KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH**

**HỌC KỲ I, NĂM HỌC 2024-2025**

**VIẾT CHƯƠNG TRÌNH MÔ PHỎNG GIẢI THUẬT**

**TÔ MÀU ĐỒ THỊ**

*Giảng viên hướng dẫn: Sinh viên thực hiện:*

ThS. Trầm Hoàng Nam Họ tên: Trương Nguyễn Tố Nguyên

MSSV: 110122127

Lớp: DA22TTB

***Trà Vinh, tháng 12 năm 2024***

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày … tháng … năm … …*

**Giảng viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày … tháng … năm … …*

**Giảng viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày … tháng … năm … …*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày … tháng … năm … …*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành đến thầy Trầm Hoàng Nam đã hướng dẫn tận tình trong quá trình thực hiện đồ án. Thầy đã cho em những gợi ý và cách phát triển đồ án, giúp em giải quyết những khó khăn gặp phải, để em có thể hoàn thành đồ án một cách hiệu quả.

**MỤC LỤC**

[MỞ ĐẦU 12](#_Toc186375992)

[CHƯƠNG 1: TỔNG QUAN 13](#_Toc186375993)

[1.1 Hướng nghiên cứu 13](#_Toc186375994)

[1.1.1 Cài đặt và so sánh giải thuật 13](#_Toc186375995)

[1.1.2 Xây dựng giao diện trực quan 13](#_Toc186375996)

[1.1.3 Đánh giá kết quả 13](#_Toc186375997)

[CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT 14](#_Toc186375998)

[2.1 Cơ sở lý thuyết 14](#_Toc186375999)

[2.1.1 Đồ thị 14](#_Toc186376000)

[2.1.2 Bậc của đỉnh trong đồ thị 15](#_Toc186376001)

[2.1.3 Bài toán tô màu đồ thị 16](#_Toc186376002)

[2.1.4 Một số định lý về tô màu đồ thị 16](#_Toc186376003)

[2.2 Giả thiết khoa học 16](#_Toc186376004)

[2.3 Phương pháp nghiên cứu 17](#_Toc186376005)

[2.3.1 Nghiên cứu một số thuật toán cho bài toán tô màu đồ thị 17](#_Toc186376006)

[2.3.2 Phân tích thuật toán 17](#_Toc186376007)

[CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU 19](#_Toc186376008)

[3.1 Cài đặt chương trình 19](#_Toc186376009)

[3.1.1 Khai báo các thư viện cần thiết 19](#_Toc186376010)

[3.1.2 Khởi tạo lớp Dothi 19](#_Toc186376011)

[3.1.3 Lớp Tomau 23](#_Toc186376012)

[3.2 Kiểm thử 33](#_Toc186376013)

[3.2.1 Ví dụ 1 33](#_Toc186376014)

[3.2.2 Ví dụ 2 36](#_Toc186376015)

[3.2.3 Ví dụ 3 39](#_Toc186376016)

[CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU 43](#_Toc186376017)

[4.1 Giao diện người dùng 43](#_Toc186376018)

[4.1.1 Giao diện chính 43](#_Toc186376019)

[4.1.2 Cửa sổ nhập đỉnh và cạnh thủ công 43](#_Toc186376020)

[4.1.3 Cửa sổ nhập ma trận kề 44](#_Toc186376021)

[4.2 Cách sử dụng 44](#_Toc186376022)

[4.2.1 Thêm đỉnh và cạnh thủ công 44](#_Toc186376023)

[4.2.2 Nhập ma trận kề 45](#_Toc186376024)

[4.2.3 Vẽ đồ thị 47](#_Toc186376025)

[4.2.4 Tô màu đồ thị 48](#_Toc186376026)

[4.3 Hiệu năng 50](#_Toc186376027)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 51](#_Toc186376028)

[DANH MỤC TÀI LIỆU THAM KHẢO 52](#_Toc186376029)

**DANH MỤC HÌNH ẢNH**

[Hình 1. Ví dụ bậc của đồ thị vô hướng 15](#_Toc186302648)

[Hình 2. Ví dụ bậc của đồ thị có hướng 15](#_Toc186302649)

[Hình 3. Ví dụ thuật toán Welch-Powell 18](#_Toc186302650)

[Hình 4. Ví dụ thuật toán Backtracking 18](#_Toc186302651)

[Hình 5. Khai báo thư viện 19](#_Toc186302652)

[Hình 6. Khởi tạo lớp Dothi 19](#_Toc186302653)

[Hình 7. Phương thức thêm đỉnh 20](#_Toc186302654)

[Hình 8. Phương thức thêm cạnh 20](#_Toc186302655)

[Hình 9. Phương thức tạo đồ thị dựa trên ma trận kề 20](#_Toc186302656)

[Hình 10. Phương thức thực hiện giải thuật Welch-Powell 21](#_Toc186302657)

[Hình 11. Phương thức thực hiện giải thuật Backtracking 22](#_Toc186302658)

[Hình 12. Lớp Tomau 23](#_Toc186302659)

[Hình 13. Khởi tạo một số đối tượng 23](#_Toc186302660)

[Hình 14. Xây dựng giao diện nhập liệu 24](#_Toc186302661)

[Hình 15. Phương thức nhập thủ công 25](#_Toc186302662)

[Hình 16. Phương thức tạo đỉnh thủ công 25](#_Toc186302663)

[Hình 17. Phương thức tạo cạnh thủ công 26](#_Toc186302664)

[Hình 18. Phương thức nhập ma trận kề 26](#_Toc186302665)

[Hình 19. Phương thức tạo đồ thị từ ma trận 27](#_Toc186302666)

[Hình 20. Phương thức tạo tọa độ các đỉnh từ ma trận kề 28](#_Toc186302667)

[Hình 21. Phương thức tạo các cạnh từ ma trận kề 28](#_Toc186302668)

[Hình 22. Phương thức vẽ đồ thị 29](#_Toc186302669)

[Hình 23. Phương thức tô màu bằng giải thuật Welch-Powell 30](#_Toc186302670)

[Hình 24. Phương thức tô màu bằng giải thuật Backtracking 31](#_Toc186302671)

[Hình 25. Phương thức vẽ đồ thị khi thực hiện tô màu 32](#_Toc186302672)

[Hình 26. Chạy chương trình 32](#_Toc186302673)

[Hình 27. Ma trận ví dụ 1 33](#_Toc186302674)

[Hình 28. Đồ thị ví dụ 1 34](#_Toc186302675)

[Hình 29. Kết quả tô màu Welch-Powell ví dụ 1 35](#_Toc186302676)

[Hình 30. Kết quả tô màu Backtracking ví dụ 1 36](#_Toc186302677)

[Hình 31. Ma trận ví dụ 2 37](#_Toc186302678)

[Hình 32. Đồ thị ví dụ 2 37](#_Toc186302679)

[Hình 33. Kết quả tô màu Welch-Powell ví dụ 2 38](#_Toc186302680)

[Hình 34. Kết quả tô màu Backtracking ví dụ 2 39](#_Toc186302681)

[Hình 35. Ma trận ví dụ 3 40](#_Toc186302682)

[Hình 36. Đồ thị ví dụ 3 40](#_Toc186302683)

[Hình 37. Kết quả tô màu Welch-Powell ví dụ 3 41](#_Toc186302684)

[Hình 38. Kết quả tô màu Backtracking ví dụ 3 42](#_Toc186302685)

[Hình 39. Giao diện chính 43](#_Toc186302686)

[Hình 40. Cửa sổ nhập đỉnh và cạnh thủ công 43](#_Toc186302687)

[Hình 41. Cửa sổ nhập ma trận kề 44](#_Toc186302688)

[Hình 42. Thông báo thêm đỉnh thành công 44](#_Toc186302689)

[Hình 43. Thông báo thêm cạnh thành công 45](#_Toc186302690)

[Hình 44. Thông báo thêm cạnh thất bại 45](#_Toc186302691)

[Hình 45. Thông báo nhập ma trận kề thành công 46](#_Toc186302692)

[Hình 46. Thông báo nhập sai kiểu ma trận kề 46](#_Toc186302693)

[Hình 47. Thông báo các lỗi khác khi nhập ma trận kề 46](#_Toc186302694)

[Hình 48. Yêu cầu nhập đồ thị 47](#_Toc186302695)

[Hình 49. Ví dụ đồ thị 48](#_Toc186302696)

[Hình 50. Ví dụ tô màu Welch-Powell 49](#_Toc186302697)

[Hình 51. Ví dụ tô màu Backtracking 50](#_Toc186302698)

**DANH MỤC BẢNG BIỂU**

**TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH**

Tô màu đồ thị là một bài toán trong lý thuyết đồ thị, yêu cầu của bài toán là gán màu cho các đỉnh sao cho hai đỉnh kề nhau khác màu. Bài toán này có nhiều ứng dụng trong thực tế như lập lịch thi, tô màu bản đồ và thiết kế mạng.

Khi thực hiện bài toán, có một số vấn đề phải quan tâm:

1. Làm sao để tối ưu số màu.
2. Nếu gặp đồ thị lớn, phải dùng giải thuật nào cho hiệu quả?
3. Trực quan hóa bài toán để dễ tiếp thu.

Các hướng tiếp cận:

1. Giải thuật Welch-Powell: Gán màu theo thứ tự các đỉnh, sử dụng màu mà các đỉnh kề chưa dùng tới.
2. Giải thuật Backtracking: Thử các màu trên từng đỉnh, nếu màu không hợp lệ thì lùi về đỉnh trước và thay đổi màu, nếu không tìm được cách tối ưu giải thuật này vẫn trả về kết quả.

Cách giải quyết vấn đề:

1. Xây dựng chương trình bao gồm:

* Giao diện người dùng: cho phép thực hiện các thao tác thêm đỉnh, thêm cạnh, nhập ma trận kề, vẽ đồ thị, tô màu đồ thị và quan sát kết quả thông qua nhấp chuột và sử dụng bàn phím.
* Hai thuật toán cho phép so sánh hiệu năng bằng kết quả hiển thị: sắc số và thời gian thực hiện.

1. Công cụ lập trình:

* Sử dụng Python kết hợp với thư viện Tkinter.

Kết quả đạt được:

1. Chương trình mô phỏng:

* Hoạt động ổn định, giao diện người dùng thân thiện cho phép nhập liệu và hiển thị kết quả rõ ràng.
* Tô màu thành công bằng giải thuật Welch-Powell hoặc Backtracking.

1. Hiệu quả giải thuật:

* Giải thuật Welch-Powell hoạt động nhanh, nhưng có thể cho kết quả không tối ưu.
* Giải thuật Backtracking cho số màu tối ưu nhưng sẽ mất nhiều thời gian với các đồ thị lớn.

# MỞ ĐẦU

Lý do chọn đề tài: Tô màu đồ thị là một bài toán quan trọng trong lý thuyết đồ thị, đồng thời cũng có nhiều ứng dụng trong thực tế. Em đã đọc được một số đề tài có liên quan đến bài toán này, mà trong đó, chưa có ai tích hợp thêm giao diện đồ họa để thể hiện trực quan, em sẽ cố gắng thực hiện phần này. Sau khi nghiên cứu xong đề tài, những kiến thức nắm giữ được sẽ có hỗ trợ rất lớn cho các dự án sau này.

1. Mục đích nghiên cứu:

* Xây dựng giao diện người dùng thân thiện: Xây dựng một giao diện dễ sử dụng, giúp người dùng dễ dàng quan sát kết quả thực hiện bài toán tô màu đồ thị.
* Cài đặt giải thuật: Cài đặt khoảng hai giải thuật tô màu, đếm sắc số và tính thời gian thực hiện nhằm so sánh hiệu năng giữa chúng.

