

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

~~~~~\*~~~~~



TRƯỜNG ĐẠI HỌC  
Sư phạm Thành phố Hồ Chí Minh  
HCMC University of Education



## ĐỒ ÁN CUỐI KỲ

ĐỀ TÀI  
MINH HOẠ TRỰC QUAN GIAO DIỆN ĐỒ HOẠ  
BẰNG THUẬT TOÁN KRUSKAL VÀ FLOYD

Học phần: 2111COMP170101 – LÝ THUYẾT ĐỒ THỊ VÀ  
ỨNG DỤNG

TP HỒ CHÍ MINH – 11/2021

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**  
**TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH**  
**KHOA CÔNG NGHỆ THÔNG TIN**

~~~~~\*~~~~~

ĐỒ ÁN CUỐI KỲ

ĐỀ TÀI
MINH HOẠ TRỰC QUAN GIAO DIỆN ĐỒ HOẠ
BẰNG THUẬT TOÁN KRUSKAL VÀ FLOYD

Học phần: 2111COMP170101 – LÝ THUYẾT ĐỒ THỊ VÀ
ỨNG DỤNG

| | | | |
|------------------------|-----------------------|---|---------------|
| Sinh viên thực hiện 1: | Nguyễn Hữu Minh Dương | : | 46.01.104.039 |
| Sinh viên thực hiện 2: | Nguyễn Chí Khang | : | 46.01.104.078 |
| Sinh viên thực hiện 3: | Huỳnh Thị Yên Khoa | : | 46.01.104.087 |
| Sinh viên thực hiện 4: | Trần Ngọc Phương Linh | : | 46.01.104.091 |
| Sinh viên thực hiện 5: | Phước Công Nguyên | : | 46.01.104.125 |

Giảng viên hướng dẫn: ThS. Lương Trần Ngọc Khiết

TP HỒ CHÍ MINH – 11/2021

MỤC LỤC

| | |
|--|----|
| 1. Lý do chọn đề tài | 4 |
| 2. Những kiến thức cơ sở | 4 |
| 2.1. Khái niệm thuật toán: | 4 |
| 2.2. Định nghĩa về đồ thị: | 4 |
| 2.2.1. Đồ thị vô hướng..... | 4 |
| 2.2.2. Đồ thị có hướng..... | 5 |
| 3. Lý thuyết của thuật toán Kruskal và Floyd | 5 |
| 3.1. Thuật toán Kruskal | 5 |
| 3.1.1. Khái niệm | 5 |
| 3.1.2. Các bước thực hiện..... | 5 |
| 3.1.3. Ví dụ | 5 |
| 3.2. Thuật toán Floyd | 6 |
| 3.2.1. Khái niệm | 6 |
| 3.2.2. Ví dụ | 7 |
| 4. Chương trình..... | 8 |
| 4.1. Giới thiệu chương trình | 8 |
| 4.2. Chức năng của chương trình | 8 |
| 4.3. Mô tả chương trình | 9 |
| 4.4. Mã nguồn..... | 13 |

NHIỆM VỤ THÀNH VIÊN NHÓM

| STT | Họ và tên | MSSV | Nhiệm vụ |
|-----|-----------------------|---------------|-----------------|
| 1 | Nguyễn Hữu Minh Dương | 46.01.104.039 | Code chính |
| 2 | Nguyễn Chí Khang | 46.01.104.078 | Code phụ + Word |
| 3 | Huỳnh Thị Yến Khoa | 46.01.104.087 | Video demo |
| 4 | Trần Ngọc Phương Linh | 46.01.104.091 | Báo cáo đồ án |
| 5 | Phước Công Nguyên | 46.01.104.125 | Code phụ + word |

Bảng phân công nhiệm vụ

Tự đánh giá:

Các thành viên trong nhóm hoàn thành tốt nhiệm vụ được giao, đúng thời gian.

NỘI DUNG

1. Lý do chọn đề tài

Lý thuyết đồ thị là một vấn đề phức tạp nhưng lại chiếm một vị trí quan trọng trong khối kiến thức cơ sở của ngành Công nghệ thông tin. Để hiểu được học phần này cần phải có thời gian tìm hiểu lâu dài. Việc dạy và học về đồ thị cũng gặp nhiều khó khăn. Các thuật toán là một trong những phần khó nhất trong môn học, ví dụ như việc học thuật toán Kruskal qua những dòng lệnh, người học khó có thể hiểu được những kiến thức mà người dạy truyền đạt. Chính vì vậy, việc minh họa trực quan giao diện đồ họa bằng các thuật toán lý thuyết đồ thị là việc làm cần thiết để người học có thể hiểu được nguyên lý của các thuật toán. Tính ưu việt của việc minh họa giao diện đồ thị là rất trực quan, có chú giải, đồng thời với giao diện đồ thị như vậy, người học có thể tương tác với hệ thống mô phỏng. Việc sử dụng phương thức này giúp cho người dạy dễ dàng truyền đạt ý tưởng của thuật toán, giúp cho người học hiểu rõ hơn về những thuật toán.

Như vậy, việc minh họa trực quan giao diện đồ thị bằng những thuật toán lý thuyết đồ thị giúp cho công tác dạy và học trở nên hiệu quả hơn. Thuật toán về đồ thị chiếm số lượng khá lớn, tuy nhiên số thuật toán được đưa vào giảng dạy còn hạn chế. Vì vậy trong khuôn khổ của mình, chúng em xin nghiên cứu việc minh họa trực quan giao diện đồ thị bằng thuật toán Kruskal và thuật toán Floyd.

