**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO**

**ĐỀ TÀI:**

**BÀI TOÁN MÃ ĐI TUẦN**

**GVHD**: TS.Văn Thiên Hoàng

**Môn**: Cấu trúc dữ liệu và giải thuật

**SVTH**: Phạm Minh Hiếu-2011063873-20DTHB4

Phan Thanh Tùng-2011062425-20DTHB4

Thành phố Hồ Chí Minh, tháng 11 năm 2021

**MỤC LỤC**

[**MỤC LỤC** ii](#_Toc86572304)

[**LỜI NÓI ĐẦU** iii](#_Toc86572305)

[**I.** **GIỚI THIỆU BÀI TOÁN MÃ ĐI TUẦN** 1](#_Toc86572306)

[**II.** **MÔ TẢ BÀI TOÁN** 3](#_Toc86572307)

[**1.** **Xác định thông tin vào (Input):** 3](#_Toc86572308)

[**2.** **Xác định thông tin ra (Output):** 3](#_Toc86572309)

[**3.** **Xác định vị trí di chuyển từ 2 đến (N x N) sao cho quân mã:** 3](#_Toc86572310)

[**III.** **GIỚI THIỆU Ý TƯỞNG GIẢI QUYẾT** 3](#_Toc86572311)

[**IV.** **XÂY DỰNG CẤU TRÚC DỮ LIỆU CHO BÀI TOÁN** 5](#_Toc86572312)

[**V.** **GIẢI PHÁP – THUẬT TOÁN (ĐỆ QUY QUAY LUI)** 7](#_Toc86572313)

[**VI.** **ĐỘ PHỨC TẠP CỦA THUẬT TOÁN** 7](#_Toc86572314)

[**VII.** **LẬP TRÌNH VÀ THỬ NGHIỆM** 8](#_Toc86572315)

[**1.** **Lập trình** 8](#_Toc86572316)

[**2.** **Kết quả thử nghiệm** 8](#_Toc86572317)

[**VIII.** **KẾT LUẬN** 10](#_Toc86572318)

[**TÀI LIỆU THAM KHẢO** 11](#_Toc86572319)

**LỜI NÓI ĐẦU**

***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 bức 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 **cấu trúc dữ liệu và giải thuật** được xem là cơ sở, nền tảng đầu tiên.

**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.

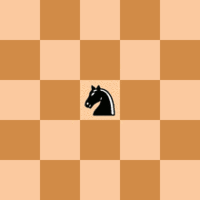
Sau một thời gian học tập và nghiên cứu ngôn ngữ lập trình C, kĩ thuật lập trình và cấu trúc dữ liệu và giải thuật, để nắm bắt những kiến thức học một cách tốt hơn, nhóm chúng em đã thực hiện đề tài: “**Mã đi tuần”** *(Knight’s Tour).*

Trong quá trình thực hiện đề tài, mặc dù đã tìm hiểu và nghiên cứu nhưng chắc chắn khó tránh những thiếu sót, chúng em rất mong nhận được những sự chỉ dẫn, ý kiến đóng góp của quý thầy cô để chúng em ngày càng hoàn thiện hơn. Đồng thời chúng em xin gửi lời cảm ơn đến thầy **Văn Thiên Hoàng** đã giúp nhóm em hoàn thành đề tài này.

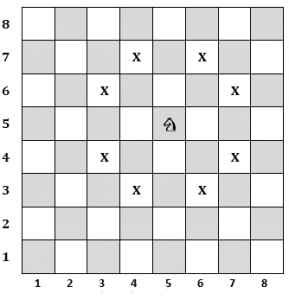
**Nhóm sinh viên thực hiện**

1. **GIỚI THIỆU BÀI TOÁN MÃ ĐI TUẦN**

* Bài toán mã đi tuần *(Knight’s Tour)*
  + Mã đi tuần (hay hành trình của quân mã) là bài toán về việc di chuyển một quân mã trên bàn cờ vua (8 x 8). Quân mã được đặt ở một ô trên một bàn cờ trống nó phải di chuyển theo quy tắc của cờ vua để đi qua mỗi ô trên bàn cờ đúng một lần.
  + Nếu một quân mã đi hết 64 vị trí và tại vị trí cuối cùng có thể di chuyển đến vị trí bắt đầu thông qua một nước cờ thì đó gọi là một hành trình đóng
  + Có những hành trình, trong đó quân mã sau khi đi hết tất cả 64 ô của bàn cờ và từ ô cuối của hành trình không thể đi về ô xuất phát chỉ bằng một nước đi. Những hành trình như vậy được gọi là hành trình mở.
  + Ví dụ sau đây là một lời giải cho bài toán mã đi tuần trên bàn cờ (5 × 5) (hành trình mở).