2. Đối tượng nghiên cứu:

* Đối tượng lý thuyết: Nghiên cứu một số thuật toán tìm được trong các tài liệu.
* Đối tượng thử nghiệm: Thử nghiệm trên các đồ thị ngẫu nhiên, kích thước nhỏ để dễ kiểm tra tính chính xác.

3. Phạm vi nghiên cứu:

* Phạm vi thuật toán: Hai thuật toán Welch-Powell và Backtracking.
* Phạm vi đồ thị: Đồ thị vô hướng, không trọng số với kích thước nhỏ.

# TỔNG QUAN

## Hướng nghiên cứu

### Cài đặt và so sánh giải thuật

Trong nghiên cứu này, sẽ tập trung vào cài đặt và so sánh hai giải thuật:

* Giải thuật Welch-Powell: Có ưu điểm là dễ tiếp cận.
* Giải thuật Backtracking: Cho kết quả tối ưu hơn.

### Xây dựng giao diện trực quan

Cho phép người dùng nhập đồ thị thử nghiệm, quan sát kết quả.

Trong đó, kết quả chương trình trả về gồm:

* Sắc số.
* Thời gian thực hiện.

### Đánh giá kết quả

Đánh giá hiệu quả của hai giải thuật dựa trên các tiêu chí:

* Sắc số.
* Thời gian thực hiện.

# NGHIÊN CỨU LÝ THUYẾT

## Cơ sở lý thuyết

### Đồ thị

Đồ thị là một cấu trúc rời rạc bao gồm các đỉnh và các cạnh nối các đỉnh này. Chúng ta phân biệt các loại đồ thị khác nhau bởi kiểu và số lượng cạnh nối hai đỉnh nào đó của đồ thị. [1]

#### Định nghĩa 1

Đồ thị vô hướng G = (V, E) bao gồm V là tập các đỉnh, và E là tập các cặp không có

thứ tự gồm hai phần tử khác nhau của V gọi là các cạnh. [1]

#### Định nghĩa 2

* Hai đỉnh được gọi là liền kề nhau nếu có cạnh nối hai đỉnh đó với nhau. Cạnh

nối 2 đỉnh lại với nhau được gọi là cạnh liên thuộc. [1]

* Hai cạnh được gọi là kề nhau nếu giữa chúng có đỉnh chung. [1]
* Nếu e=(v, v) là một cạnh của đồ thị thì e được gọi là một khuyên. [1]

#### Định nghĩa 3

* Nếu mỗi cạnh e = (u, v) là không phân biệt thứ tự của các đỉnh u và v (từ u tới v

không kể hướng) thì ta nói đồ thị G=(V, E) là đồ thị vô hướng. [1]

* Nếu mỗi cạnh e = (u, v) có phân biệt thứ tự của các đỉnh u và v (tức là từ u tới v

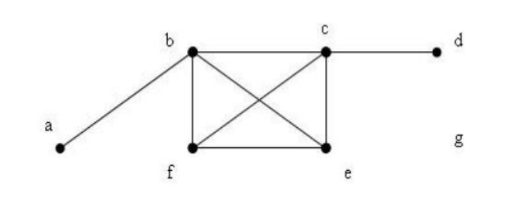
khác từ v tới u) thì ta nói đồ thị G=(V, E) là đồ thị có hướng. Cạnh của đồ thị có

hướng được gọi là cung. [1]

### **Bậc của đỉnh trong đồ thị**

#### Đồ thị vô hướng

Định nghĩa: Đỉnh v của đồ thị G được gọi là có bậc n nếu v kề với n đỉnh khác (v là đầu mút của n cạnh). Ký hiệu: deg(v) hay d(v). [1]



Hình 1. Ví dụ bậc của đồ thị vô hướng

Ví dụ: xét ví dụ cho ở hình 3, ta có:

deg(a)=deg(d)=1, deg(b)=deg(c)=4, deg(e)=deg(f)=3, deg(g)=0

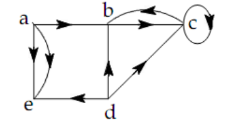
#### Đồ thị có hướng

Định nghĩa: Cho G là đồ thị có hướng

* **Bậc vào** của đỉnh v, ký hiệu: deg-(v) (hoặc din(v)) là số cung có đỉnh cuối là v. [1]
* **Bậc ra** của đỉnh v, ký hiệu: deg+(v) (hoặc dout(v)) là số cung có đỉnh đầu là v. [1]

**Chú** **ý:** Một vòng tại một đỉnh sẽ góp thêm một đơn vị vào bậc vào và một đơn vị

bậc ra của đỉnh này. [1]



Hình 2. Ví dụ bậc của đồ thị có hướng

Ví dụ: xét ví dụ cho ở hình 3, ta có:

din(a)=0, dout(a)=3, din(b)=3, dout(b)=1, din(c)=3, dout(c)=2, din(d)=0, dout(d)=3, din(e)=3, dout(e)=0

### Bài toán tô màu đồ thị

#### Định nghĩa

Tô màu một đơn đồ thị là việc gán màu cho các đỉnh của nó sao cho hai đỉnh liền kề có màu khác nhau. Mỗi đồ thị có thể có nhiều cách tô màu khác nhau. [1]

**Số màu hay sắc số** (Chromatic number) của một đồ thị G là số màu tối thiểu cần thiết để tô màu G. Ký hiệu: ƴ(G). [1]

### Một số định lý về tô màu đồ thị

#### Định lý 1

Mọi đơn đồ thị đầy đủ Kn đều có: ƴ(Kn) = n. [1]

#### Định lý 2

Mọi chu trình độ dài lẻ đều có sắc số là 3. [1]

#### Định lý 3

Nếu G có chứa một đồ thị con đẳng cấu với Kn thì ƴ(G) ≥ n. [1]

#### Định lý 4

Một đơn đồ thị G=(V, E) có thể tô bằng 2 màu khi và chỉ khi nó không có chu trình độ dài lẻ. [1]

#### Định lý 5 (định lý 4 màu) (định lý Appel-Haken, 1976)

Mọi đồ thị phẳng đều có sắc số không lớn hơn 4. [1]

## Giả thiết khoa học

* Đồ thị đầu vào là đơn đồ thị, không chứa khuyên hoặc cạnh song song.
* Mọi đỉnh và cạnh của đồ thị đều được biểu diễn chính xác, không có sai sót hoặc thiếu sót dữ liệu.

## Phương pháp nghiên cứu

### Nghiên cứu một số thuật toán cho bài toán tô màu đồ thị

#### Thuật toán Welch-Powell

Để tô màu một đồ thị G, ta có thể sử dụng thuật toán Welch-Powell như sau:

* Sắp xếp các đỉnh G theo bậc giảm dần. [1]
* Dùng một màu để tô đỉnh đầu tiên và cũng dùng màu này để tô màu các đỉnh

liên tiếp trong danh sách mà không kề với đỉnh đầu tiên. [1]

* Bắt đầu trở lại đầu danh sách, tô màu thứ hai cho đỉnh chưa được tô và lặp lại

quá trình trên cho đến khi tất cả các đỉnh đều được tô màu. [1]

#### Thuật toán Backtracking

Để tô màu một đồ thị G, ta có thể sử dụng thuật toán Backtracking như sau:

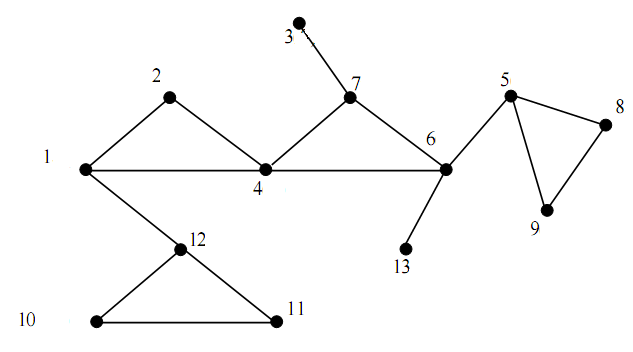
* Thử màu cho từng đỉnh, bắt đầu từ đỉnh đầu tiên.
* Nếu không có màu hợp lệ, quay về đỉnh trước đó để thử một màu khác.
* Kết thúc khi tất cả các đỉnh đều được tô màu.

### Phân tích thuật toán

#### Thuật toán Welch-Powell

Thuật toán chưa cho ta sắc số của đồ thị, kết quả chỉ thu được một cách tô màu tốt hơn tô màu ngẫu nhiên.

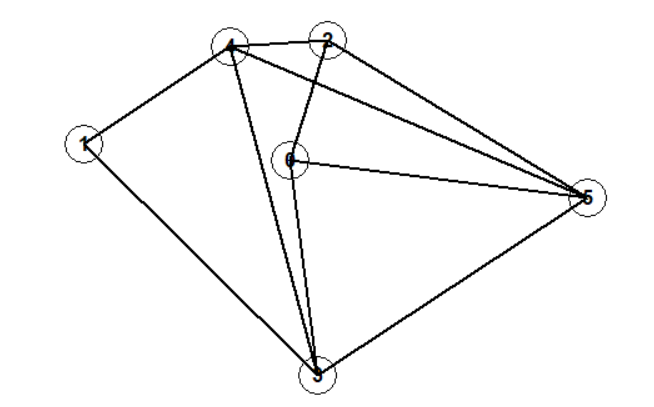
Thuật toán không cho được kết quả tối ưu, ví dụ đồ thị bên dưới:



Hình 3. Ví dụ thuật toán Welch-Powell

#### Thuật toán Backtracking

Thuật toán này có thể mất nhiều thời gian cho việc thử màu từng đỉnh, tuy nhiên khả năng cho kết quả tối ưu cao hơn thuật toán Welch-Powell, ví dụ đồ thị bên dưới:



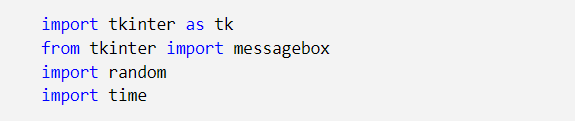
Hình 4. Ví dụ thuật toán Backtracking

Với đồ thị này, thuật toán cho ta sắc số nhưng mất thời gian hơn thuật toán Welch-Powell.

# HIỆN THỰC HÓA NGHIÊN CỨU

## Cài đặt chương trình

### Khai báo các thư viện cần thiết

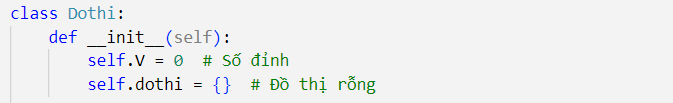
****

Hình 5. Khai báo thư viện

Trong đó:

* Thư viện tkinter là thư viện hỗ trợ tạo giao diện người dùng.
* Messagebox là một thành phần của thư viện tkinter.
* Random là thư viện tạo các giá trị ngẫu nhiên, trong chương trình, thư viện này sẽ hỗ trợ tạo tọa độ ngẫu nhiên cho các đỉnh.
* Time là thư viện về thời gian, trong chương trình sẽ hỗ trợ việc tính thời gian thực hiện giải thuật.

### Khởi tạo lớp Dothi



Hình 6. Khởi tạo lớp Dothi

Trong đó:

* def \_\_init\_\_(self) là phương thức khởi tạo.
* self.V là biến lưu số đỉnh, ban đầu có giá trị là 0.
* self.dothi là một dictionary lưu trữ danh sách kề của các đỉnh.