2. Những kiến thức cơ sở

2.1. Khái niệm thuật toán:

Thuật toán là một hệ thống chặt chẽ và rõ ràng các quy tắc nhằm xác định một dãy các thao tác trên cấu trúc dữ liệu sao cho: với một bộ dữ liệu vào, sau một hữu hạn bước thực hiện các thao tác đã chỉ ra, ta đạt được mục tiêu đã định.

2.2. Định nghĩa về đồ thị:

Trong toán học và tin học, đồ thị là đối tượng nghiên cứu cơ bản của lý thuyết đồ thị. Một cách không chính thức, đồ thị là một tập các đối tượng gọi là đỉnh nối với nhau bởi các cạnh. Thông thường, đồ thị được vẽ dưới dạng một tập các điểm (đỉnh, nút) nối với nhau bởi các đoạn thẳng (cạnh). Tùy theo ứng dụng mà một số cạnh có thể có hướng.

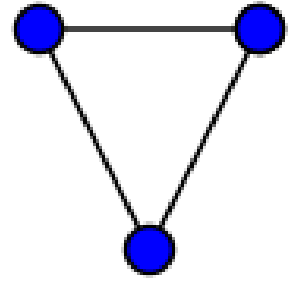
2.2.1. Đồ thị vô hướng

Đồ thị vô hướng hoặc đồ thị G là một cặp không có thứ tự (unordered pair) $G:=(V, E)$, trong đó

- V : tập các đỉnh hoặc nút,

- E: tập các cặp không thứ tự chứa các đỉnh phân biệt, được gọi là cạnh. Hai đỉnh thuộc một cạnh được gọi là các đỉnh đầu cuối của cạnh đó.

Trong nhiều tài liệu, tập các cạnh bao gồm cả các cặp đỉnh không phân biệt, các cạnh này được gọi là các khuyên. V (và E) thường là các tập hữu hạn, phần lớn các kết quả nghiên cứu đã biết không đúng (hoặc khác) khi áp dụng cho đồ thị vô hạn (infinite graph) vì nhiều luận cứ không dùng được trong trường hợp vô hạn.

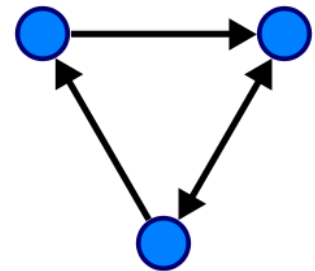


Hình 2.1 Đồ thị vô hướng

2.2.2. Đồ thị có hướng

Đồ thị có hướng G là một cặp có thứ tự $G:=(V, A)$, trong đó:

- V: tập các đỉnh hoặc nút,
- A: tập các cặp có thứ tự chứa các đỉnh, được gọi là các cạnh có hướng hoặc cung. Một cạnh $e = (x, y)$ được coi là có hướng từ x tới y ; x được gọi là điểm đầu/gốc và y được gọi là điểm cuối/ngọn của cạnh.



Hình 2.2 Đồ thị có hướng

Ngoài ra còn đơn đồ thị, đa đồ thị, đồ thị hỗn hợp, ...

3. Lý thuyết của thuật toán Kruskal và Floyd

3.1. Thuật toán Kruskal

3.1.1. Khái niệm

Kruskal là thuật toán để tìm cây khung nhỏ nhất của một đồ thị liên thông có trọng số. Thuật toán này là tìm một tập hợp các cạnh tạo thành một cây chứa tất cả các đỉnh của đồ thị và có tổng trọng số các cạnh là nhỏ nhất. Thuật toán được xuất bản lần đầu tiên năm 1956 bởi Joseph Kruskal.

3.1.2. Các bước thực hiện

Cho G là đồ thị liên thông, có trọng số và n đỉnh

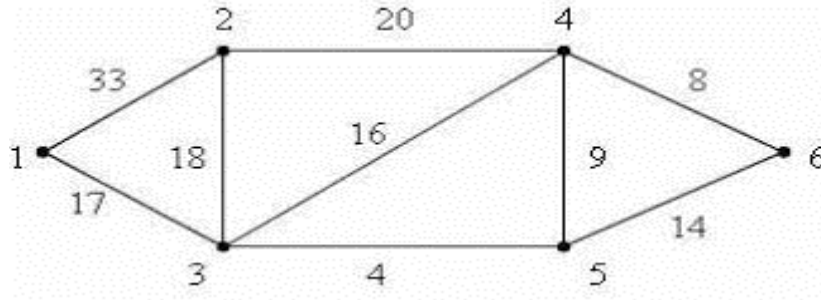
Bước 1: Trước hết, chọn các cạnh ngắn nhất e_1 trong các cạnh G .

Bước 2: Khi đã chọn k cạnh e_1, e_2, \dots, e_n thì chọn tiếp cạnh e_{k+1} ngắn nhất trong các cạnh còn lại của G sao cho không tạo thành chu trình với các cạnh đã chọn trước.

Bước 3: Chọn đủ $n-1$ cạnh thì dừng.

