy với những thuật toán thông minh để giải quyết nhanh một bài toán là một bước tiến trong kỹ thuật lập trình.

Thuật toán **Quay lui dùng Stack** được biết đến như là phương pháp tốt hơn của “**Mã đi tuần**”. Giải thuật chọn ra, quay lui với tốc độ nhanh hơn vì có sử dụng ***Stack***. Điều đó hướng đến việc tìm ra kết quả nhanh nhất, ít chiếm dụng bộ nhớ nhất và tất nhiên, độ phức tạp là ít hơn.

Qua quá trình nghiên cứu, thực hiện để hoàn thành đề tài này, nhóm tác giả đã thấy được tầm quan trọng, những ưu điểm mà Giải thuật mang lại. Nếu nói: Cấu trúc dữ liệu là một con đường, thì Giải thuật có thể được ví như là phương tiện giao thông giúp ta di chuyển nhanh trên con đường ấy. Nhóm tác giả sẽ cố gắng để ngày càng hoàn thiện khả năng tư duy cũng như kỹ năng lập trình để ngày càng tiến bộ hơn trên lĩnh vực Công Nghệ thông tin.

*Một lần nữa, nhóm tác giả xin cảm ơn quý thầy cô đã giúp đỡ nhóm hoàn thành đề tài này.*

# 

# TÀI LIỆU THAM KHẢO

---------------------------\*\*\*\*\*\*\*\*\*\*-------------------------

1. [https://vi.wikipedia.org/wiki/Bài\_toán\_mã\_đi\_tuần](https://vi.wikipedia.org/wiki/B%C3%A0i_to%C3%A1n_m%C3%A3_%C4%91i_tu%E1%BA%A7n) **- Wikipedia.**
2. <https://vi.wikipedia.org/wiki/Ng%C4%83n_x%E1%BA%BFp> - **Wikipedia**
3. **PDF – Giáo Trình Cấu Trúc Dữ Liệu – Phan Chí Tùng – ĐHBK Đà Nẵng.**
4. **Phân tích Thiết kế Thuật Toán – Nguyễn Văn Linh – Đại học Cần Thơ – 2003.**
5. **Cấu trúc dữ liệu và giải thuật – Lê Văn Vinh – ĐH Sư Phạm Kỹ Thuật TP.HCM**