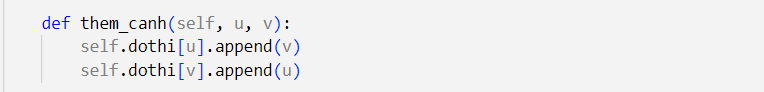
#### Phương thức thêm đỉnh



Hình 7. Phương thức thêm đỉnh

Khi phương thức này được gọi, một đỉnh mới self.V sẽ được thêm vào đồ thị với giá trị là danh sách rỗng (do chưa có đỉnh kề), số lượng đỉnh trong đồ thị tăng một.

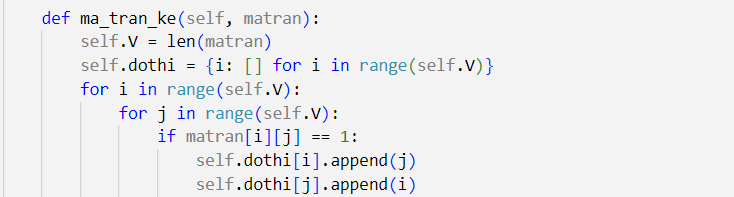
#### Phương thức thêm cạnh



Hình 8. Phương thức thêm cạnh

Khi phương thức này được gọi, đỉnh v sẽ được thêm vào danh sách kề của đỉnh u và ngược lại để đảm bảo đồ thị là vô hướng.

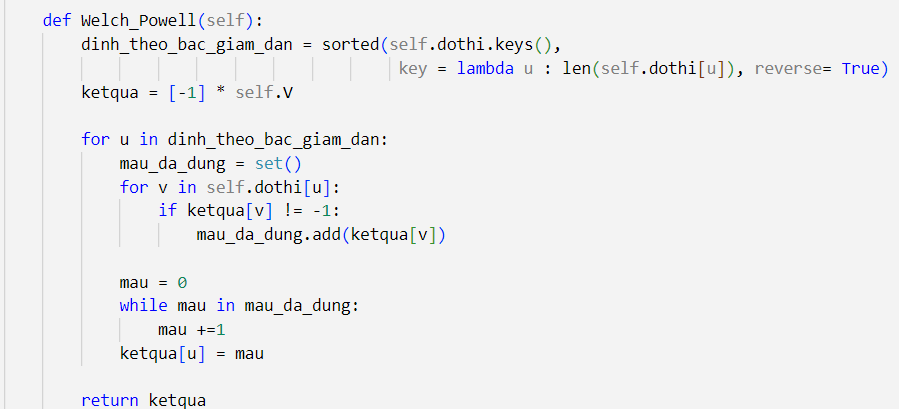
#### Phương thức tạo đồ thị dựa trên ma trận kề



Hình 9. Phương thức tạo đồ thị dựa trên ma trận kề

* Ban đầu, ta gán số đỉnh bằng độ dài ma trận (ma trận vuông có số đỉnh bằng số dòng/cột).
* Khởi tạo danh sách kề rỗng cho tất cả đỉnh từ 0 đến self.V – 1.
* Lặp qua từng hàng và cột, nếu i và j được nối với nhau bởi một cạnh, ta thêm đỉnh i vào danh sách kề của đỉnh j và ngược lại.

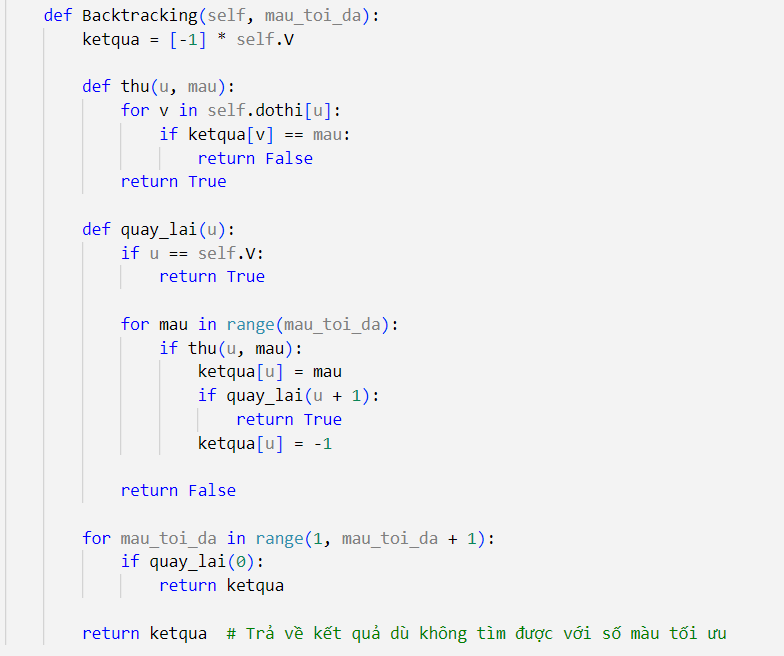
#### Phương thức thực hiện giải thuật Welch-Powell



Hình 10. Phương thức thực hiện giải thuật Welch-Powell

* Sắp xếp các đỉnh theo bậc giảm dần (theo chiều dài danh sách kề của đỉnh u).
* Tạo danh sách chứa màu cho tất cả các đỉnh, giá trị ban đầu là -1 do chưa tô màu.
* Lặp qua từng đỉnh u theo danh sách vừa sắp xếp:
  + Tạo một tập hợp mau\_da\_dung lưu màu đã tô cho các đỉnh kề; lặp qua từng đỉnh kề v, nếu đỉnh v đã được tô màu thì thêm màu này vào danh sách mau\_da\_dung.
  + Bắt đầu từ màu đầu tiên, kiểm tra xem màu đã có trong danh sách mau\_da\_dung chưa, nếu đã có thì kiểm tra màu tiếp theo, cho đến khi tô được màu cho đỉnh.
* Trả kết quả.

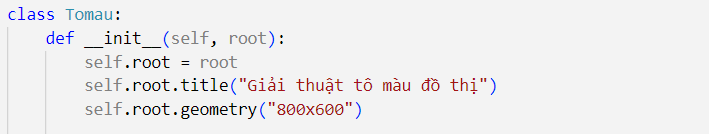
#### Phương thức thực hiện giải thuật Backtracking



Hình 11. Phương thức thực hiện giải thuật Backtracking

* Tạo danh sách lưu màu cho tất cả đỉnh, ban đầu tất cả đều có giá trị -1 (chưa tô màu).
* Tạo phương thức con thử màu cho từng đỉnh u: nếu đỉnh kề của u đã dùng màu này, trả về False, ngược lại trả về True.
* Tạo phương thức con:
  + Nếu tất cả đỉnh u đã được tô màu, trả về True.
  + Lặp qua từng màu trong danh sách mau\_toi\_da, nếu màu này là True ở phương thức trước đó, tô cho đỉnh u.
  + Thử tô màu cho đỉnh kế tiếp, nếu thất bại thì quay về.
* Lặp từ màu 1 đến hết danh sách mau\_toi\_da, nếu tìm được màu hợp lệ, trả về True, ngược lại trả về danh sách kết quả.
* Nếu không tìm được màu tối ưu, vẫn trả về kết quả.

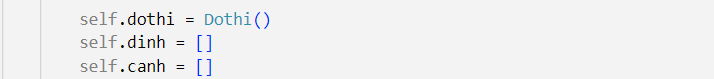
### Lớp Tomau



Hình 12. Lớp Tomau

* def \_\_init\_\_ là phương thức khởi tạo.
* self.root là tham chiếu đến giao diện chính, ta đặt tiêu đề và căn chỉnh kích thước cho giao diện.

#### Khởi tạo một số đối tượng



Hình 13. Khởi tạo một số đối tượng

Trong đó:

* self.dothi là đối tượng của lớp Dothi, dùng để gọi các phương thức của lớp này.
* self.dinh là danh sách lưu tọa độ cho các đỉnh.
* self.canh là danh sách lưu các cạnh.

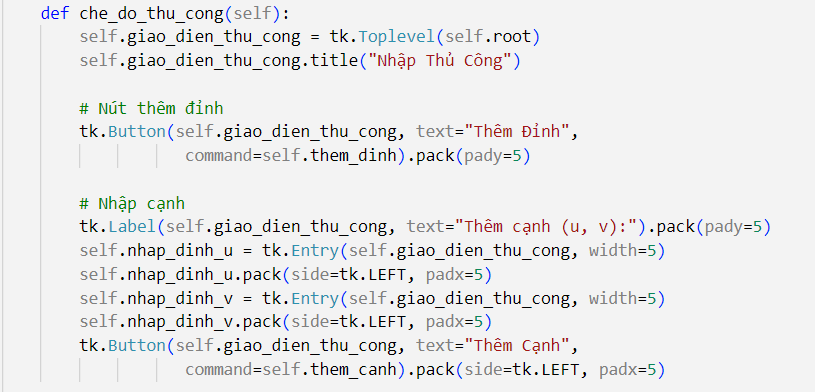
#### Xây dựng giao diện nhập liệu



Hình 14. Xây dựng giao diện nhập liệu

* Tạo một khung trong giao diện chính, căn chỉnh khoảng cách trên dưới 10px.
* Tạo button trong khung vừa tạo, khi click vào button sẽ gọi phương thức tương ứng:
  + Nút Thêm Đỉnh và Cạnh Thủ Công sẽ gọi phương thức che\_do\_thu\_cong.
  + Nút Nhập Ma Trận Kề sẽ gọi phương thức che\_do\_ma\_tran\_ke.
  + Nút Vẽ Đồ Thị sẽ gọi phương thức ve\_do\_thi.
  + Nút Tô Màu (Welch-Powell) sẽ gọi phương thức to\_mau\_Welch\_Powell.
  + Nút Tô Màu (Backtracking) sẽ gọi phương thức to\_mau\_Backtracking.
* Tạo label hiển thị kết quả tô màu trong giao diện chính.
* Tạo canvas trong giao diện chính để hỗ trợ vẽ và tô màu đồ thị.

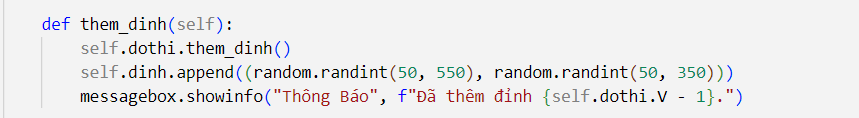
#### Phương thức nhập thủ công



Hình 15. Phương thức nhập thủ công

* Tạo cửa sổ con trên giao diện chính.
* Tạo button để gọi phương thức thêm đỉnh cho đồ thị.
* Tạo label cho biết thao tác nhập cạnh bên dưới.
* Tạo các entry để nhập cạnh.
* Tạo thêm một button để gọi phương thức thêm cạnh vào đồ thị.

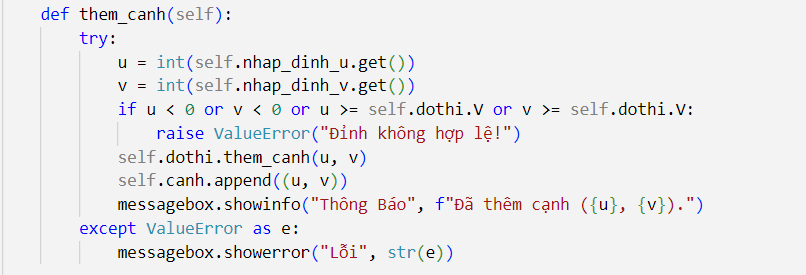
#### Phương thức tạo đỉnh thủ công



Hình 16. Phương thức tạo đỉnh thủ công

* Gọi phương thức them\_dinh của lớp Dothi để tiến hành thêm đỉnh.
* Tạo tọa độ ngẫu nhiên cho đỉnh và lưu vào danh sách self.dinh.
* Thông báo cho người dùng biết đã thêm đỉnh thành công.

