

CT175-Lý thuyết đồ thị-HK1-2025-2026

Bảng Điều khiển / Các khoá học của tôi / CT175HK1_2025_2026

/ Tuần 3 - Tính liên thông của đồ thị

/ * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ngẫu nhiên)

Bắt đầu vào lúc	Saturday, 4 October 2025, 12:02 AM
Trạng thái	Đã xong
Kết thúc lúc	Saturday, 4 October 2025, 12:11 AM
Thời gian thực hiện	8 phút 53 giây
Điểm	0,98/1,00
Điểm	9,78 trên 10,00 (98%)

Câu hỏi 1

Đúng

Đạt điểm 0,98 trên 1,00

🚩 Đặt cờ

Thuật toán Tarjan tìm các thành phần liên thông mạnh (Tarjan's strongly connected components algorithm) dựa trên thuật toán duyệt đồ thị theo chiều sâu (sử dụng đệ quy) kết hợp với đánh số các đỉnh và một ngăn xếp lưu trữ các thành phần liên thông mạnh (strongly connected components gọi tắt là SCC).

```
void SCC(int u) {
    //1. Đánh số cho đỉnh u, đưa u vào ngăn xếp
    num[u] = k; min_num[u] = k; k++;
    push(s, u); on_stack[u] = true;

    //2. Lần lượt xét các đỉnh kề của u
    for (v là các đỉnh kề của u)
```

```

if (v chưa duyệt) {
    //2a. Duyệt v và cập nhật lại min_num[u]
    SCC(v);
    min_num[u] = min(min_num[u], min_num[v]);
} else if (v còn trên stack) {
    //2b. cập nhật lại min_num[u]
    min_num[u] = min(min_num[u], num[v]);
} else {
    //2c. bỏ qua, không làm gì cả
}

//3. Kiểm tra num[u] == min_num[u]
if (num[u] == min_num[u]) {
    //Lấy các đỉnh trong stack ra cho đến khi gặp u
    //Các đỉnh này thuộc về một thành phần liên thông
}
}

```

Thuật toán trên gồm 3 bước chính:

1. Đánh số cho đỉnh u, đưa u vào ngăn xếp. Sau khi đánh số xong tăng k lên.
Thông thường, k được khởi tạo bằng 1.
2. Xét các đỉnh kề v của u, có 3 trường hợp xảy ra:
 - v chưa duyệt => gọi đệ quy duyệt v và cập nhật lại min_num[u]
 - v duyệt rồi nhưng vẫn còn trên stack => cập nhật lại min_num[u]
 - v duyệt rồi và không còn trên stack => bỏ qua
3. Kiểm tra num[u] và min_num[u]: nếu bằng nhau thì u là đỉnh khớp (articulation vertex) hay đỉnh cắt (cut vertex)
 - Lần lượt lấy các đỉnh trong stack ra cho đến khi u được lấy ra. Các đỉnh được lấy ra này chính là thành phần liên thông mạnh chứa u.

Cho đồ thị **có hướng** gồm **6** đỉnh và **9** cung như bên như bên dưới. Hãy áp dụng thuật toán Tarjan để tìm các thành phần liên thông mạnh của G bắt đầu từ đỉnh **E**.

Dựa vào kết quả của thuật toán, vẽ các thành phần liên thông tìm được. Với mỗi thành phần liên thông, vẽ các đỉnh và các cung bên trong thành phần liên thông này. Nói cách khác, xoá bỏ các cung của đồ thị gốc mà 2 đỉnh của nó nằm ở 2 thành phần liên thông khác nhau.

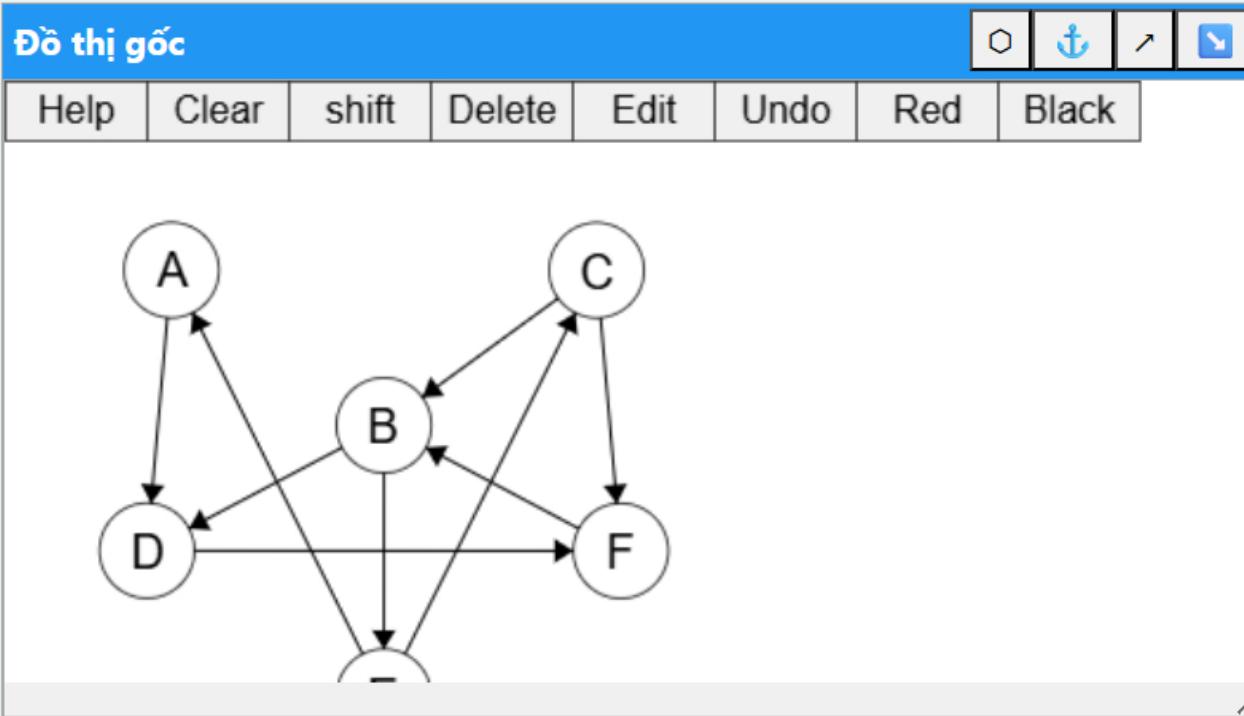
Quy ước

- Mỗi thể hiện của hàm **SCC(u)** được minh họa bằng một khung chữ nhật. Việc gọi đệ quy được minh họa bằng một hình chữ nhật nhỏ hơn bên trong.