* Cách di chuyển của một quân mã
  + Nước đi của một quân mã giống hình chữ L và nó có thể di chuyển tất cả các hướng. Ở một vị trí thích hợp thì quân mã có thể di chuyển đến được 8 vị trí.



1. **MÔ TẢ BÀI TOÁN**
2. **Xác định thông tin vào (Input):**

**-** Một bảng vuông (bàn cờ) kích thước (N x N).

**-** Tọa độ xuất phát của quân mã N(sx, sy).

1. **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 x N) (vị trí kết thúc) của quân mã.

**-** Từ vị trí k sang vị trí thứ k + 1 phải theo đúng luật di chuyển quân mã trong cờ vua.

1. **Xác định vị trí di chuyển từ 2 đến (N x N) sao cho quân mã:**

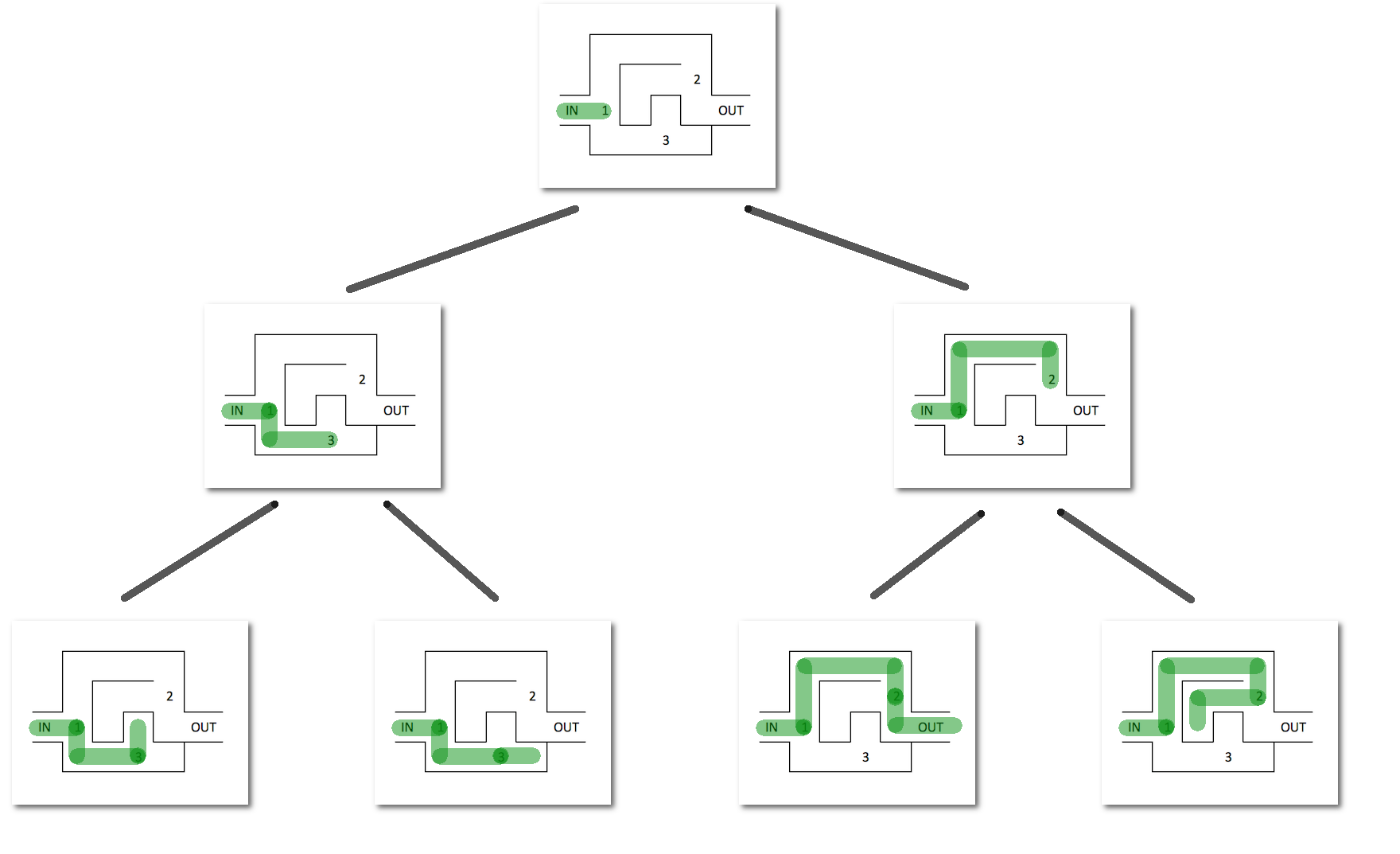
**-** Di chuyển đúng luật.