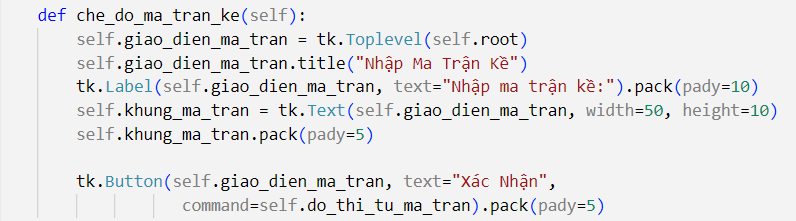
#### Phương thức tạo cạnh thủ công



Hình 17. Phương thức tạo cạnh thủ công

* Nhận đỉnh đã nhập ở phương thức che\_do\_nhap\_thu\_cong.
* Kiểm tra xem đỉnh u và v có nhỏ hơn 0 hay lớn hơn hoặc bằng self.dothi.V, nếu có, thông báo lỗi.
* Gọi phương thức them\_canh của lớp Dothi để thêm cạnh.
* Lưu cạnh vào danh sách self.canh.
* Thông báo cho người dùng biết đã thêm cạnh thành công.

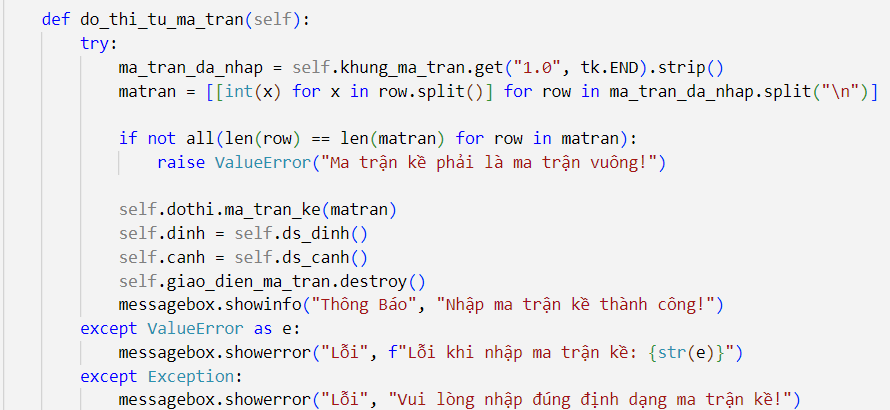
#### Phương thức nhập ma trận kề



Hình 18. Phương thức nhập ma trận kề

* Tạo cửa sổ con trên giao diện chính.
* Tạo label cho biết thao tác cần thực hiện.
* Tạo khung text cho phép người dùng nhập dữ liệu.
* Tạo button để gọi phương thức tạo đồ thị từ ma trận.

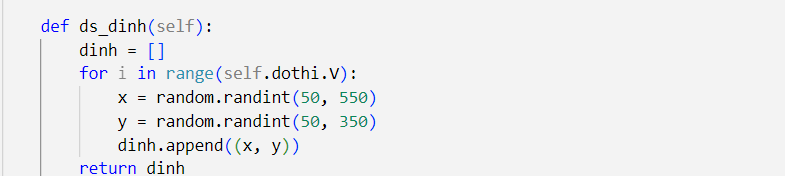
#### Phương thức tạo đồ thị từ ma trận



Hình 19. Phương thức tạo đồ thị từ ma trận

* Nhận ma trận ở phương thức che\_do\_ma\_tran\_ke, từ đầu đến cuối văn bản, dùng trip() để xóa các khoảng trắng.
* Chuyển chuỗi vừa nhận thành các dòng với \n và chuyển thành kiểu số nguyên.
* Nếu tồn tại dòng có số lượng phần tử khác với số dòng (số dòng khác với số cột) thì báo lỗi.
* Gọi phương thức ma\_tran\_ke của lớp Dothi để tạo đồ thị.
* Gọi phương thức ds\_dinh của lớp Dothi để tạo và lưu tọa độ các đỉnh.
* Gọi phương thức ds\_canh của lớp Dothi để tạo và lưu các cạnh.
* Đóng cửa sổ nhập ma trận và thông báo đã nhập thành công.
* Thông báo lỗi kiểu ValueError.
* Thông báo các lỗi khác.

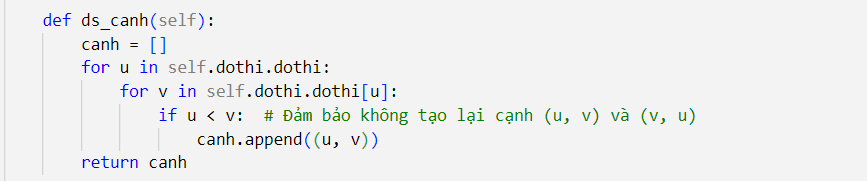
#### Phương thức tạo tọa độ các đỉnh từ ma trận kề



Hình 20. Phương thức tạo tọa độ các đỉnh từ ma trận kề

* dinh là danh sách lưu tọa độ các đỉnh.
* Lặp qua từng đỉnh i:
  + Tạo tọa độ ngẫu nhiên x trong khoảng (50, 550).
  + Tạo tọa độ ngẫu nhiên y trong khoảng (50, 350).
  + Lưu tọa độ (x, y) vào dinh.
* Trả về danh sách chứa tọa độ các đỉnh.

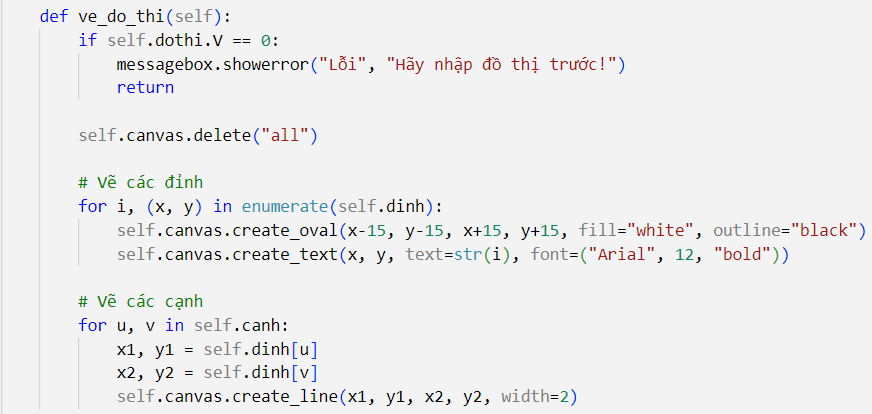
#### Phương thức tạo các cạnh từ ma trận kề



Hình 21. Phương thức tạo các cạnh từ ma trận kề

* canh là danh sách lưu các cạnh.
* Lặp qua từng đỉnh u trong danh sách các đỉnh kề:
  + Lặp qua từng đỉnh v kề với đỉnh u.
  + Nếu đỉnh hai đỉnh không bằng nhau thì thêm cặp đỉnh vào canh.
* Trả về danh sách chứa các cạnh.

#### Phương thức vẽ đồ thị



Hình 22. Phương thức vẽ đồ thị

* Kiểm tra đồ thị đã có đỉnh chưa, nếu chưa thì báo lỗi và kết thúc phương thức.
* Xóa nội dung trên canvas để vẽ đồ thị mới.
* Lặp qua từng đỉnh i và tọa độ của các đỉnh này:
  + Vẽ vòng tròn bao quanh chỉ số của đỉnh với bán kính 15px, màu ban đầu là trắng và viền đen.
  + Thêm chỉ số i vào tâm vòng tròn.
* Lặp qua từng cạnh:
  + Lấy tọa độ đỉnh u.
  + Lấy tọa độ đỉnh v.
  + Vẽ đường nối 2 đỉnh với nhau.

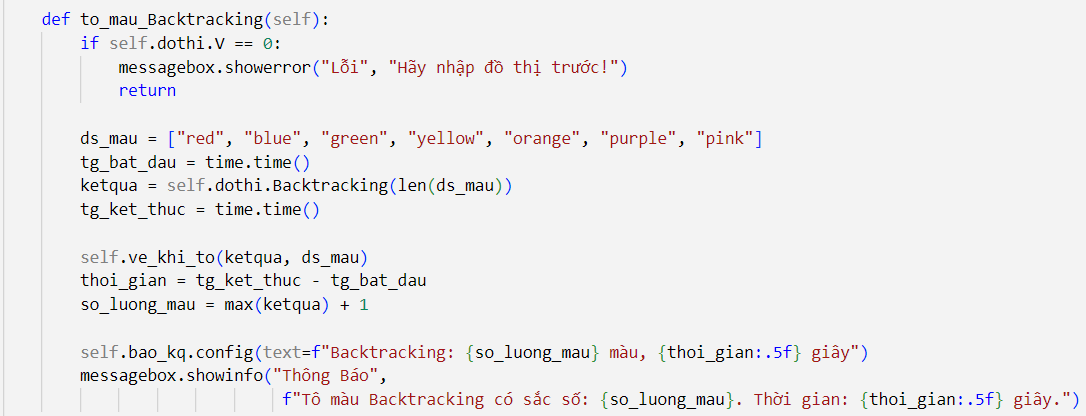
#### Phương thức tô màu bằng giải thuật Welch-Powell



Hình 23. Phương thức tô màu bằng giải thuật Welch-Powell

* Kiểm tra đồ thị đã có đỉnh chưa, nếu chưa thì báo lỗi và kết thúc phương thức.
* Tạo một danh sách lưu các màu sẽ sử dụng, có thể tạo nhiều màu hơn.
* Dùng tg\_bat\_dau để lưu thời gian bắt đầu thực hiện giải thuật.
* Gọi phương thức Welch\_Powell của lớp Dothi để tiến hành giải thuật.
* Dùng tg\_ket\_thuc để lưu thời gian kết thúc giải thuật.
* Gọi phương thức ve\_khi\_to để vẽ đồ thị.
* Tính thời gian thực hiện giải thuật.
* Tính số màu đã sử dụng.
* Hiện kết quả dạng văn bản trên giao diện bằng label bao\_kq đã tạo trước đó.
* Thông báo kết quả bằng hộp thoại.

#### Phương thức tô màu bằng giải thuật Backtracking

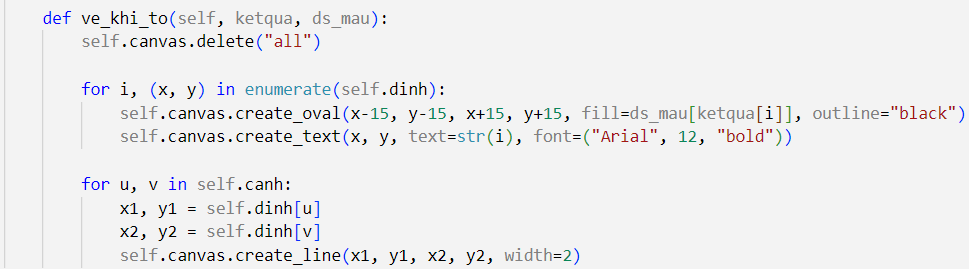


Hình 24. Phương thức tô màu bằng giải thuật Backtracking

* Kiểm tra đồ thị đã có đỉnh chưa, nếu chưa thì báo lỗi và kết thúc phương thức.
* Tạo một danh sách lưu các màu sẽ sử dụng, có thể tạo nhiều màu hơn.
* Dùng tg\_bat\_dau để lưu thời gian bắt đầu thực hiện giải thuật.
* Gọi phương thức Backtracking của lớp Dothi để thực hiện giải thuật.
* Dùng tg\_ket\_thuc để lưu thời gian kết thúc giải thuật.
* Gọi phương thức ve\_khi\_to để vẽ đồ thị.
* Tính thời gian thực hiện giải thuật.
* Tính số màu đã sử dụng.
* Hiện kết quả dạng văn bản trên giao diện bằng label bao\_kq đã tạo trước đó.
* Thông báo kết quả bằng hộp thoại.