- Hãy làm bài theo thứ tự từ ngoài vào trong và từ trên xuống dưới.
- Nếu làm sai 1 bước, có thể bấm "**Lùi lại 1 bước**" để làm lại bước trước đó.
- Liệt kê các đỉnh theo thứ tự **A, B, C, ...**
- **k được khởi tạo = 1.**

Answer: (penalty regime: 10, 20, ... %)

Reset answer



Áp dụng thuật toán Tarjan bằng cách duyệt đệ quy theo chiều sâu bắt đầu từ đỉnh E

⌘ Lùi lại 1 bước Số bước: **37**

SCC(E)

1. Đánh số cho đỉnh E và đưa nó vào ngăn xếp S.

Gán num[E] = min_num[E] = k = ; k++;

Đưa E vào S. Kết quả: S =

1

2. Với các đỉnh kề v của E: **2**

2a. v = 'A' chưa duyệt **3**

SCC(A)

1. Đánh số cho đỉnh A và đưa nó vào ngăn xếp S.

Gán num[A] = min_num[A] = k = ; k++;

Đưa A vào S. Kết quả: S =

4

2. Với các đỉnh kề v của A: D

5

2a. v = 'D' chưa duyệt 6

SCC(D)

1. Đánh số cho đỉnh D và đưa nó vào ngăn xếp S.

Gán num[D] = min_num[D] = k = ; k++;

Đưa D vào S. Kết quả: S =

7

2. Với các đỉnh kề v của D: F

8

2a. v = 'F' chưa duyệt 9

SCC(F)

1. Đánh số cho đỉnh F và đưa nó vào ngăn xếp S.

Gán num[F] = min_num[F] = k = ;
k++;

Đưa F vào S. Kết quả: S =

10

2. Với các đỉnh kề v của F:

B 11

2a. v = 'B' chưa duyệt 12

SCC(B)

1. Đánh số cho đỉnh B và đưa nó vào ngăn xếp S.

Gán num[B] = min_num[B] = k =

; k++;

Đưa B vào S. Kết quả: S =

Thực hiện đánh số và đưa vào Stack

13

2. Với các đỉnh kề v của B:

Xét

14

2b. v = 'D' duyệt rồi nhưng vẫn còn trên stack

15

Cập nhật min_num[B] =
min(min_num[B], num[D]) =

16

2b. v = 'E' duyệt rồi nhưng vẫn còn trên stack

17

Cập nhật min_num[B] =
min(min_num[B], num[E]) =

18

3. Kiểm tra num và min_num của B

X 19

Cập nhật min_num[F] = min(min_num[F],
min_num[B]) =

20

3. Kiểm tra num và min_num của F

num[F] == min_num[F]

num[F] != min_num[F] X 21

Cập nhật min_num[D] = min(min_num[D],

min_num[F]) = Cập nhật ✓ 22

3. Kiểm tra num và min_num của D

num[D] == min_num[D]

num[D] != min_num[D] X

23

Cập nhật min_num[A] = min(min_num[A], min_num[D]) =

1

Cập nhật

✓ 24

3. Kiểm tra num và min_num của A

num[A] == min_num[A]

num[A] != min_num[A] X

25

Cập nhật min_num[E] = min(min_num[E], min_num[A]) = 1

Cập nhật

✓ 26

2a. v = 'C' chưa duyệt Duyệt nó ✓ 27

SCC(C)



1. Đánh số cho đỉnh C và đưa nó vào ngăn xếp S.

Gán num[C] = min_num[C] = k = 6 ; k++;

Đưa C vào S. Kết quả: S = E,A,D,F,B,C

Thực hiện đánh số và đưa vào Stack

✓ 28

2. Với các đỉnh kề v của C: B,F

Xét

✓ 29

2b. v = 'B' duyệt rồi nhưng vẫn còn trên stack

Cập nhật min_num[u]

✓ 30

Cập nhật $\text{min_num}[C] = \min(\text{min_num}[C], \text{num}[B]) =$

5

Cập nhật

✓ 31

2b. $v = 'F'$ duyệt rồi nhưng vẫn còn trên stack

Cập nhật $\text{min_num}[u]$

✓ 32

Cập nhật $\text{min_num}[C] = \min(\text{min_num}[C], \text{num}[F]) =$

4

Cập nhật

✓ 33

3. Kiểm tra num và min_num của C

$\text{num}[C] == \text{min_num}[C]$

$\text{num}[C] != \text{min_num}[C]$

X 34

Cập nhật $\text{min_num}[E] = \min(\text{min_num}[E], \text{min_num}[C]) =$

1

Cập nhật

✓ 35

3. Kiểm tra num và min_num của E

$\text{num}[E] == \text{min_num}[E]$

$\text{num}[E] != \text{min_num}[E]$

✓ 36

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được E

Các đỉnh được lấy ra: C,B,F,D,A,E

Nội dung còn lại của ngăn xếp:

Cập nhật

✓ 37

Vẽ các thành phần liên thông

Help

Clear

shift

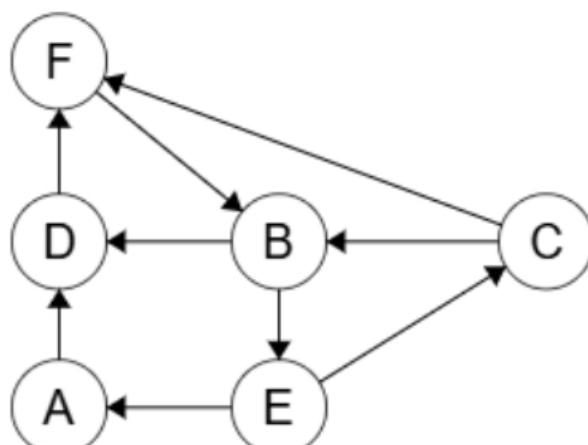
Delete

Edit

Undo

Red

Black



	Test	Got
✓	Test tự động	<p>1. Kiểm tra DFS</p> <ul style="list-style-type: none"> - Bước 1. SCC(E): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 1$ - Bước 2. SCC(E): 2. Xử lý các đỉnh kề của E - Bước 3. SCC(E): 2a. Duyệt A - Bước 4. SCC(A): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 2$ - Bước 5. SCC(A): 2. Xử lý các đỉnh kề của A - Bước 6. SCC(A): 2a. Duyệt D - Bước 7. SCC(D): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 3$ - Bước 8. SCC(D): 2. Xử lý các đỉnh kề của D - Bước 9. SCC(D): 2a. Duyệt F - Bước 10. SCC(F): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 4$ - Bước 11. SCC(F): 2. Xử lý các đỉnh kề của F - Bước 12. SCC(F): 2a. Duyệt B - Bước 13. SCC(B): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 5$ - Bước 14. SCC(B): 2. Xử lý các đỉnh kề của B - Bước 15. SCC(B): 2b. Cập nhật min_id của D - Bước 16. SCC(B): 2b. Cập nhật min_id của D - Bước 17. SCC(B): 2b. Cập nhật min_id của E - Bước 18. SCC(B): 2b. Cập nhật min_id của E - Bước 19. SCC(B): 3b. $id[u] \neq min_id[u]$ với $u = B$ - Bước 20. SCC(F): 2a. Cập nhật min_id của B - Bước 21. SCC(F): 3b. $id[u] \neq min_id[u]$ với $u = F$ - Bước 22. SCC(D): 2a. Cập nhật min_id của F - Bước 23. SCC(D): 3b. $id[u] \neq min_id[u]$ với $u = D$ - Bước 24. SCC(A): 2a. Cập nhật min_id của D - Bước 25. SCC(A): 3b. $id[u] \neq min_id[u]$ với $u = A$ - Bước 26. SCC(E): 2a. Cập nhật min_id của A - Bước 27. SCC(E): 2a. Duyệt C - Bước 28. SCC(C): 1. Đánh số và đưa u vào ngăn xếp, với $min_id[u] = 6$ - Bước 29. SCC(C): 2. Xử lý các đỉnh kề của C - Bước 30. SCC(C): 2b. Cập nhật min_id của B - Bước 31. SCC(C): 2b. Cập nhật min_id của B - Bước 32. SCC(C): 2b. Cập nhật min_id của F - Bước 33. SCC(C): 2b. Cập nhật min_id của F

- Bước 34. SCC(C): 3a. $\text{id}[u] \neq \text{min_id}[u]$ với $u = C$
 - Bước 35. SCC(E): 2a. Cập nhật min_id của C
 - Bước 36. SCC(E): 3a. $\text{id}[u] == \text{min_id}[u]$ với $u = E$
 - Bước 37. SCC(E): 3a. Tìm được SCC của E
2. Kiểm tra các thành phần liên thông
Các thành phần liên thông Ok.

Passed all tests! ✓

Đúng

Marks for this submission: 1,00/1,00. Accounting for previous tries, this gives **0,98/1,00**.

Hoàn thành việc xem lại

◀ * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ví dụ)

Chuyển tới...

* Tự học - DFS đệ quy toàn bộ đồ thị ►

Bảng câu hỏi

1
✓

Hoàn thành việc xem lại

Bạn đang đăng nhập với tên Duc Luu Chau Minh (Thoát)

CT175HK1 2025 2026

Vietnamese (vi)

English (en)

[Vietnamese \(vi\)](#)

[Data retention summary](#)

[Get the mobile app](#)

CT175-Lý thuyết đồ thị-HK1-2025-2026

Bảng Điều khiển / Các khoá học của tôi / CT175HK1_2025_2026

- / Tuần 3 - Tính liên thông của đồ thị
- / * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ngẫu nhiên)

Bắt đầu vào lúc	Friday, 3 October 2025, 10:20 PM
Trạng thái	Đã xong
Kết thúc lúc	Saturday, 4 October 2025, 12:01 AM
Thời gian thực hiện	1 giờ 41 phút
Điểm	0,92/1,00
Điểm	9,18 trên 10,00 (92%)

Câu hỏi 1

Đúng

Đạt điểm 0,92 trên 1,00

🚩 Đặt cờ

Thuật toán Tarjan tìm các thành phần liên thông mạnh (Tarjan's strongly connected components algorithm) dựa trên thuật toán duyệt đồ thị theo chiều sâu (sử dụng đệ quy) kết hợp với đánh số các đỉnh và một ngăn xếp lưu trữ các thành phần liên thông mạnh (strongly connected components gọi tắt là SCC).

```
void SCC(int u) {  
    //1. Đánh số cho đỉnh u, đưa u vào ngăn xếp  
    num[u] = k; min_num[u] = k; k++;  
    push(S, u); on_stack[u] = true;  
  
    //2. Lần lượt xét các đỉnh kề của u  
    for (v là các đỉnh kề của u)
```

```

if (v chưa duyệt) {
    //2a. Duyệt v và cập nhật lại min_num[u]
    SCC(v);
    min_num[u] = min(min_num[u], min_num[v]);
} else if (v còn trên stack) {
    //2b. cập nhật lại min_num[u]
    min_num[u] = min(min_num[u], num[v]);
} else {
    //2c. bỏ qua, không làm gì cả
}

//3. Kiểm tra num[u] == min_num[u]
if (num[u] == min_num[u]) {
    //Lấy các đỉnh trong stack ra cho đến khi gặp u
    //Các đỉnh này thuộc về một thành phần liên thông
}
}

```

Thuật toán trên gồm 3 bước chính:

1. Đánh số cho đỉnh u, đưa u vào ngăn xếp. Sau khi đánh số xong tăng k lên. Thông thường, k được khởi tạo bằng 1.
2. Xét các đỉnh kề v của u, có 3 trường hợp xảy ra:
 - v chưa duyệt => gọi đệ quy duyệt v và cập nhật lại min_num[u]
 - v duyệt rồi nhưng vẫn còn trên stack => cập nhật lại min_num[u]
 - v duyệt rồi và không còn trên stack => bỏ qua
3. Kiểm tra num[u] và min_num[u]: nếu bằng nhau thì u là đỉnh khớp (articulation vertex) hay đỉnh cắt (cut vertex)
 - Lần lượt lấy các đỉnh trong stack ra cho đến khi u được lấy ra. Các đỉnh được lấy ra này chính là thành phần liên thông mạnh chứa u.

Cho đồ thị **có hướng** gồm **6** đỉnh và **10** cung như bên dưới. Hãy áp dụng thuật toán Tarjan để tìm các thành phần liên thông mạnh của G bắt đầu từ đỉnh **E**.

Dựa vào kết quả của thuật toán, vẽ các thành phần liên thông tìm được. Với mỗi thành phần liên thông, vẽ các đỉnh và các cung bên trong thành phần liên thông này. Nói cách khác, xoá bỏ các cung của đồ thị gốc mà 2 đỉnh của nó nằm ở 2 thành phần liên thông khác nhau.

Quy ước

- Mỗi thể hiện của hàm **SCC(u)** được minh họa bằng một khung chữ nhật. Việc

gọi sẽ quy được minh họa bằng một hình ảnh như hình bên trong.