3.1.3. Ví dụ

Cho đồ thị sau:



Sắp xếp tất cả các cạnh của đồ thị theo thứ tự tăng dần.

| Điểm bắt đầu | Điểm kết thúc | Trọng số |
|--------------|---------------|----------|
| 3 | 5 | 4 |
| 4 | 6 | 8 |
| 4 | 5 | 9 |
| 5 | 6 | 14 |
| 3 | 4 | 16 |
| 1 | 3 | 17 |
| 2 | 3 | 18 |
| 2 | 4 | 20 |
| 1 | 2 | 33 |

Từ bảng trên ta có:

$$S1 = \{35\}$$

$$S2 = \{35, 46\}$$

$$S3 = \{35, 46, 45\}$$

$$S4 = \{35, 46, 45, 13\}$$

$$S5 = \{35, 46, 45, 13, 23\}$$

Như vậy tổng trọng số = $4 + 8 + 9 + 17 + 18 = 56$.

3.2. Thuật toán Floyd

3.2.1. Khái niệm

Thuật toán Floyd-Warshall còn được gọi là thuật toán Floyd được Robert Floyd tìm ra năm 1962 là thuật toán để tìm đường đi ngắn nhất giữa mọi cặp đỉnh. Floyd hoạt động được trên đồ thị có hướng, có thể có trọng số âm, tuy nhiên không có chu trình âm. Ngoài ra, Floyd còn có thể được dùng để phát hiện chu trình âm.

3.2.2. Ví dụ

Đề bài: Cho đồ thị vô hướng G sau:

Tìm đường đi ngắn nhất giữa các cặp đỉnh trong đồ thị trên.

Giải:

Thuật toán Floyd Warshall được mô tả như sau:

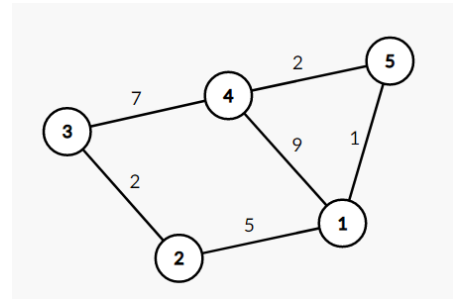
Khởi tạo ma trận khoảng cách ban đầu, ta được:

| | 1 | 2 | 3 | 4 | 5 |
|---|----------|----------|----------|----------|----------|
| 1 | 0 | 5 | ∞ | 9 | 1 |
| 2 | 5 | 0 | 2 | ∞ | ∞ |
| 3 | ∞ | 2 | 0 | 7 | ∞ |
| 4 | 9 | ∞ | 7 | 0 | 2 |
| 5 | 1 | ∞ | ∞ | 2 | 0 |

Quá trình thuật toán diễn ra như sau:

Chọn lần lượt từng đỉnh của đồ thị làm đỉnh trung gian (ta quy ước là K). Chọn một cặp hai đỉnh phân biệt và không trùng với đỉnh trung gian (ta quy ước lần lượt là I và J).

Thực hiện so sánh như ở trên: Đường đi ngắn nhất giữa I và J sẽ bằng giá trị nhỏ nhất của một trong hai giá trị sau:



Giá trị đường đi ngắn nhất hiện thời giữa I và J.

Tổng của giá trị đường đi ngắn nhất hiện thời giữa I và K, và đường đi ngắn nhất hiện thời giữa K và J.

Đầu tiên, $K = 1$. Nhờ đỉnh 1 làm trung gian, ta thấy xuất hiện đường đi từ đỉnh 2 tới đỉnh 4 (độ dài 14), và từ đỉnh 2 tới đỉnh 5 (độ dài 6). Đường đi trung gian qua đỉnh 1 để đi từ đỉnh 4 tới đỉnh 5 không tối ưu về chiều dài ($9 + 1 > 2$) nên ta không cập nhật lại đường đi ngắn nhất giữa hai đỉnh 4 và 5.

Mảng lúc này trở thành:

| | 1 | 2 | 3 | 4 | 5 |
|---|----------|-----------|----------|-----------|----------|
| 1 | 0 | 5 | ∞ | 9 | 1 |
| 2 | 5 | 0 | 2 | 14 | 6 |
| 3 | ∞ | 2 | 0 | 7 | ∞ |
| 4 | 9 | 14 | 7 | 0 | 2 |
| 5 | 1 | 6 | ∞ | 2 | 0 |

Tiếp theo, ta duyệt tới $K = 2$. Đường đi từ 3 tới 1 (độ dài 7), từ 3 tới 5 (độ dài 8) được hình thành. Đường đi từ 3 tới 4 không cập nhật độ dài ($7 < 2 + 5 + 9$).

| | 1 | 2 | 3 | 4 | 5 |
|---|---|----|---|----|---|
| 1 | 0 | 5 | 7 | 9 | 1 |
| 2 | 5 | 0 | 2 | 14 | 6 |
| 3 | 7 | 2 | 0 | 7 | 8 |
| 4 | 9 | 14 | 7 | 0 | 2 |
| 5 | 1 | 6 | 8 | 2 | 0 |

Cứ tiếp tục lựa chọn K như vậy cho tới hết, ta sẽ thu được mảng 2D hoàn chỉnh:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 5 | 7 | 3 | 1 |
| 2 | 5 | 0 | 2 | 8 | 6 |
| 3 | 7 | 2 | 0 | 7 | 8 |
| 4 | 3 | 8 | 7 | 0 | 2 |
| 5 | 1 | 6 | 8 | 2 | 0 |

Giả sử, qua mảng này, ta thấy đường đi ngắn nhất từ đỉnh 2 tới đỉnh 4 có độ dài 8. Dựa theo đồ thị thì nó là đoạn đường sau: **2 -> 1 -> 5 -> 4**.