#### Phương thức vẽ đồ thị khi thực hiện tô màu

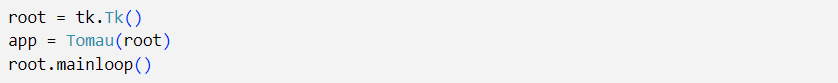
Phương thức ve\_do\_thi sẽ cho ta một đồ thị chưa được tô màu. Khi tô màu, ta vẫn phải gọi một phương thức khác để vẽ lại đồ thị dựa trên tọa độ đỉnh và cạnh đã được lưu trước đó:



Hình 25. Phương thức vẽ đồ thị khi thực hiện tô màu

* Xóa nội dung trên canvas để vẽ đồ thị mới.
* Lặp qua từng đỉnh i và tọa độ của các đỉnh này:
  + Vẽ vòng tròn bao quanh chỉ số của đỉnh với bán kính 15px, màu ban đầu là trắng và viền đen.
  + Thêm chỉ số i vào tâm vòng tròn.
* Lặp qua từng cạnh:
  + Lấy tọa độ đỉnh u.
  + Lấy tọa độ đỉnh v.
  + Vẽ đường nối 2 đỉnh với nhau.

#### Chạy chương trình



Hình 26. Chạy chương trình

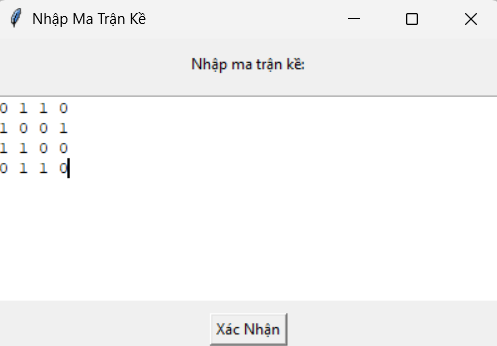
* Khởi tạo giao diện đồ họa.
* Tạo một đối tượng cho lớp Tomau.
* Giao diện sẽ được hiển thị liên tục với mainloop().

## Kiểm thử

Nhằm kiểm tra tính chính xác và so sánh hiệu năng của hai giải thuật: Welch-Powell và Backtracking. Em đã tiến hành chạy chương trình và so sánh kết quả nhiều lần. Dưới đây là một số ví dụ:

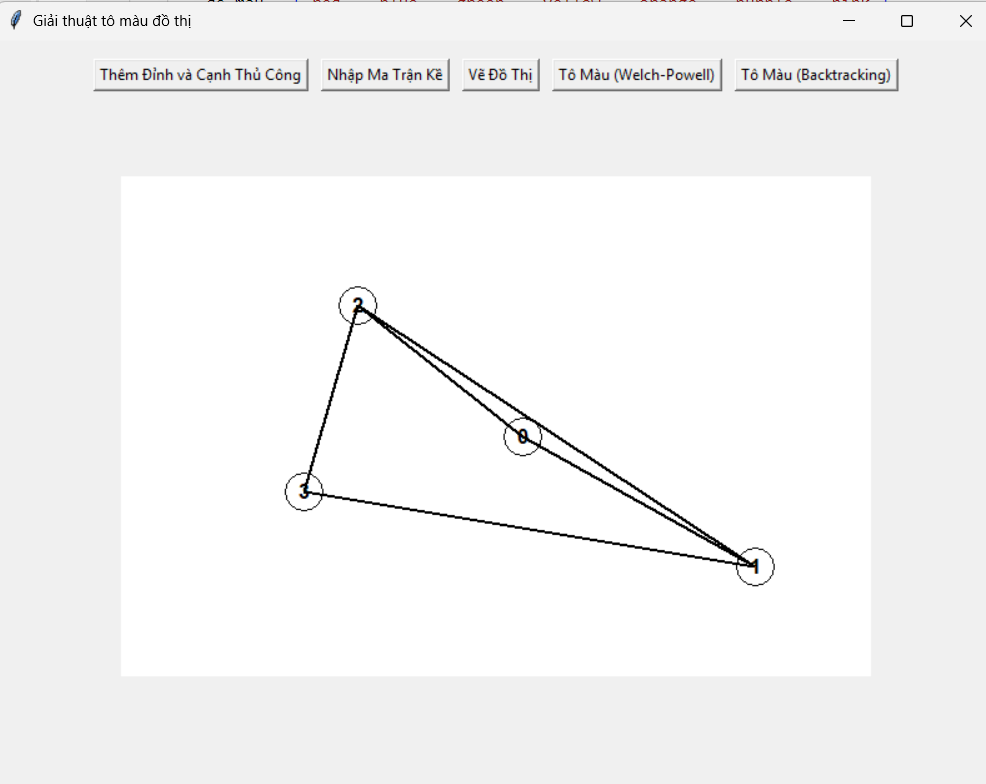
### Ví dụ 1

Nhập ma trận kề như bên dưới:

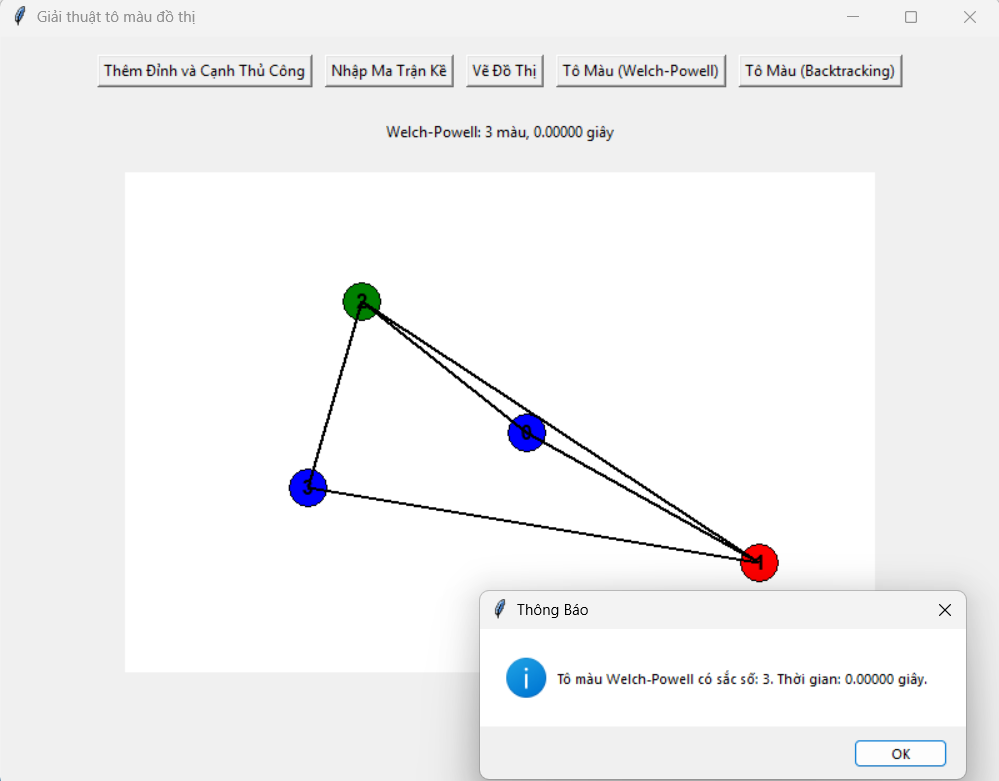


Hình 27. Ma trận ví dụ 1

Ta có đồ thị chưa tô màu:



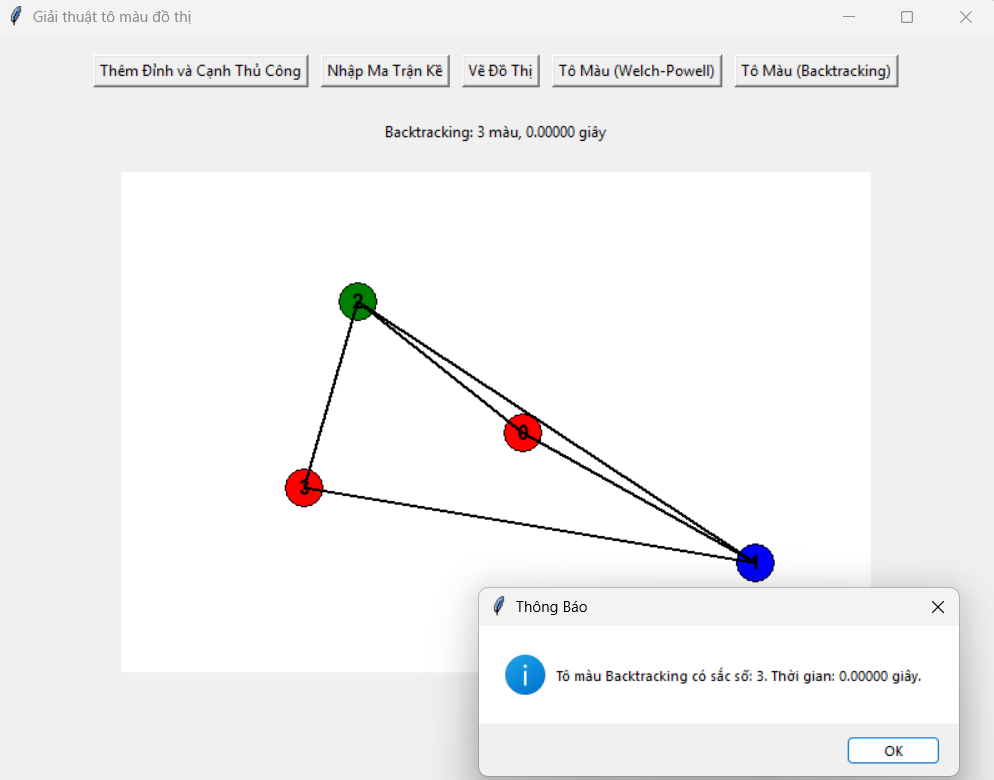
Hình 28. Đồ thị ví dụ 1



Hình 29. Kết quả tô màu Welch-Powell ví dụ 1

Với giải thuật tô màu Welch-Powell, ta tìm được sắc số là 3, thời gian thực hiện là

0, 00000 giây.



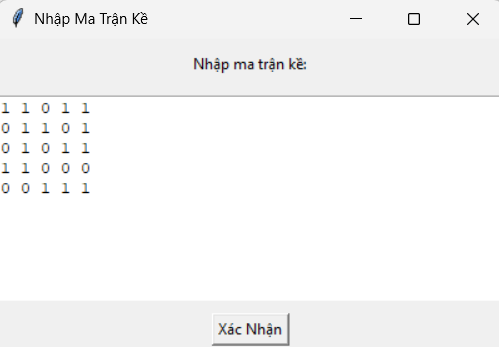
Hình 30. Kết quả tô màu Backtracking ví dụ 1

Với giải thuật Backtracking, ta cũng tìm được sắc số là 3, thời gian thực hiện là 0.00000 giây.