- Hãy làm bài theo thứ tự từ ngoài vào trong và từ trên xuống dưới.
- Nếu làm sai 1 bước, có thể bấm "**Lùi lại 1 bước**" để làm lại bước trước đó.
- Liệt kê các đỉnh theo thứ tự **A, B, C, ...**
- **k được khởi tạo = 1.**

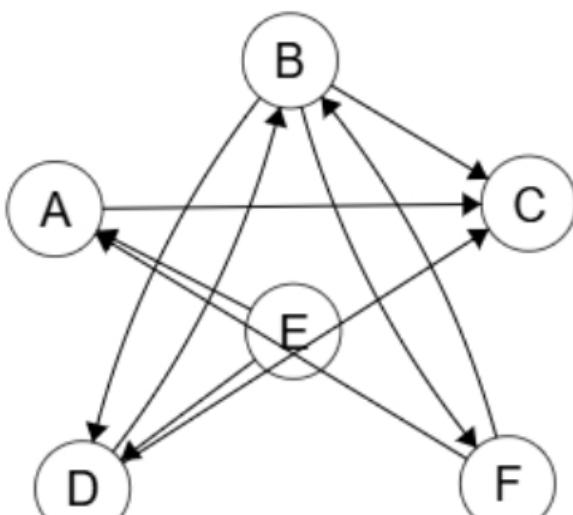
Answer: (penalty regime: 10, 20, ... %)

Reset answer

Đồ thị gốc



Help Clear shift Delete Edit Undo Red Black



Áp dụng thuật toán Tarjan bằng cách duyệt đệ quy theo chiều sâu bắt đầu từ đỉnh E

Số bước: **39**

SCC(E)

1. Đánh số cho đỉnh E và đưa nó vào ngăn xếp S.

Gán num[E] = min_num[E] = k = ; k++;

Đưa E vào S. Kết quả: S =

1

2. Với các đỉnh kề v của E: A,D **2**

2a. v = 'A' chưa duyệt **3**

SCC(A)

1. Đánh số cho đỉnh A và đưa nó vào ngăn xếp S.

Gán num[A] = min_num[A] = k = ; k++;Đưa A vào S. Kết quả: S =

Thực hiện đánh số và đưa vào Stack

 4

2. Với các đỉnh kề v của A: C

 5

2a. v = 'C' chưa duyệt

 6**SCC(C)**

1. Đánh số cho đỉnh C và đưa nó vào ngăn xếp S.

Gán num[C] = min_num[C] = k = ; k++;

Đưa C vào S. Kết quả: S =

Thực hiện đánh số và đưa vào Stack

 7

2. Với các đỉnh kề v của C:

 8

3. Kiểm tra num và min_num của C

9

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được C

Các đỉnh được lấy ra:

Nội dung còn lại của ngăn xếp:

 10

Cập nhật min_num[A] = min(min_num[A], min_num[C]) =

 11

3. Kiểm tra num và min_num của A

num[A] == min_num[A]

num[A] != min_num[A]

12

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được A

Các đỉnh được lấy ra: A

Nội dung còn lại của ngăn xếp:

E

Cập nhật 13

Cập nhật min_num[E] = min(min_num[E], min_num[A]) = 1

Cập nhật 14

2a. v = 'D' chưa duyệt Duyệt nó 15

SCC(D)



1. Đánh số cho đỉnh D và đưa nó vào ngăn xếp S.

Gán num[D] = min_num[D] = k = 4 ; k++;

Đưa D vào S. Kết quả: S = E,D

Thực hiện đánh số và đưa vào Stack 16

2. Với các đỉnh kề v của D: B,C

Xét 17

2a. v = 'B' chưa duyệt Duyệt nó 18

SCC(B)



1. Đánh số cho đỉnh B và đưa nó vào ngăn xếp S.

Gán num[B] = min_num[B] = k = 5 ; k++;

Đưa B vào S. Kết quả: S =

E,D,B

Thực hiện đánh số và đưa vào Stack 19

2. Với các đỉnh kề v của B: C,D,F

Xét

20

2c. v = 'C' duyệt rồi và không còn trên stack

X 21

2b. v = 'D' duyệt rồi nhưng vẫn còn trên stack

22

Cập nhật $\text{min_num}[B] = \min(\text{min_num}[B], \text{num}[D]) =$

4

23

2a. v = 'F' chưa duyệt 24

SCC(F)

1. Đánh số cho đỉnh F và đưa nó vào ngăn xếp S.

Gán $\text{num}[F] = \text{min_num}[F] = k = 6$;

$k++;$

Đưa F vào S. Kết quả: S =

E,D,B,F

25

2. Với các đỉnh kề v của F:

A,B

26

2c. v = 'A' duyệt rồi và không còn trên stack

X 27

2b. v = 'B' duyệt rồi nhưng vẫn còn trên stack

28

Cập nhật $\text{min_num}[F] = \min(\text{min_num}[F], \text{num}[B]) = 5$ 29

3. Kiểm tra num và min_num của F

num[F] != min_num[F] X 30

Cập nhật min_num[B] = min(min_num[B],

min_num[F]) = Cập nhật ✓ 31

3. Kiểm tra num và min_num của B

num[B] == min_num[B]

num[B] != min_num[B]

X 32

Cập nhật min_num[D] = min(min_num[D], min_num[B]) =

Cập nhật ✓ 33

2c. v = 'C' duyệt rồi và không còn trên stack Bỏ qua X 34

3. Kiểm tra num và min_num của D

num[D] == min_num[D]

num[D] != min_num[D]

✓ 35

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được D

Các đỉnh được lấy ra:

Nội dung còn lại của ngăn xếp:

Cập nhật ✓ 36

Cập nhật min_num[E] = min(min_num[E], min_num[D]) =

Cập nhật ✓ 37

3. Kiểm tra num và min_num của E

num[E] == min_num[E]

num[E] != min_num[E]

✓ 38

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được E

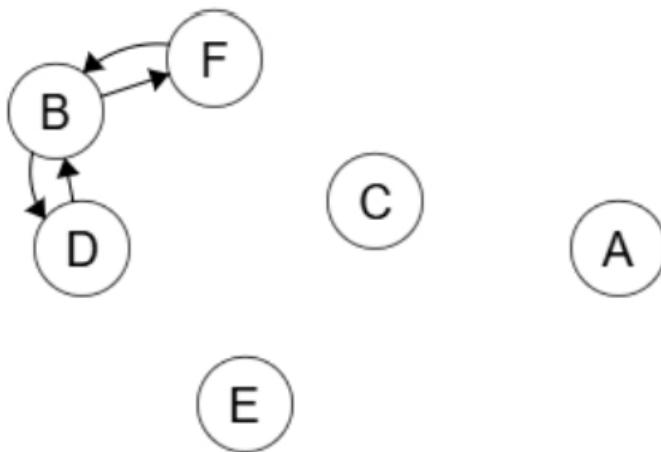
Các đỉnh được lấy ra:

Nội dung còn lại của ngăn xếp:

Cập nhật ✓ 39

Vẽ các thành phần liên thông

Help Clear shift Delete Edit Undo Red Black



	Test	Got
✓	Test tự động	<p>1. Kiểm tra DFS</p> <ul style="list-style-type: none">- Bước 1. SCC(E): 1. Đánh số và đưa u vào ngăn xếp, với $u = E$- Bước 2. SCC(E): 2. Xử lý các đỉnh kề của E- Bước 3. SCC(E): 2a. Duyệt A- Bước 4. SCC(A): 1. Đánh số và đưa u vào ngăn xếp, với $u = A$- Bước 5. SCC(A): 2. Xử lý các đỉnh kề của A- Bước 6. SCC(A): 2a. Duyệt C- Bước 7. SCC(C): 1. Đánh số và đưa u vào ngăn xếp, với $u = C$- Bước 8. SCC(C): 2. Xử lý các đỉnh kề của C- Bước 9. SCC(C): 3a. $id[u] == min_id[u]$ với $u = C$- Bước 10. SCC(C): 3a. Tìm được SCC của C- Bước 11. SCC(A): 2a. Cập nhật min_id của C- Bước 12. SCC(A): 3a. $id[u] == min_id[u]$ với $u = A$- Bước 13. SCC(A): 3a. Tìm được SCC của A- Bước 14. SCC(E): 2a. Cập nhật min_id của A- Bước 15. SCC(E): 2a. Duyệt D- Bước 16. SCC(D): 1. Đánh số và đưa u vào ngăn xếp, với $u = D$- Bước 17. SCC(D): 2. Xử lý các đỉnh kề của D- Bước 18. SCC(D): 2a. Duyệt B- Bước 19. SCC(B): 1. Đánh số và đưa u vào ngăn xếp, với $u = B$

- Bước 20. SCC(B): 2. Xử lý các đỉnh kề của B
 - Bước 21. SCC(B): 2c. Bỏ qua đỉnh C
 - Bước 22. SCC(B): 2b. Cập nhật \min_id của D
 - Bước 23. SCC(B): 2b. Cập nhật \min_id của D
 - Bước 24. SCC(B): 2a. Duyệt F
 - Bước 25. SCC(F): 1. Đánh số và đưa u vào ngăn xếp, với $\min_id[u] = \text{id}[u]$
 - Bước 26. SCC(F): 2. Xử lý các đỉnh kề của F
 - Bước 27. SCC(F): 2c. Bỏ qua đỉnh A
 - Bước 28. SCC(F): 2b. Cập nhật \min_id của B
 - Bước 29. SCC(F): 2b. Cập nhật \min_id của B
 - Bước 30. SCC(F): 3b. $\text{id}[u] != \min_id[u]$ với $u = F$
 - Bước 31. SCC(B): 2a. Cập nhật \min_id của F
 - Bước 32. SCC(B): 3b. $\text{id}[u] != \min_id[u]$ với $u = B$
 - Bước 33. SCC(D): 2a. Cập nhật \min_id của B
 - Bước 34. SCC(D): 2c. Bỏ qua đỉnh C
 - Bước 35. SCC(D): 3a. $\text{id}[u] == \min_id[u]$ với $u = D$
 - Bước 36. SCC(D): 3a. Tìm được SCC của D
 - Bước 37. SCC(E): 2a. Cập nhật \min_id của D
 - Bước 38. SCC(E): 3a. $\text{id}[u] == \min_id[u]$ với $u = E$
 - Bước 39. SCC(E): 3a. Tìm được SCC của E
2. Kiểm tra các thành phần liên thông
- Các thành phần liên thông Ok.

Passed all tests! ✓

Đúng

Marks for this submission: 1,00/1,00. Accounting for previous tries, this gives **0,92/1,00**.

Hoàn thành việc xem lại

◀ * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ví dụ)

Chuyển tới...



* Tự học - DFS đệ quy toàn bộ đồ thị ►

Bảng câu hỏi

1
✓

[Hoàn thành việc xem lại](#)

Bạn đang đăng nhập với tên Duc Luu Chau Minh (Thoát)
CT175HK1_2025_2026

[Vietnamese \(vi\)](#)

[English \(en\)](#)

[Vietnamese \(vi\)](#)

[Data retention summary](#)

[Get the mobile app](#)

CT175-Lý thuyết đồ thị-HK1-2025-2026

Bảng Điều khiển / Các khoá học của tôi / CT175HK1_2025_2026 / Tuần 3 - Tính liên thông của đồ thị
 / * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ngẫu nhiên)

Bắt đầu vào lúc	Thursday, 2 October 2025, 4:41 PM
Trạng thái	Đã xong
Kết thúc lúc	Friday, 3 October 2025, 2:58 PM
Thời gian thực hiện	22 giờ 17 phút
Điểm	1,00/1,00
Điểm	10,00 trên 10,00 (100%)

Câu hỏi 1
Đúng
Đạt điểm 1,00 trên 1,00
▼ Đặt cờ

Thuật toán Tarjan tìm các thành phần liên thông mạnh (Tarjan's strongly connected components algorithm) dựa trên thuật toán duyệt đồ thị theo chiều sâu (sử dụng đệ quy) kết hợp với đánh số các đỉnh và một ngăn xếp lưu trữ các thành phần liên thông mạnh (strongly connected components) gọi tắt là SCC.

```
void SCC(int u) {
    //1. Đánh số cho đỉnh u, đưa u vào ngăn xếp
    num[u] = k; min_num[u] = k; k++;
    push(S, u); on_stack[u] = true;

    //2. Lần lượt xét các đỉnh kề của u
    for (v là các đỉnh kề của u)
        if (v chưa duyệt) {
            //2a. Duyệt v và cập nhật lại min_num[u]
            SCC(v);
            min_num[u] = min(min_num[u], min_num[v]);
        } else if (v còn trên stack) {
            //2b. Cập nhật lại min_num[u]
            min_num[u] = min(min_num[u], num[v]);
        } else {
            //2c. bỏ qua, không làm gì cả
        }
}

//3. Kiểm tra num[u] == min_num[u]
if (num[u] == min_num[u]) {
    //Lấy các đỉnh trong stack ra cho đến khi gặp u
    //Các đỉnh này thuộc về một thành phần liên thông
}
}
```