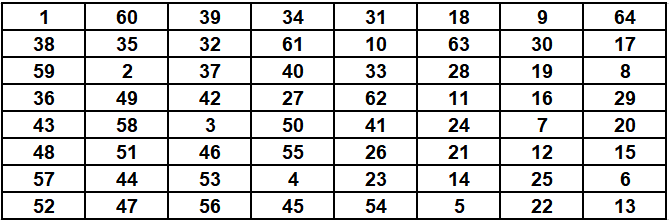
**-** Mỗi ô chỉ được đi qua 1 lần.

**-** Vị trí cuối cùng (N x N) có thể nhảy tới vị trí xuất phát (1).

1. **GIỚI THIỆU Ý TƯỞNG GIẢI QUYẾT**

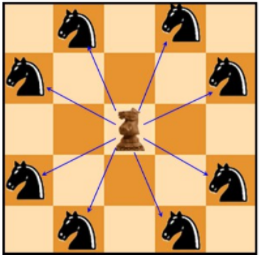
* Bài toán đặt ra yêu cầu là tìm đường cho con Mã có thể đặt chân đến tất cả các điểm trên bàn cờ với mỗi vị trí (hay từng ô) con Mã chỉ được đi đến duy nhất một lần.
* Với bài toán này nhóm em sử dụng thuật toán quay lui (Backtracking). Ý tưởng khá đơn giản, khi đứng trước 2 con đường, ta sẽ chọn 1 đường để đi, nếu đường đó là đường cụt, thì ta quay lại vị trí ban đầu và đi đường thứ 2.
* Sau đây là hình ảnh miêu tả ý tưởng của thuật toán:



* Thuật toán sẽ thử hết tất cả những khả năng có thể xảy ra cho đến khi mà nó đạt được mục đích của bài toán.
* Quá trình này sẽ diễn ra liên tục cho đến khi tất cả các điểm trong bàn cờ được con Mã đi đến.
* Với ý tưởng như vậy bài toán sẽ được giải quyết như sau:
* Cho quân Mã đi hết đến tất cả các vị trí có thể đi trên trên bàn cờ.
* Nếu con Mã đi vào vị trí mà nó không thể đi tiếp (tức là những điểm nó có thể đi từ điểm hiện tại đã đặt chân đến rồi) thì quay lại vị trí trước đó và lựa chọn con đường khác để di chuyển.
* Việc này lặp đi lặp lại sẽ diễn ra liên tục cho đến khi tất cả các điểm trên bàn cờ được quân Mã đi tới.
* Ví dụ với bàn cờ (8x8):
* Đây là thứ tự các bước đi của con Mã, nó sẽ nhảy từ 1 🡪 2 🡪 3 cho đến khi nào quân Mã nhảy đến hết tất cả các vị trí trên bàn cờ.