=> Ở đồ thị này, hai giải thuật cho kết quả như nhau về cả sắc số lẫn thời gian thực hiện.

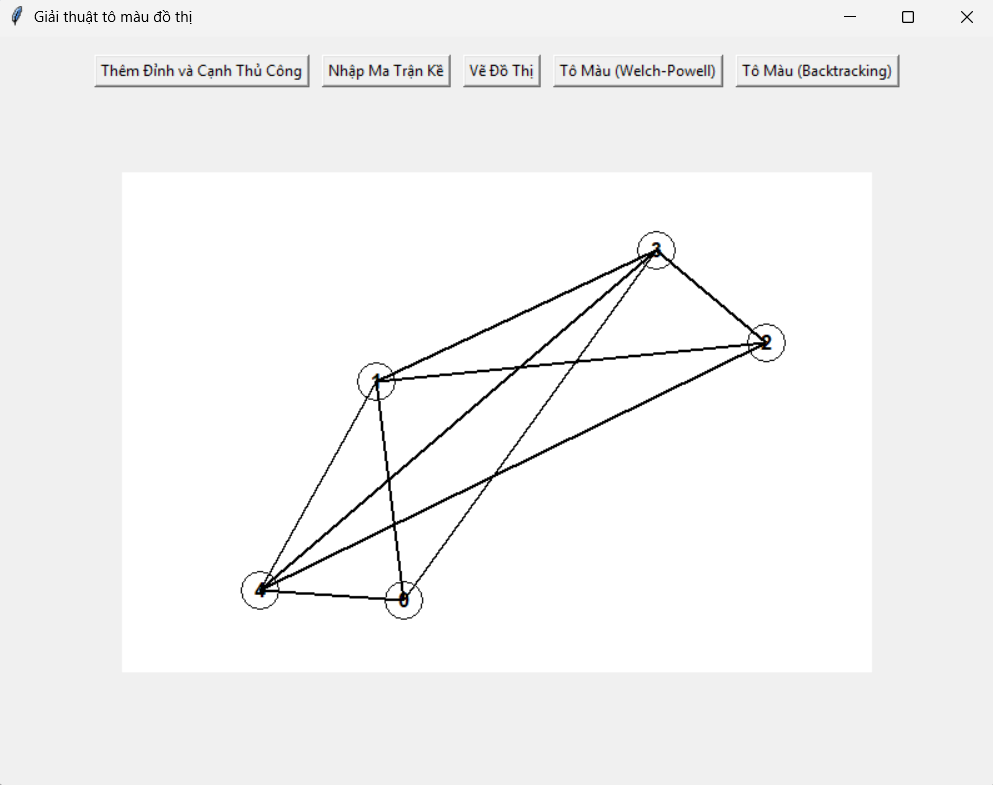
### Ví dụ 2

Nhập ma trận kề như bên dưới:

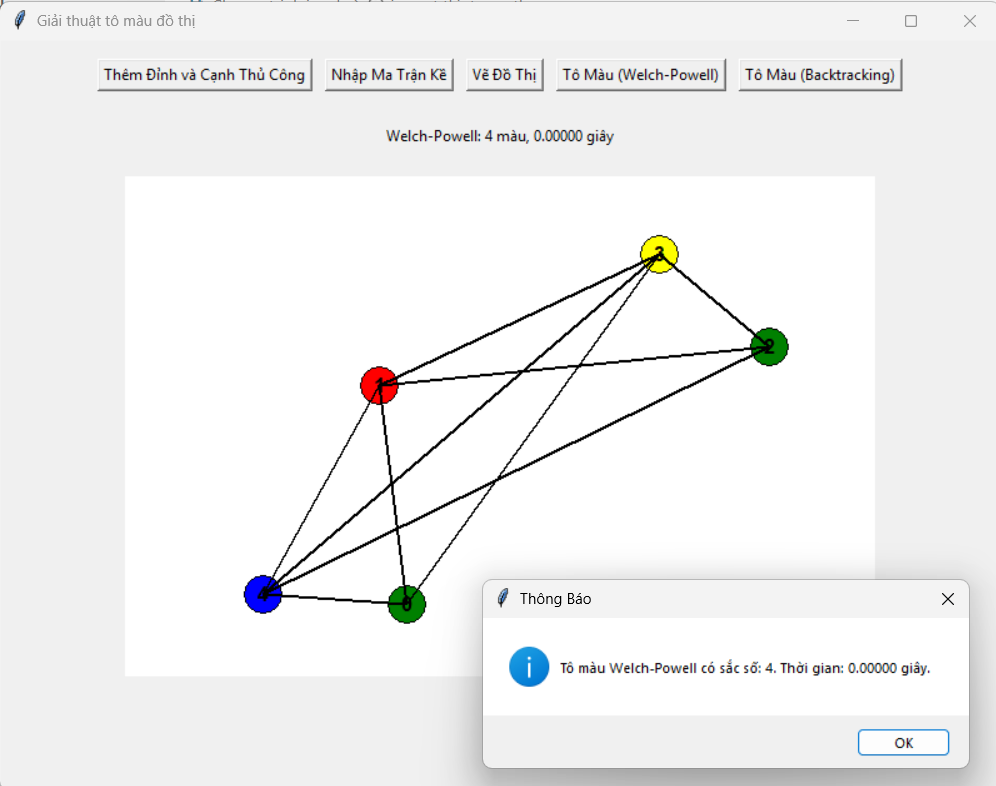


Hình 31. Ma trận ví dụ 2

Ta có đồ thị chưa tô màu:

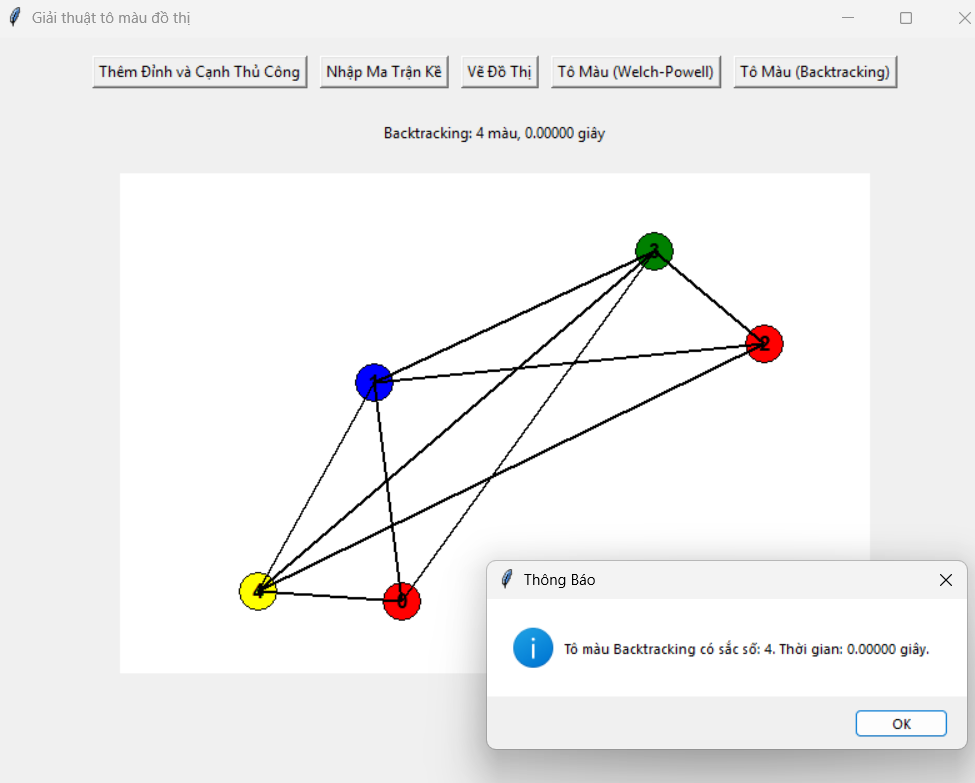
****

Hình 32. Đồ thị ví dụ 2



Hình 33. Kết quả tô màu Welch-Powell ví dụ 2

Giải thuật Welch-Powell cho sắc số là 4 và thời gian thực hiện là 0, 00000 giây.



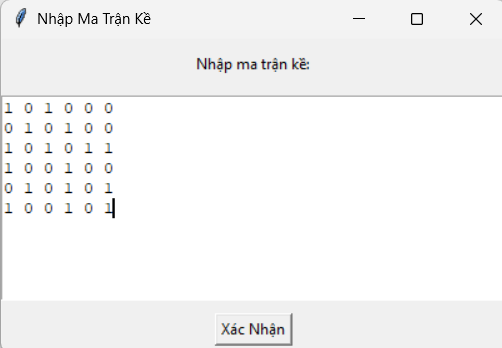
Hình 34. Kết quả tô màu Backtracking ví dụ 2

Giải thuật Backtracking cũng cho sắc số là 4 và có thời gian thực hiện như giải thuật Welch-Powell.

=> Ở đồ thị này, hiệu năng của hai giải thuật vẫn như nhau.

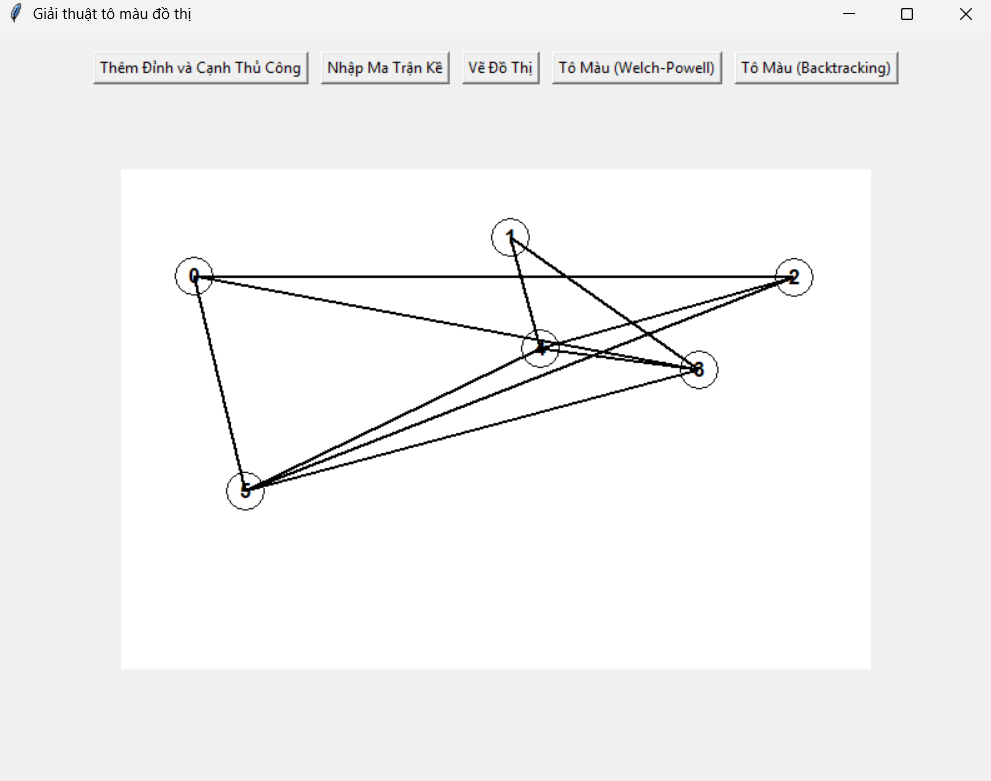
### Ví dụ 3

Nhập ma trận kề như bên dưới:

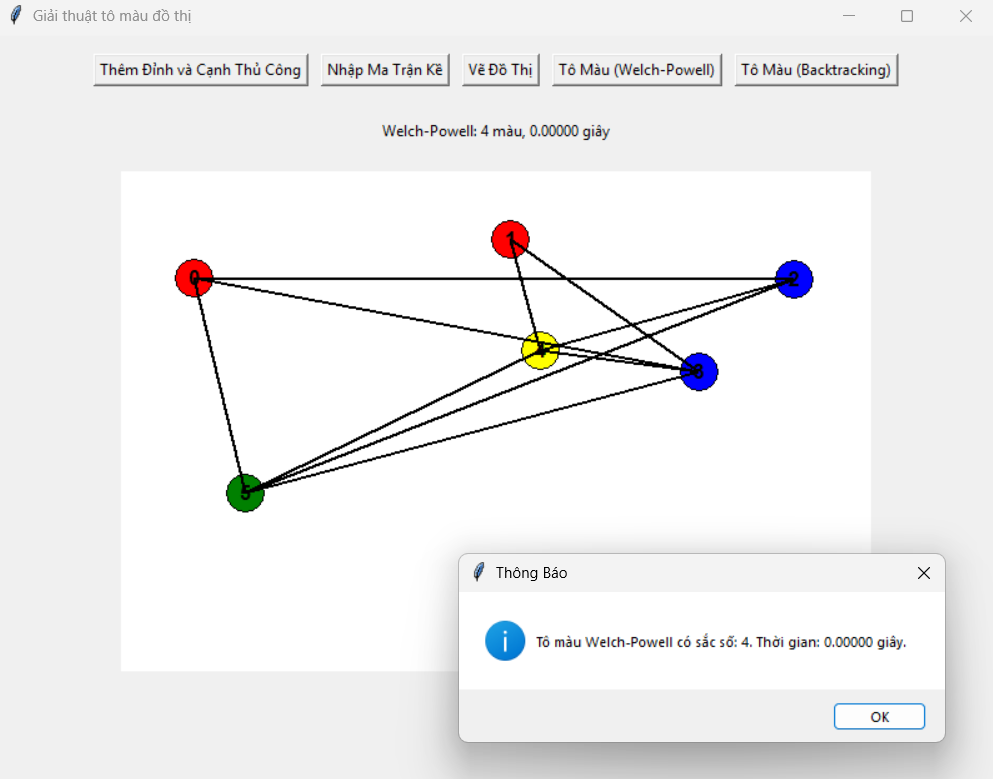


Hình 35. Ma trận ví dụ 3

Ta có đồ thị chưa tô màu:

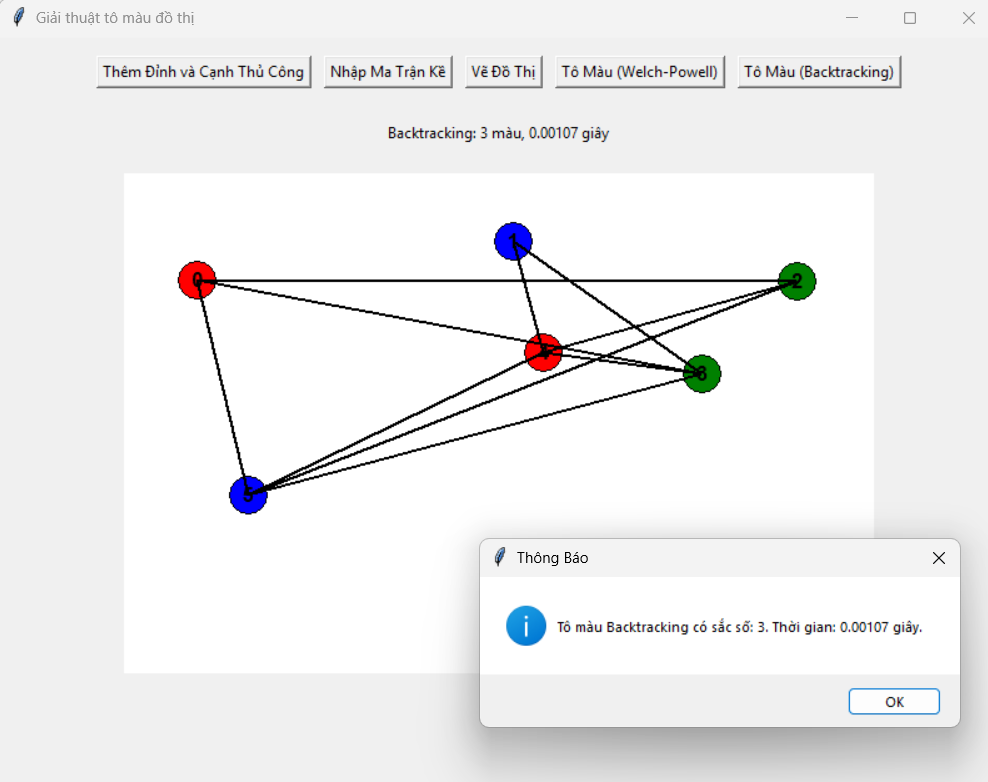


Hình 36. Đồ thị ví dụ 3



Hình 37. Kết quả tô màu Welch-Powell ví dụ 3

Giải thuật Welch-Powell cho sắc số là 4, thời gian thực hiện là 0, 00000 giây.



Hình 38. Kết quả tô màu Backtracking ví dụ 3

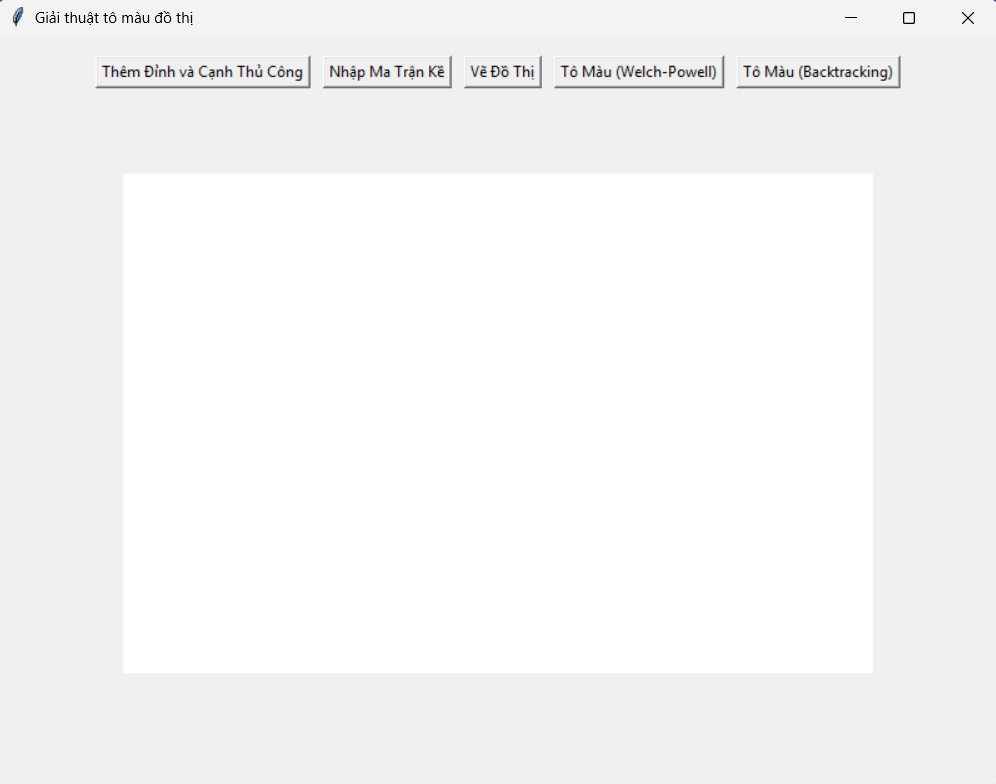
Giải thuật Backtracking cho sắc số là 3, thời gian thực hiện là 0, 00107 giây.

=> Với đồ thị này, giải thuật Backtracking cho số màu tối ưu và mất nhiều thời gian hơn so với giải thuật Welch-Powell.

# KẾT QUẢ NGHIÊN CỨU

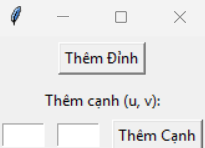
## Giao diện người dùng

### Giao diện chính



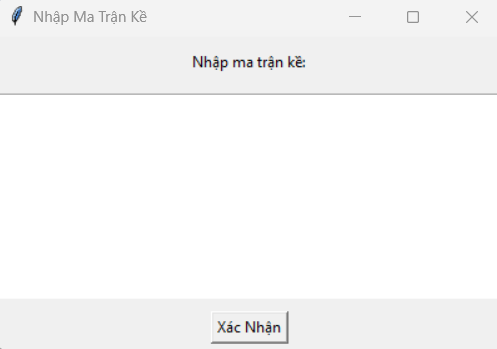
Hình 39. Giao diện chính

### Cửa sổ nhập đỉnh và cạnh thủ công



Hình 40. Cửa sổ nhập đỉnh và cạnh thủ công

### Cửa sổ nhập ma trận kề

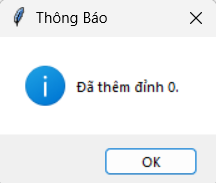
****

Hình 41. Cửa sổ nhập ma trận kề

## Cách sử dụng

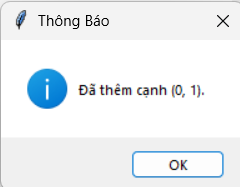
### Thêm đỉnh và cạnh thủ công

* Muốn thêm đỉnh và cạnh thủ công, ta click vào nút Thêm Đỉnh và Cạnh Thủ Công. Ta sẽ được cửa sổ như Hình 40.
* Click vào nút Thêm Đỉnh sẽ nhận được thông báo tạo đỉnh mới thành công, ví dụ:



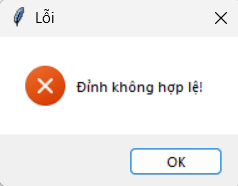
Hình 42. Thông báo thêm đỉnh thành công

* Nhập đỉnh vào khung nhập liệu và click nút Thêm Cạnh sẽ nhận được thông báo:
  + Nếu nhập đúng:



Hình 43. Thông báo thêm cạnh thành công

* + Nếu nhập sai:



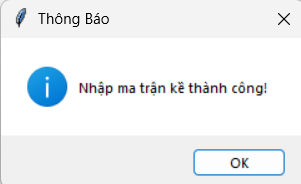
Hình 44. Thông báo thêm cạnh thất bại

**Lưu ý:** để tránh nhập sai đỉnh dẫn đến việc tạo cạnh thất bại, hãy lưu ý đến thông báo nhập đỉnh thành công và ghi nhớ tên đỉnh.

### Nhập ma trận kề

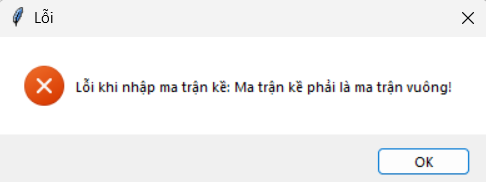
Ngoài nhập đỉnh và cạnh thủ công, ta có thể chọn cách nhập ma trận kề để tạo đồ thị.