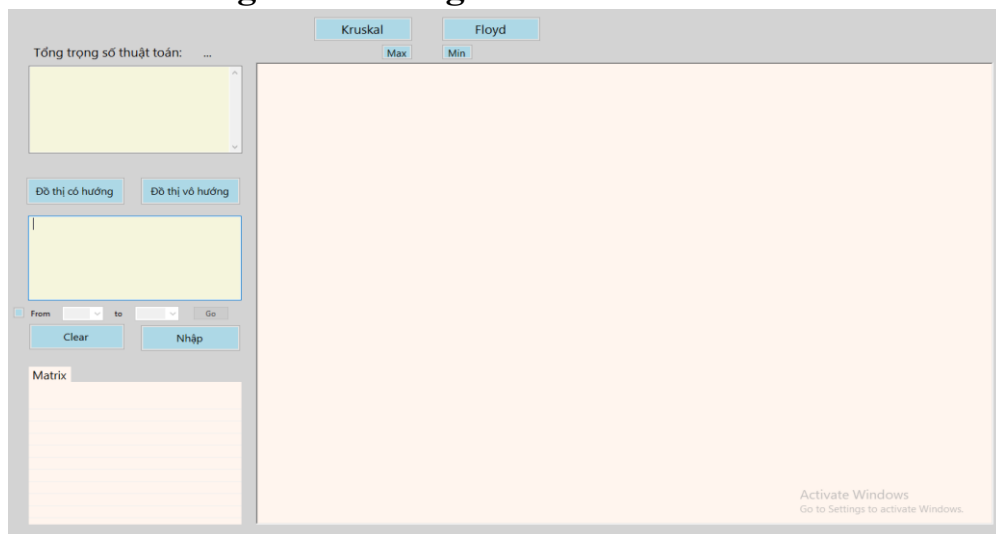
4. Chương trình

4.1. Giới thiệu chương trình

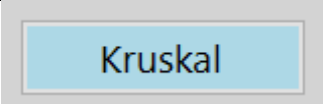
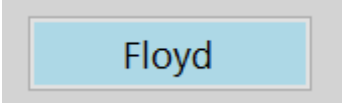

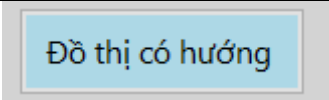
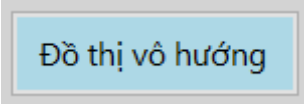
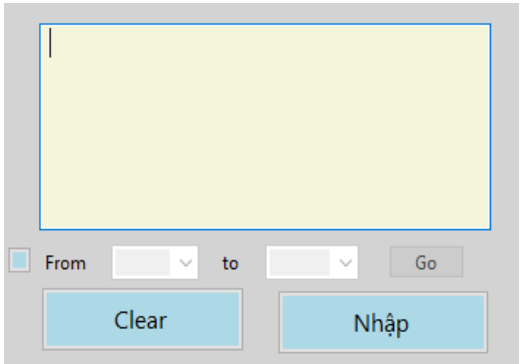
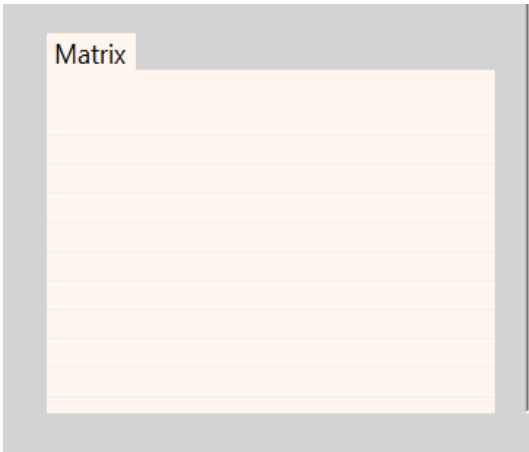
Chương trình này minh họa trực quan về tìm đường đi ngắn nhất của đồ thị bằng thuật toán Kruskal và thuật toán Floyd.

Ngôn ngữ chính được sử dụng để viết chương trình là ngôn ngữ C#.

4.2. Chức năng của chương trình



Hình 4.1 Màn hình chính của chương trình

| NÚT | CHỨC NĂNG |
|---|---|
|  | Tìm đường đi của đồ thị theo thuật toán Kruskal |
|  | Tìm đường đi của đồ thị theo thuật toán Floyd |
|  | Tìm đường đi (hoặc cây khung) lớn nhất và nhỏ nhất của đồ thị |
|  | Xác định đồ thị cần nhập là đồ thị có hướng |
|  | Xác định đồ thị cần nhập là đồ thị vô hướng |
|  | Nhập dữ liệu của đồ thị, trong đó: <ul style="list-style-type: none"> - Clear: Xóa tất cả - Nhập: Thực hiện minh họa đồ thị sau khi nhập - From (A) to (B): Tìm từ đỉnh A tới đỉnh B - Refresh: Vẽ lại đồ thị |
|  | Hiện đồ thị dưới dạng ma trận |

4.3. Mô tả chương trình

Để chương trình hoạt động chúng ta sẽ nhập dữ liệu vào khung dữ liệu.

Tổng trọng số thuật toán: ...