Thuật toán trên gồm 3 bước chính:

1. Đánh số cho đỉnh u, đưa u vào ngăn xếp. Sau khi đánh số xong tăng k lên. Thông thường, k được khởi tạo bằng 1.
2. Xét các đỉnh kề v của u, có 3 trường hợp xảy ra:
 - v chưa duyệt => gọi đệ quy duyệt v và cập nhật lại min_num[u]
 - v duyệt rồi nhưng vẫn còn trên stack => cập nhật lại min_num[u]
 - v duyệt rồi và không còn trên stack => bỏ qua
3. Kiểm tra num[u] và min_num[u]: nếu bằng nhau thì u là đỉnh khớp (articulation vertex) hay đỉnh cắt (cut vertex)
 - Lần lượt lấy các đỉnh trong stack ra cho đến khi u được lấy ra. Các đỉnh được lấy ra này chính là thành phần liên thông chứa u.

Cho đồ thị **có hướng** gồm **7** đỉnh và **10** cung như bên dưới. Hãy áp dụng thuật toán Tarjan để tìm các thành phần liên thông mạnh của G bắt đầu từ đỉnh **5**.

Dựa vào kết quả của thuật toán, vẽ các thành phần liên thông tìm được. Với mỗi thành phần liên thông, vẽ các đỉnh và các cung bên trong thành phần liên thông này. Nói cách khác, xoá bỏ các cung của đồ thị gốc mà 2 đỉnh của nó nằm ở 2 thành phần liên thông khác nhau.

Quy ước

- Mỗi thể hiện của hàm **SCC(u)** được minh họa bằng một khung chữ nhật. Việc gọi đệ quy được minh họa bằng một hình chữ nhật nhỏ hơn bên trong.
- Hãy làm bài theo thứ tự từ ngoài vào trong và từ trên xuống dưới.
- Nếu làm sai 1 bước, có thể bấm "**Lùi lại 1 bước**" để làm lại bước trước đó.
- Liệt kê các đỉnh theo thứ tự **1, 2, 3, ...**
- **k được khởi tạo = 1.**

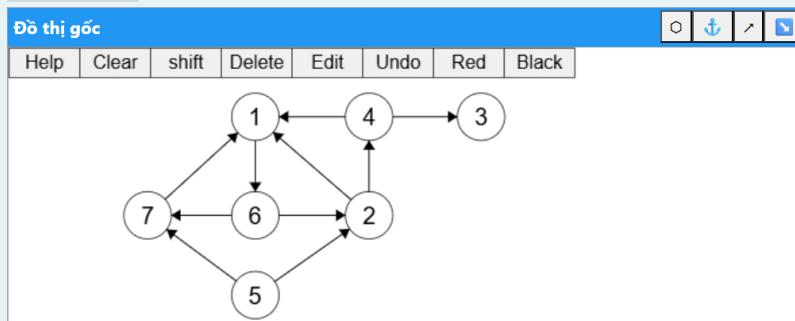
Answer: (penalty regime: 10, 20, ... %)

Reset answer

Bảng câu hỏi

1
✓

Hoàn thành việc xem lại



Áp dụng thuật toán Tarjan bằng cách duyệt đệ quy theo chiều sâu bắt đầu từ đỉnh 5

Số bước: 43

SCC(5)

1. Đánh số cho đỉnh 5 và đưa nó vào ngăn xếp S.

Gán num[5] = min_num[5] = k = ; k++;

Đưa 5 vào S. Kết quả: S =

1

2. Với các đỉnh kề v của 5: 2,7 2

2a. v = '2' chưa duyệt 3

SCC(2)

1. Đánh số cho đỉnh 2 và đưa nó vào ngăn xếp S.

Gán num[2] = min_num[2] = k = ; k++;

Đưa 2 vào S. Kết quả: S =

4

2. Với các đỉnh kề v của 2: 1,4 5

2a. v = '1' chưa duyệt 6

SCC(1)

1. Đánh số cho đỉnh 1 và đưa nó vào ngăn xếp S.

Gán num[1] = min_num[1] = k = ; k++;

Đưa 1 vào S. Kết quả: S =

7

2. Với các đỉnh kề v của 1: 6 8

2a. v = '6' chưa duyệt 9

SCC(6)

1. Đánh số cho đỉnh 6 và đưa nó vào ngăn xếp S.

Gán num[6] = min_num[6] = k = ; k++;

Đưa 6 vào S. Kết quả: S =

10

2. Với các đỉnh kề v của 6: 2,7 11

2b. v = '2' duyệt rồi nhưng vẫn còn trên stack

12

Cập nhật min_num[6] = min(min_num[6], num[2]) =

13

2a. v = '7' chưa duyệt 14

1. Đánh số cho đỉnh 7 và đưa nó vào ngăn xếp S.

Gán num[7] = min_num[7] = k = ; k++;

Đưa 7 vào S. Kết quả: S =

5,2,1,b,/

Thực hiện đánh số và đưa vào Stack ✓ 15

2. Với các đỉnh kề v của 7: 1 Xét

✓ 16

2b. v = '1' duyệt rồi nhưng vẫn còn trên stack

Cập nhật min_num[u] ✓ 17

Cập nhật min_num[7] = min(min_num[7], num[1]) =

3. Kiểm tra num và min_num của 7

num[7] == min_num[7] num[7] != min_num[7] X 19

Cập nhật min_num[6] = min(min_num[6], min_num[7]) =

2 Cập nhật ✓ 20

3. Kiểm tra num và min_num của 6

num[6] == min_num[6] num[6] != min_num[6] X 21

Cập nhật min_num[1] = min(min_num[1], min_num[6]) = 2

Cập nhật ✓ 22

num[1] == min_num[1] num[1] != min_num[1] X 23

Cập nhật min_num[2] = min(min_num[2], min_num[1]) = 2 Cập nhật

✓ 24

2a. v = '4' chưa duyệt Duyệt nó ✓ 25

SCC(4) ←→ →←

1. Đánh số cho đỉnh 4 và đưa nó vào ngăn xếp S.