* Click vào nút Nhập Ma Trận Kề trên giao diện chính, ta được cửa sổ như Hình 41.
* Nhập ma trận kề vào khung nhập liệu và click vào nút Xác Nhận sẽ nhận được thông báo:
  + Nếu nhập đúng:



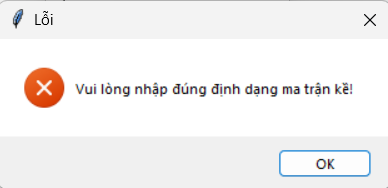
Hình 45. Thông báo nhập ma trận kề thành công

* Nếu nhập sai:



Hình 46. Thông báo nhập sai kiểu ma trận kề

Thông báo này xuất hiện khi nhập một ma trận không phải ma trận vuông.



Hình 47. Thông báo các lỗi khác khi nhập ma trận kề

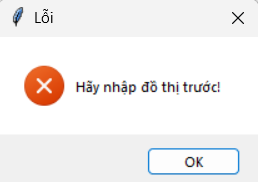
Thông báo này xuất hiện khi mắc các lỗi khác.

Lưu ý: để tránh nhập sai ma trận kề, hãy nhập đúng định dạng. Ma trận phải là ma trận vuông (số dòng và số cột bằng nhau), các phần tử cách nhau bằng khoảng trắng và phải là số nguyên 0, 1.

### Vẽ đồ thị

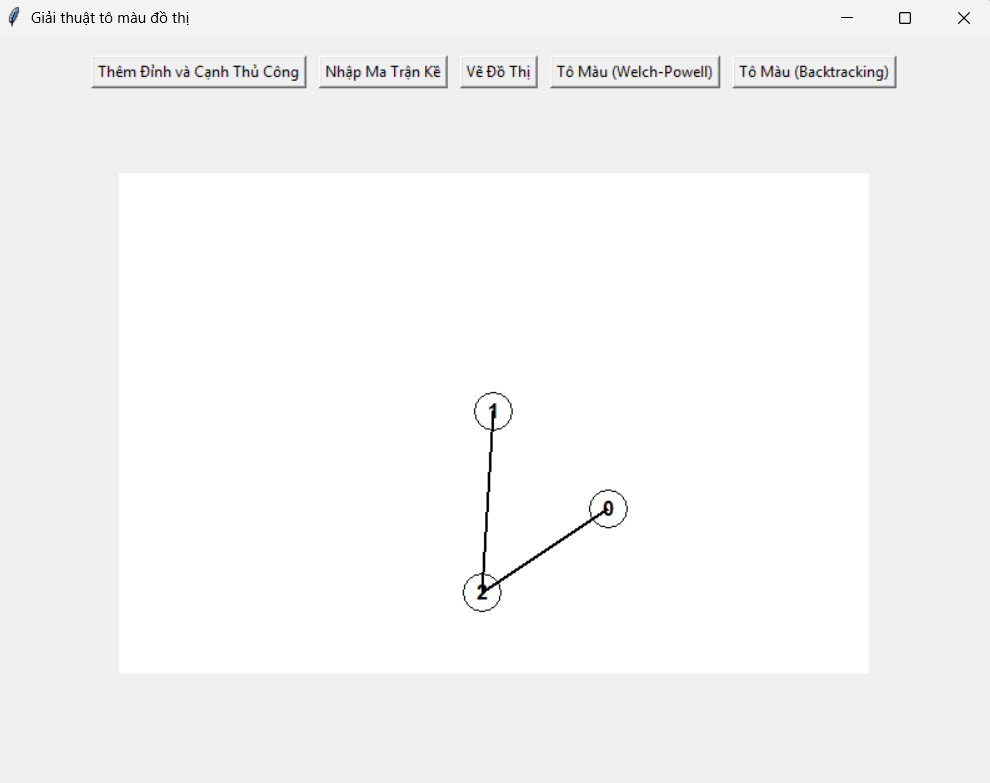
Click vào nút vẽ đồ thị:

* Nếu chưa nhập đỉnh hoặc nhập ma trận kề chưa thành công, sẽ nhận được thông báo:

****

Hình 48. Yêu cầu nhập đồ thị

* Nếu đã nhập đỉnh hoặc nhập ma trận kề thành công, sẽ nhận được kết quả, ví dụ:

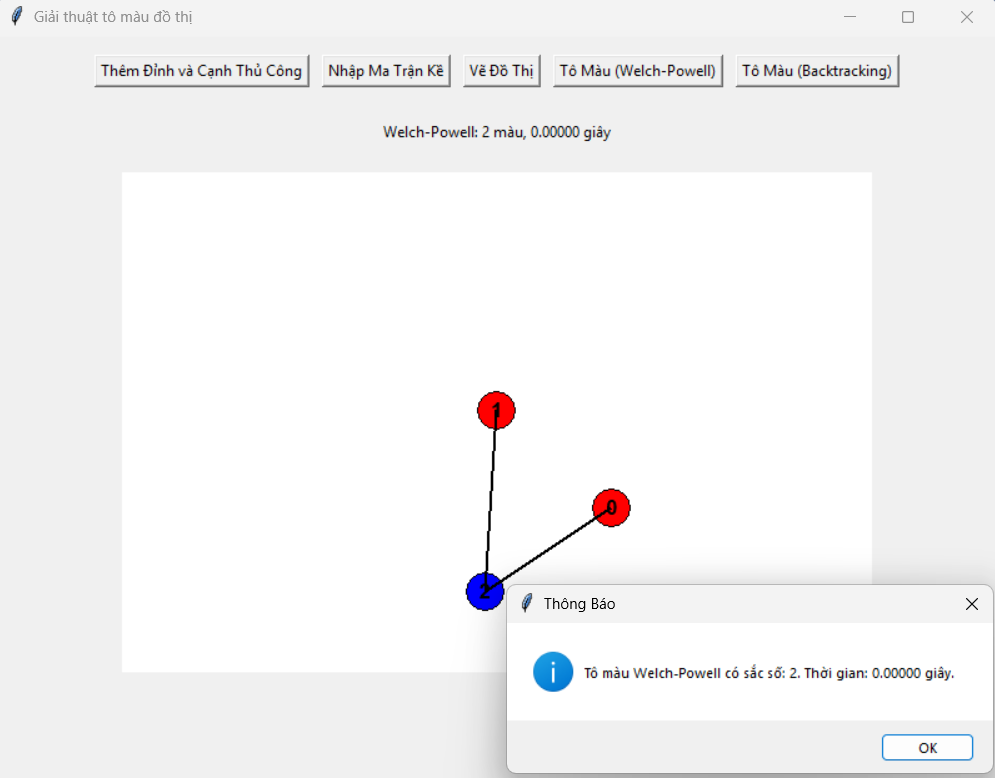
****

Hình 49. Ví dụ đồ thị

### Tô màu đồ thị

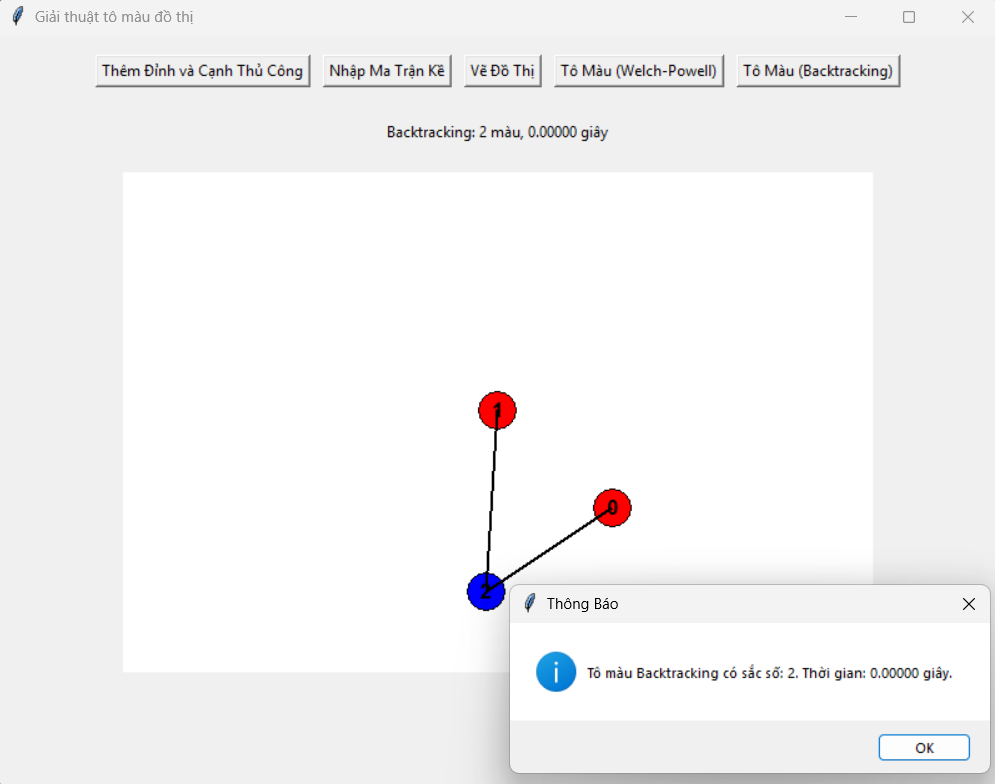
Click chuột vào một trong hai nút Tô Màu (Welch-Powell) và Tô Màu (Backtracking), sẽ nhận được kết quả tương ứng với từng giải thuật:

* Giải thuật Welch-Powell:



Hình 50. Ví dụ tô màu Welch-Powell

* Giải thuật Backtracking:



Hình 51. Ví dụ tô màu Backtracking

## Hiệu năng

Sau khi chạy thử chương trình với nhiều đồ thị có kích thước nhỏ khác nhau, em có đánh giá về hiệu năng của hai giải thuật như sau:

* Giải thuật Backtracking cho số màu tối ưu hơn.
* Hai giải thuật đều có tốc độ tô màu rất nhanh, khá ít trường hợp có một giải thuật có tốc độ chậm hơn 0, 00000 giây.
* Trong quá trình thử nghiệm, có thể thấy giải thuật Backtracking khá mất thời gian với đồ thị phức tạp (Ví dụ 3).

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận: Xây dựng thành công giao diện người dùng, đồng thời cũng thành công cài đặt giải thuật, tích hợp với đồ họa, giúp trực quan hóa bài toán tô màu đồ thị. Có thể ứng dụng vào thực tế như sắp xếp lịch thi, thời khóa biểu, đèn giao thông,…

Khó khăn:

* Về lý thuyết: Có rất ít tài liệu liên quan đến thuật toán Backtracking, việc nghiên cứu lý thuyết khó khăn hơn mong đợi.
* Về thuật toán: Do thực hiện nghiên cứu với các đồ thị nhỏ, việc thực hiện thuật toán khá đơn giản, nên gần như không thể so sánh kết quả thực hiện. Hầu hết các đồ thị thử nghiệm, hai giải thuật đều cho kết quả giống nhau cả về sắc số và thời gian thực hiện.

Hướng phát triển: Có thể mở rộng phạm vi đồ thị, cài đặt chương trình sử dụng thuật toán phức tạp hơn nhằm thử nghiệm với những đồ thị kích thước vừa và lớn để ứng dụng nhiều hơn trong thực tế.

# DANH MỤC TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | T. H. Nam, TÀI LIỆU GIẢNG DẠY MÔN LÝ THUYẾT ĐỒ THỊ, Trà Vinh, 2013. |
| [2] | P. T. Anh và P. M. Trung, “NGHIÊN CỨU BÀI TOÁN TÔ MÀU ĐỒ THỊ,” Tp.Hồ Chí Minh, 2017. |