Kruskal Floyd

Max Min

Đồ thị có hướng Đồ thị vô hướng

Nhập dữ liệu vào khung

From to Go

Clear Nhập

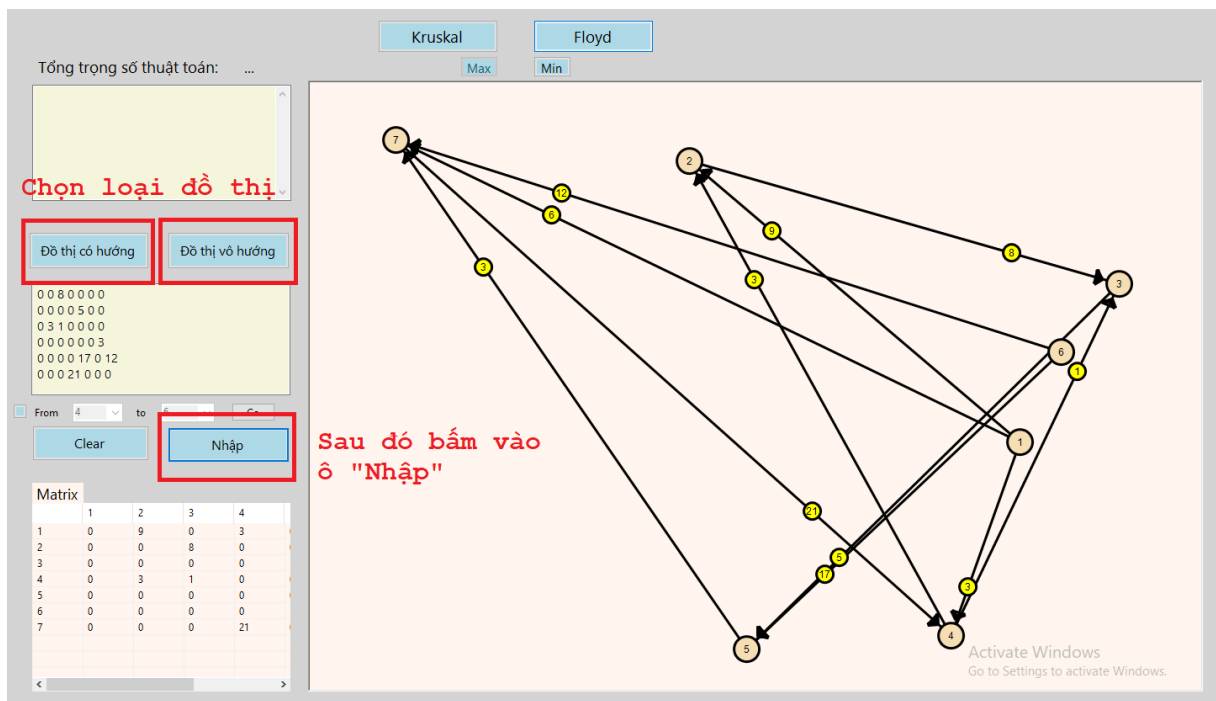
Matrix

Activate Windows
Go to Settings to activate Windows.

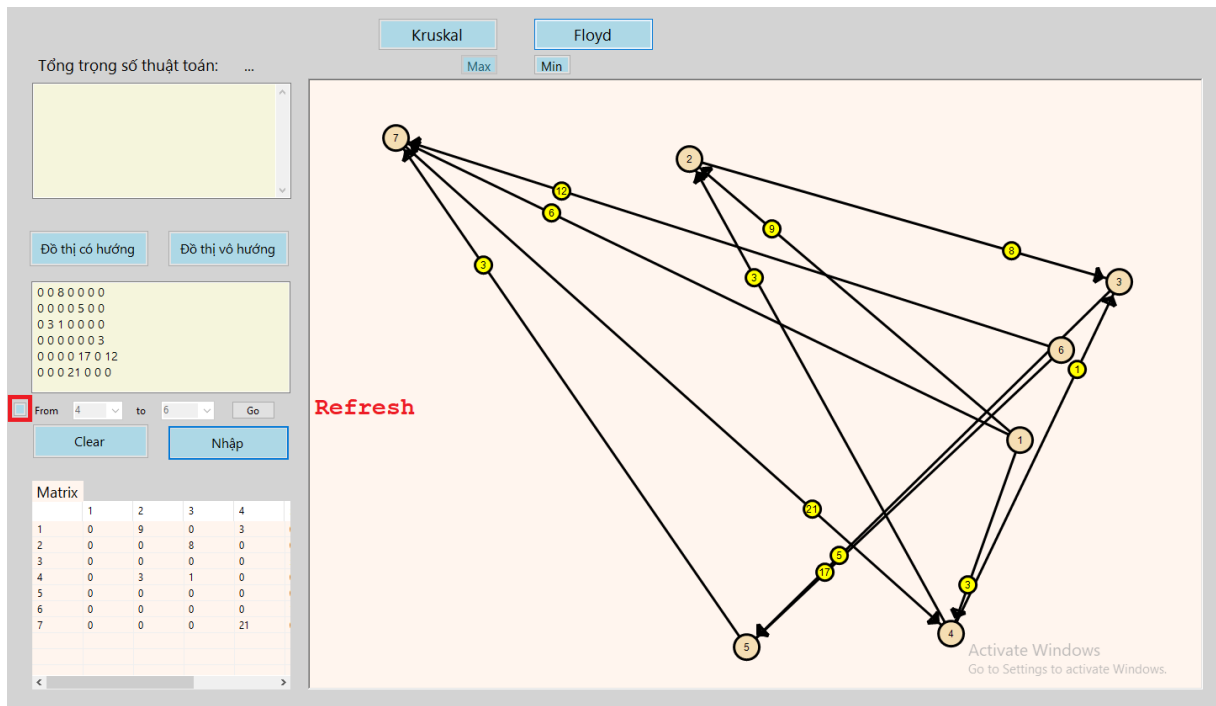
Tiếp theo, chúng ta sẽ xác định loại đồ thị.