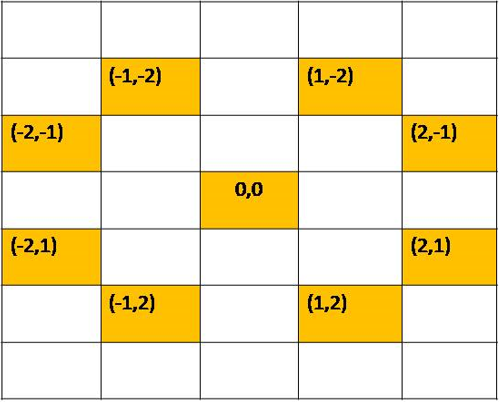
1. **XÂY DỰNG CẤU TRÚC DỮ LIỆU CHO BÀI TOÁN**

* Biểu diễn bàn cờ bằng một ma trận sol[N][N].
* Giá trị của mỗi sol[x][y] (0 <= x, y < N) trên bàn cờ như sau:
* sol[x][y] = -1: Ô (x, y) chưa được quân mã đi qua.
* sol[x][y] = k: Ô(x, y) được quân Mã đi đến nước thứ k (1 <= k <= (N x N)).
* Cấu trúc dữ liệu cho mỗi bước đi của quân Mã:

Nước đi của con Mã được phân tích như sau:

* Từ 1 vị trí (x, y) trên bàn cờ có tối đa 8 vị trí khác nhau (next\_x, next\_y) cho quân Mã nhảy tới. Việc thu được vị trí (next\_x, next\_y) từ (x, y) có thể giải quyết bằng phương pháp cộng thêm độ chênh lệch về tọa độ chứa sẵn trong 2 mảng phụ sau:

int xMove[8] = {2, 1, -1, -2, -2, -1, 1, 2};

 int yMove[8] = {1, 2, 2, 1, -1, -2, -2, -1};

* Vị trí (next\_x, next\_y) cần tìm bị ràng buộc bởi 2 điều kiện:

***Thứ* *1***, (next\_x, next\_y) phải còn nằm trong bàn cờ, tức là (0<=next\_x < N) và (0<=next\_y < N).

***Thứ 2***, sol[next\_x][ next\_y] phải bằng -1, tức là vị trí này chưa được quân Mã đi qua.

1. **GIẢI PHÁP – THUẬT TOÁN (ĐỆ QUY QUAY LUI)**

* Thuật toán đệ quy quay lui như sau:
* Bắt đầu từ vị trí xuất phát (1).
* Từ vị trí (k) ta đi tìm vị trí thứ (k+1).
* Sử dụng 2 mảng phụ đề 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 x N) và hành trình đóng thì có được kết quả.
* Nếu chưa là vị trí thứ (N x 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 thì quay lại, tìm theo hướng (k + 1) khác.

1. **ĐỘ PHỨC TẠP CỦA THUẬT TOÁN**

* Nhìn tổng thể thì chương trình này chỉ có hai lệnh ở dòng (1) và dòng (3) vậy nên ta sẽ xét các dòng trong bài để xác định độ phức tạp của thuật toán:
* Với vòng lặp ta sẽ cụ thể ở dưới ta sẽ có i < 8, vậy i = 7 với giá trị khởi tạo cho i = 0 thì vòng lặp này sẽ lặp lại 7 – 0 + 1 = 8 lần vì dao động của nó sẽ khoảng từ 1 ~ 8 lần. Vậy độ phức tạp sẽ là **O(8)**.
* Tiếp theo ta có 1 hàm đệ quy quan trong nhất của thuật toán , ta sẽ phụ thuộc vào giá trị bị thay đổi sau mỗi lần đệ quy đó là k ở dòng (8), ta thấy , nếu k+1 cho đến khi mà k bằng giới hạn của bàn cờ với trường hợp cụ thể là 5\*5 = 25 thì k tối đa sẽ là k = 25 sẽ dừng hẳn chương trình, trả về giá trị là true còn nếu trong trường hợp mà k+1 với đệ quy lên 25 lần ở worst case thì ta sẽ có là k lần dao động khoảng từ 1 ~ 25 lần. Vậy nên độ phức tạp ở đây mà nhóm em đưa ra đó là **O(n)** với T(n) = T(n+1) + 1 khi k > 1 .
* Khi giá trị của k chưa bằng 25 mà đã không còn đường để đi trong 8 cách trên thì nó buộc phải lùi lại tương tự như đệ quy ở trên, Độ phức tạp sẽ là **O(n)**.
* Số lần lặp lại tối đa của vòng for là 8 lần với Độ phức tạp của 2 lần đệ quy ở worstcase và Do đệ quy ở trong vòng lặp nên ta sẽ sử dụng phép nhân để tính độ phức tạp.
* Độ Phức tạp: O(1) + = .

int BackTrack(int x, int y, int k, int sol[N][N], int xMove[N], int yMove[N])