Gán num[4] = min_num[4] = k = 6 ; k++;

Đưa 4 vào S. Kết quả: S = 5,2,1,6,7,4

Thực hiện đánh số và đưa vào Stack ✓ 26

2. Với các đỉnh kề v của 4: 1,3 Xét ✓ 27

2b. v = '1' duyệt rồi nhưng vẫn còn trên stack Cập nhật min_num[u]

✓ 28

Cập nhật min_num[4] = min(min_num[4], num[1]) = 3

Cập nhật ✓ 29

2a. v = '3' chưa duyệt Duyệt nó ✓ 30

SCC(3) ←→ →←

1. Đánh số cho đỉnh 3 và đưa nó vào ngăn xếp S.

Đưa 3 vào S. Kết quả: S = 5,2,1,6,7,4,3

Thực hiện đánh số và đưa vào Stack ✓ 31

2. Với các đỉnh kề v của 3: Xét ✓ 32

3. Kiểm tra num và min_num của 3

num[3] == min_num[3] num[3] != min_num[3] ✓ 33

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 3

Các đỉnh được lấy ra: 3

Nội dung còn lại của ngăn xếp:

5,2,1,6,7,4

Cập nhật min_num[4] = min(min_num[4], min_num[3]) = 3

Cập nhật ✓ 35

3. Kiểm tra num và min_num của 4

num[4] == min_num[4] num[4] != min_num[4] X 36

Cập nhật min_num[2] = min(min_num[2], min_num[4]) = 2 Cập nhật

✓ 37

3. Kiểm tra num và min_num của 2

num[2] == min_num[2] num[2] != min_num[2] ✓ 38

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 2

Các đỉnh được lấy ra: 4,7,6,1,2

Nội dung còn lại của ngăn xếp: 5

Cập nhật ✓ 39

Cập nhật min_num[5] = min(min_num[5], min_num[2]) = 1 Cập nhật ✓ 40

2c. v = '7' duyệt rồi và không còn trên stack Bỏ qua X 41

3. Kiểm tra num và min_num của 5

num[5] == min_num[5] num[5] != min_num[5] ✓ 42

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 5

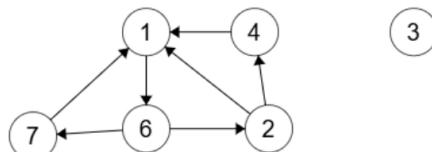
Các đỉnh được lấy ra: 5

Nội dung còn lại của ngăn xếp:

Cập nhật ✓ 43

Vẽ các thành phần liên thông

Help Clear shift Delete Edit Undo Red Black



(5)

Test	Got
✓ Test tự động	1. Kiểm tra DFS - Bước 1. SCC(5): 1. Đánh số và đưa u vào ngăn xếp, với u = 5 - Bước 2. SCC(5): 2. Xử lý các đỉnh kề của 5 - Bước 3. SCC(5): 2a. Duyệt 2 - Bước 4. SCC(2): 1. Đánh số và đưa u vào ngăn xếp, với u = 2 - Bước 5. SCC(2): 2. Xử lý các đỉnh kề của 2 - Bước 6. SCC(2): 2a. Duyệt 1 - Bước 7. SCC(1): 1. Đánh số và đưa u vào ngăn xếp, với u = 1 - Bước 8. SCC(1): 2. Xử lý các đỉnh kề của 1 - Bước 9. SCC(1): 2a. Duyệt 6 - Bước 10. SCC(6): 1. Đánh số và đưa u vào ngăn xếp, với u = 6 - Bước 11. SCC(6): 2. Xử lý các đỉnh kề của 6 - Bước 12. SCC(6): 2b. Cập nhật min_id của 2 - Bước 13. SCC(6): 2b. Cập nhật min_id của 2 - Bước 14. SCC(6): 2a. Duyệt 7 - Bước 15. SCC(7): 1. Đánh số và đưa u vào ngăn xếp, với u = 7 - Bước 16. SCC(7): 2. Xử lý các đỉnh kề của 7 - Bước 17. SCC(7): 2b. Cập nhật min_id của 1 - Bước 18. SCC(7): 2b. Cập nhật min_id của 1 - Bước 19. SCC(7): 3b. id[u] != min_id[u] với u = 7 - Bước 20. SCC(6): 2a. Cập nhật min_id của 7 - Bước 21. SCC(6): 3b. id[u] != min_id[u] với u = 6 - Bước 22. SCC(1): 2a. Cập nhật min_id của 6 - Bước 23. SCC(1): 3b. id[u] != min_id[u] với u = 1 - Bước 24. SCC(2): 2a. Cập nhật min_id của 1 - Bước 25. SCC(2): 2a. Duyệt 4 - Bước 26. SCC(4): 1. Đánh số và đưa u vào ngăn xếp, với u = 4 - Bước 27. SCC(4): 2. Xử lý các đỉnh kề của 4

- Bước 28. SCC(4): 2b. Cập nhật `min_id` của 1
- Bước 29. SCC(4): 2b. Cập nhật `min_id` của 1
- Bước 30. SCC(4): 2a. Duyệt 3
- Bước 31. SCC(3): 1. Đánh số và đưa u vào ngăn xếp, với $u = 3$
- Bước 32. SCC(3): 2. Xử lý các đỉnh kề của 3
- Bước 33. SCC(3): 3a. $id[u] == min_id[u]$ với $u = 3$
- Bước 34. SCC(3): 3a. Tìm được SCC của 3
- Bước 35. SCC(4): 2a. Cập nhật `min_id` của 3
- Bước 36. SCC(4): 3b. $id[u] != min_id[u]$ với $u = 4$
- Bước 37. SCC(2): 2a. Cập nhật `min_id` của 4
- Bước 38. SCC(2): 3a. $id[u] == min_id[u]$ với $u = 2$
- Bước 39. SCC(2): 3a. Tìm được SCC của 2

- Bước 41. SCC(5): 2c. Bỏ qua đỉnh 7
- Bước 42. SCC(5): 3a. $id[u] == min_id[u]$ với $u = 5$
- Bước 43. SCC(5): 3a. Tìm được SCC của 5
2. Kiểm tra các thành phần liên thông
Các thành phần liên thông Ok.

Passed all tests! ✓

Đúng

Marks for this submission: 1,00/1,00.

Hoàn thành việc xem lại

◀ * Tự học - Áp dụng thuật toán
Tarjan tìm các BPLT của đồ thị có
hướng (ví dụ)

* Tự học - DFS đệ quy toàn bộ
đồ thị ►

Chuyển tới...