- Nếu là đồ thị có hướng thì chọn vào ô “Đồ thị có hướng”.
- Nếu là đồ thị vô hướng thì chọn vào ô “Đồ thị vô hướng”.

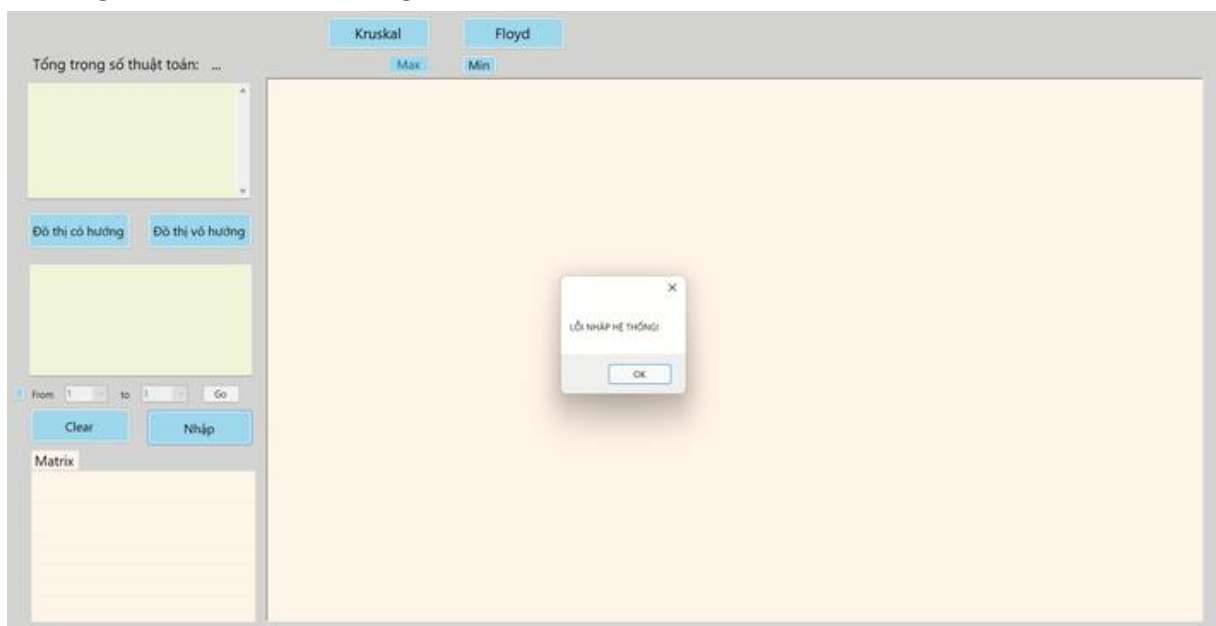
Sau khi xác định loại đồ thị, chúng ta sẽ nhấn vào ô “Nhập” để đồ thị được hiện ra.



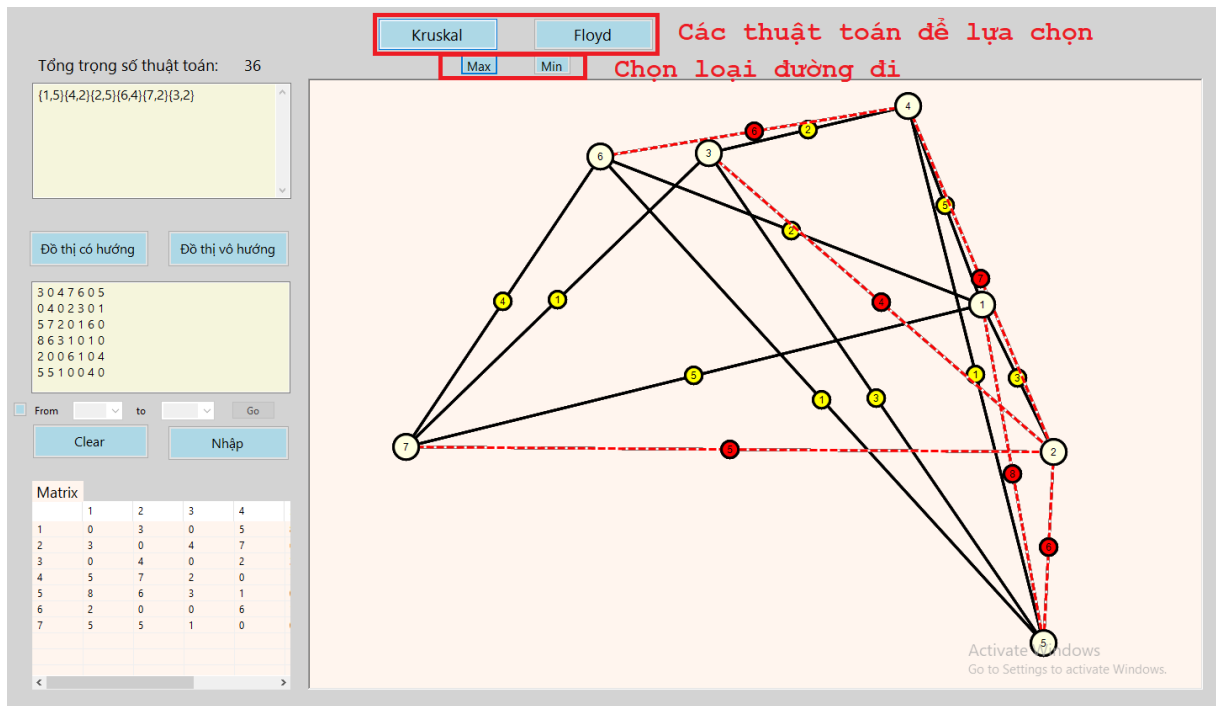
Nếu đồ thị hiện ra không đúng như ý muốn hoặc đồ thị bị được bắt mắt, chúng ta có thể vẽ lại đồ thị bằng cách chọn vào ô Refresh.