{

    int i, next\_x, next\_y;

(1)    if (k == N \* N)

(2)        return true;

(3)    for (i = 0; i < 8; i++)

    {

(4)        next\_x = x + xMove[i];

(5)        next\_y = y + yMove[i];

(6)        if (isSafe(next\_x, next\_y, sol))

        {

(7)            sol[next\_x][next\_y] = k + 1;

(8)            if (BackTrack(next\_x, next\_y, k + 1, sol, xMove, yMove) == true)

(9)                return true;

(10)            else

(11)                sol[next\_x][next\_y] = -1;

        }

    }

    return false;

}

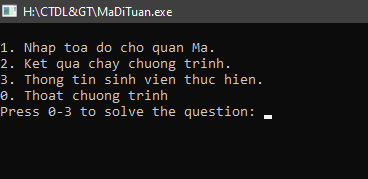
1. **LẬP TRÌNH VÀ THỬ NGHIỆM**
2. **Lập trình**

Chương trình bài toán **“*Mã đi tuần”*** được viết bằng ngôn ngữ lập trình C.

Code và toàn bộ chương trình được đính kèm theo.

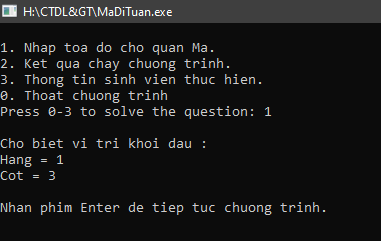
1. **Kết quả thử nghiệm**

Chạy file MaDiTuan.exe ta được kết quả như sau:

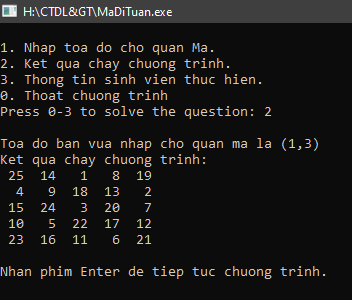


Chức năng của **MENU**, bao gồm

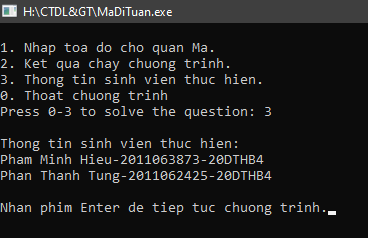
* **Chọn (1)** để nhập tọa độ cho quân Mã.



* **Chọn (2)** để xem kết quả chạy của chương trình.



* **Chọn (3)** để xem thông tin sinh viên thực hiện.



* **Chọn (0)** để thoát chương trình.

1. **KẾT LUẬN**

* ***“Mã đi tuần”*** là một ví dụ điển hình kết hợp giữa giải thuật và cấu trúc dữ liệu để giải quyết bài toán.
* Phương pháp đệ quy quay lui 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 khá là 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.
* Bài báo cáo của nhóm em đã cố gắng tìm hiểu và tổng hợp những nội dung liên quan đến đề tài ***“Mã đi tuần”*** mà nhóm đã thực hiện. Tuy nhiên, nhóm em rất mong nhận được sự đóng góp ý kiến của quý thầy để bài báo cáo càng hoàn thiện hơn.

Xin chân thành cảm ơn!

**TÀI LIỆU THAM KHẢO**

1. [The Knight's tour problem | Backtracking-1 - GeeksforGeeks](https://www.geeksforgeeks.org/the-knights-tour-problem-backtracking-1/?ref=lbp)
2. [learn-machine-learning-in-two-months/backtracking.MD at master · bangoc123/learn-machine-learning-in-two-months (github.com)](https://github.com/bangoc123/learn-machine-learning-in-two-months/blob/master/algorithms/graph/backtracking/backtracking.MD)
3. [Knight's tour problem using backtracking and its analysis (codesdope.com)](https://www.codesdope.com/course/algorithms-knights-tour-problem/)
4. [Bài toán Mã đi tuần - Hướng dẫn giải bài toán Mã đi tuần (nguyenvanhieu.vn)](https://nguyenvanhieu.vn/bai-toan-ma-di-tuan/)
5. [Đồ Án MÃ ĐI TUẦN ( KNIGHT’s TOUR ) (123docz.net)](https://123docz.net/document/2637086-do-an-ma-di-tuan-knight-s-tour.htm)