Lưu ý: trường hợp nhập sai hoặc thiếu dữ liệu ban đầu sau khi nhấn “Nhập”, chương trình sẽ hiện ra thông báo “LỖI NHẬP HỆ THỐNG”.



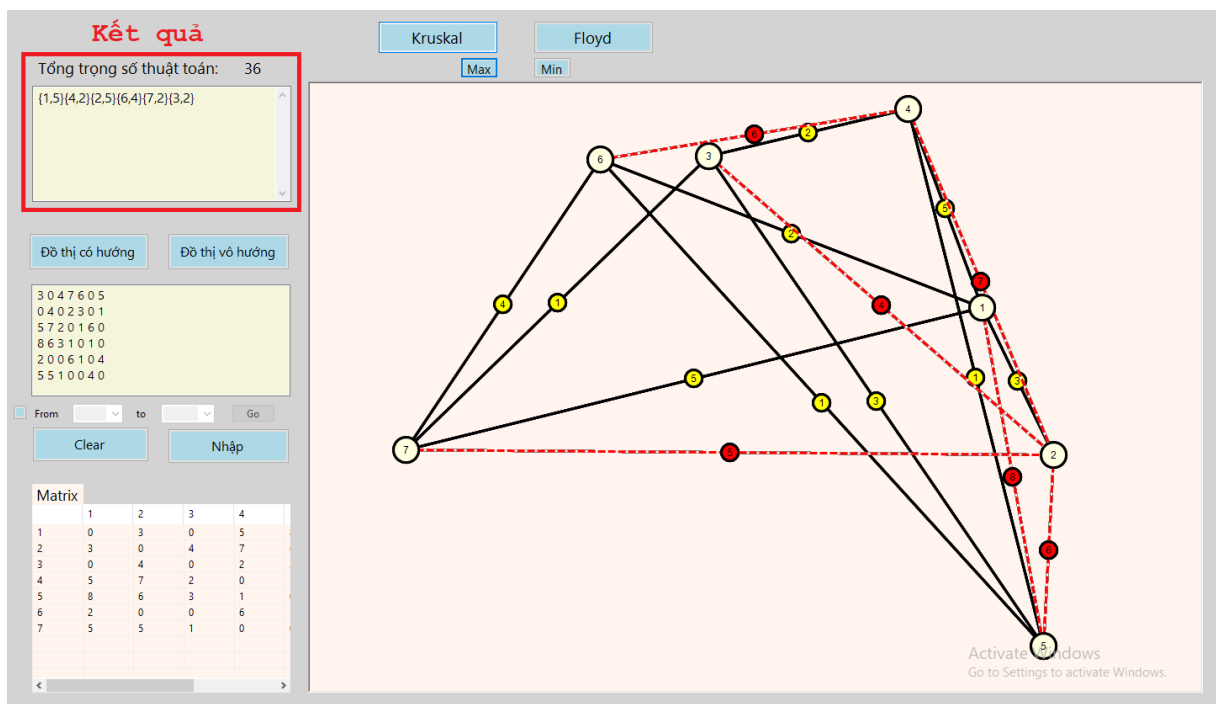
Sau khi chọn xong dạng đồ thị, chúng ta sẽ chọn tới thuật toán. Có hai thuật toán để chúng ta lựa chọn: Thuật toán Kruskal và Thuật toán Floyd.



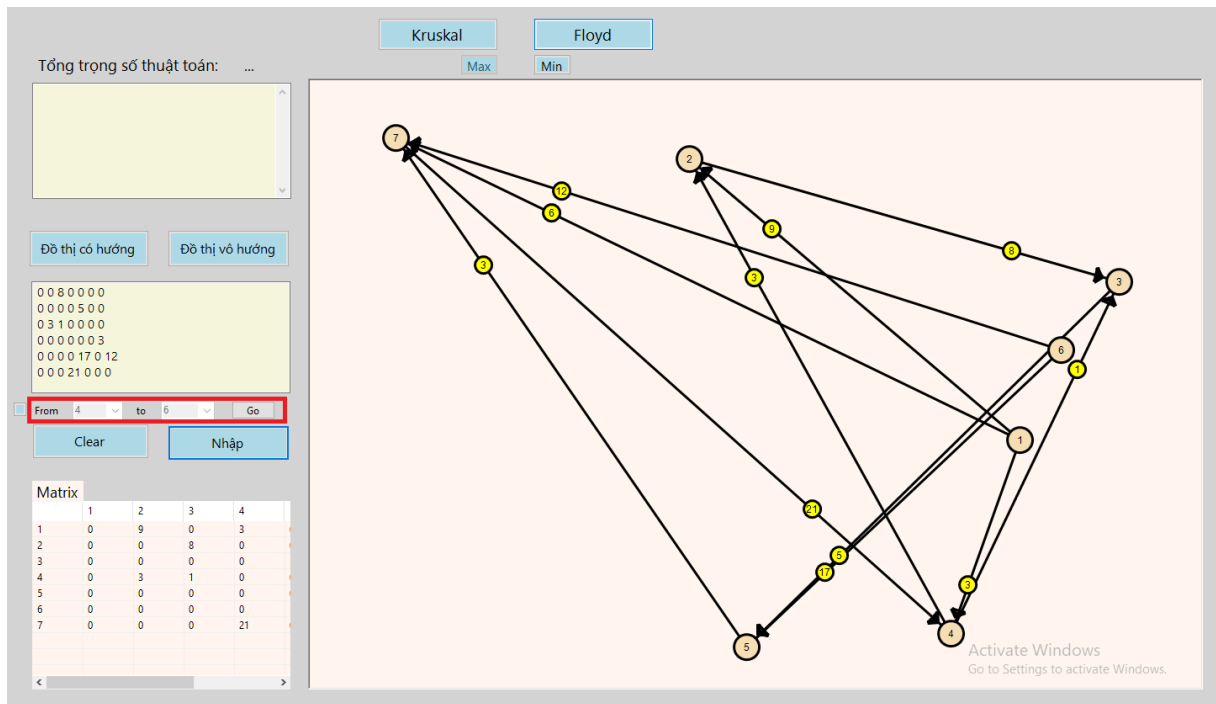
Chú ý:

- Nếu chọn thuật toán Kruskal, chúng ta có thể tìm cây khung lớn nhất bằng cách nhấn vào ô “Max” hoặc tìm cây khung nhỏ nhất bằng cách nhấn vào “Min”.
- Nếu chọn thuật toán Floyd, chúng ta chỉ có thể tìm được đường đi ngắn nhất bằng cách chọn vào ô “Min”.

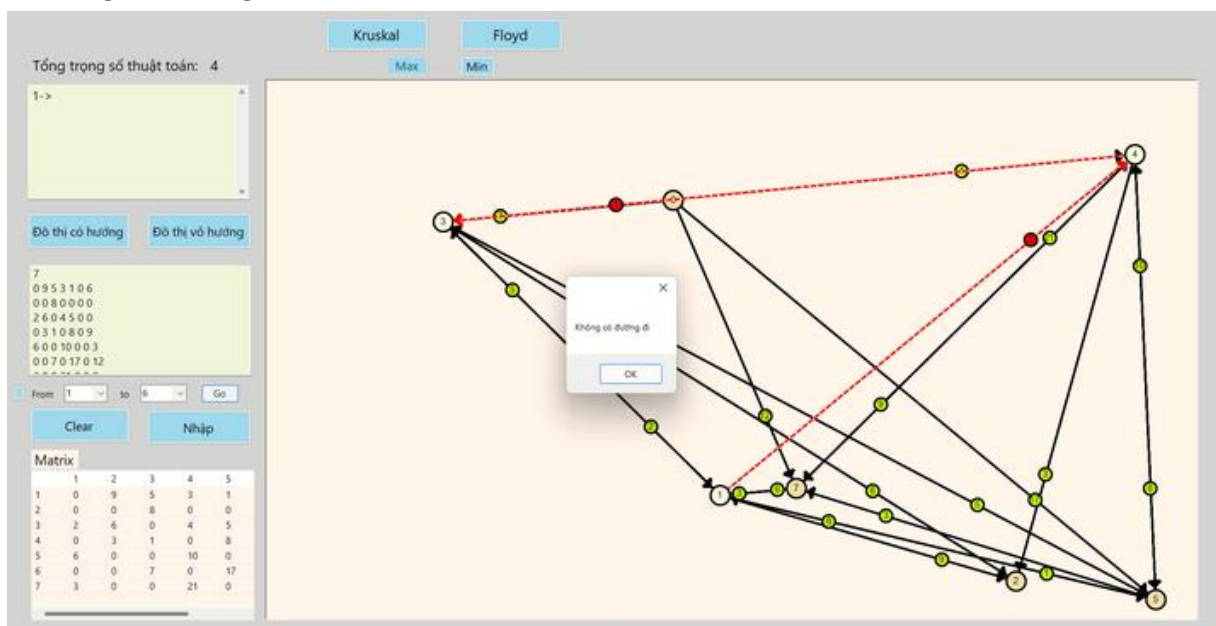
Sau khi chọn được loại cần tìm, chương trình sẽ hiện ra kết quả.



Trường hợp muốn chọn đường đi từ đỉnh A đến đỉnh B, chúng ta sẽ nhập dữ liệu đỉnh bắt đầu “From” và đỉnh cuối “To”, sau đó nhấn ô “Go” để hiện ra đường đi.



Lưu ý: Nếu chọn đường đi không đúng, hệ thống sẽ hiện lên hộp thoại thông báo “Không có đường đi”.



4.4. Mã nguồn

Link mã nguồn và video demo:

https://drive.google.com/drive/folders/1P-ntPSsjzDBibZVGxqtYYBj_rc9Uud1p

Ghi chú: Thầy down file về nhớ thay đổi vị trí file debug giúp em nha thầy. Chúng em cảm ơn thầy nhiều.

TÀI LIỆU THAM KHẢO

[1]. Thuật toán Floyd – Tìm đường đi ngắn nhất giữa mọi cặp đỉnh

<https://vietcodes.github.io/algo/floyd>

[2]. Thuật toán Kruskal - Tìm cây khung nhỏ nhất

<https://vietcodes.github.io/algo/kruskal>

[3]. Thuật toán Kuskal

https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n_Kruskal

[4]. Thuật toán Floyd

https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n_Floyd%E2%80%93Warshall

[5]. Tham khảo code

<https://github.com/dbvan/BFS>