Bạn đang đăng nhập với tên [Duc Luu Chau Minh \(Thoát\)](#)

[CT175HK1_2025_2026](#)

[Data retention summary](#)

[Get the mobile app](#)

CT175-Lý thuyết đồ thị-HK1-2025-2026

Bảng Điều khiển / Các khóa học của tôi / CT175HK1_2025_2026 / Tuần 3 - Tính liên thông của đồ thị / * Tự học - Áp dụng thuật toán Tarjan tìm các BPLT của đồ thị có hướng (ví dụ)

Bắt đầu vào lúc	Thursday, 2 October 2025, 3:11 PM
Trạng thái	Đã xong
Kết thúc lúc	Friday, 3 October 2025, 5:31 AM
Thời gian thực hiện	14 giờ 19 phút
Điểm	0,90/1,00
Điểm	9,00 trên 10,00 (90%)

Câu hỏi 1
Đúng
Đạt điểm 0,90 trên 1,00
Đặt cờ

Thuật toán Tarjan tìm các thành phần liên thông mạnh (Tarjan's strongly connected components algorithm) dựa trên thuật toán duyệt đồ thị theo chiều sâu (sử dụng đệ quy) kết hợp với đánh số các đỉnh và một ngăn xếp lưu trữ các thành phần liên thông mạnh (strongly connected components) gọi tắt là SCC.

```
void SCC(int u) {
    //1. Đánh số cho đỉnh u, đưa u vào ngăn xếp
    num[u] = k; min_num[u] = k; k++;
    push(S, u); on_stack[u] = true;

    //2. Lần lượt xét các đỉnh kề của u
    for (v là các đỉnh kề của u)
        if (v chưa duyệt) {
            //2a. Duyệt v và cập nhật lại min_num[u]
            SCC(v);
            min_num[u] = min(min_num[u], min_num[v]);
        } else if (v còn trên stack) {
            //2b. Cập nhật lại min_num[u]
            min_num[u] = min(min_num[u], num[v]);
        } else {
            //2c. bỏ qua, không làm gì cả
        }
}

//3. Kiểm tra num[u] == min_num[u]
if (num[u] == min_num[u]) {
    //Lấy các đỉnh trong stack ra cho đến khi gặp u
    //Các đỉnh này thuộc về một thành phần liên thông
}
}
```

Thuật toán trên gồm 3 bước chính:

- Đánh số cho đỉnh u, đưa u vào ngăn xếp. Sau khi đánh số xong tăng k lên. Thông thường, k được khởi tạo bằng 1.
- Xét các đỉnh kề v của u, có 3 trường hợp xảy ra:
 - v chưa duyệt => gọi đệ quy duyệt v và cập nhật lại min_num[u]
 - v duyệt rồi nhưng vẫn còn trên stack => cập nhật lại min_num[u]
 - v duyệt rồi và không còn trên stack => bỏ qua
- Kiểm tra num[u] và min_num[u]: nếu bằng nhau thì u là đỉnh khớp (articulation vertex) hay đỉnh cắt (cut vertex)
 - Lần lượt lấy các đỉnh trong stack ra cho đến khi u được lấy ra. Các đỉnh được lấy ra này chính là thành phần liên thông mạnh chứa u.

Cho đồ thị **có hướng** gồm **8** đỉnh và **14** cung như bên dưới. Hãy áp dụng thuật toán Tarjan để tìm các thành phần liên thông mạnh của G bắt đầu từ đỉnh **1**.

Dựa vào kết quả của thuật toán, vẽ các thành phần liên thông tìm được. Với mỗi thành phần liên thông, vẽ các đỉnh và các cung bên trong thành phần liên thông này. Nói cách khác, xoá bỏ các cung của đồ thị gốc mà 2 đỉnh của nó nằm ở 2 thành phần liên thông khác nhau.

Quy ước

- Mỗi thể hiện của hàm **SCC(u)** được minh họa bằng một khung chữ nhật. Việc gọi đệ quy được minh họa bằng một hình chữ nhật nhỏ hơn bên trong.
- Hãy làm bài theo thứ tự từ ngoài vào trong và từ trên xuống dưới.
- Nếu làm sai 1 bước, có thể bấm "**Lùi lại 1 bước**" để làm lại bước trước đó.
- Liệt kê các đỉnh theo thứ tự **1, 2, 3, ...**
- k được khởi tạo = 1.**

Answer: (penalty regime: 10, 20, ... %)

Reset answer

Bảng câu hỏi

1
✓

Hoàn thành việc xem lại

Đồ thị gốc

Help | Clear | shift | Delete | Edit | Undo | Red | Black

Áp dụng thuật toán Tarjan bằng cách duyệt đệ quy theo chiều sâu bắt đầu từ đỉnh 1

Lùi lại 1 bước | Số bước: 52

SCC(1) [←→] [→←]

1. Đánh số cho đỉnh 1 và đưa nó vào ngăn xếp S.
 Gán num[1] = min_num[1] = k = 1 ; k++;
 Đưa 1 vào S. Kết quả: S = 1

Thực hiện đánh số và đưa vào Stack ✓ 1

2. Với các đỉnh kề v của 1: 2,3 Xét ✓ 2

2a. v = '2' chưa duyệt Duyệt nó ✓ 3

SCC(2) [←→] [→←]

1. Đánh số cho đỉnh 2 và đưa nó vào ngăn xếp S.
 Gán num[2] = min_num[2] = k = 2 ; k++;
 Đưa 2 vào S. Kết quả: S = 1,2

Thực hiện đánh số và đưa vào Stack ✓ 4

2. Với các đỉnh kề v của 2: 1,8 Xét ✓ 5

2b. v = '1' duyệt rồi nhưng vẫn còn trên stack Cập nhật min_num[u] ✓ 6
 Cập nhật min_num[2] = min(min_num[2], num[1]) = 1 Cập nhật ✓ 7

2a. v = '8' chưa duyệt Duyệt nó ✓ 8

SCC(8) [←→] [→←]

1. Đánh số cho đỉnh 8 và đưa nó vào ngăn xếp S.
 Gán num[8] = min_num[8] = k = 3 ; k++;
 Đưa 8 vào S. Kết quả: S = 1,2,8

Thực hiện đánh số và đưa vào Stack ✓ 9

2. Với các đỉnh kề v của 8: 7 Xét ✓ 10

2a. v = '7' chưa duyệt Duyệt nó ✓ 11

SCC(7) [←→] [→←]

1. Đánh số cho đỉnh 7 và đưa nó vào ngăn xếp S.
 Gán num[7] = min_num[7] = k = 4 ; k++;
 Đưa 7 vào S. Kết quả: S = 1,2,8,7

Thực hiện đánh số và đưa vào Stack ✓ 12

2. Với các đỉnh kề v của 7: 6 Xét ✓ 13

2a. v = '6' chưa duyệt Duyệt nó ✓ 14

SCC(6) [←→] [→←]

1. Đánh số cho đỉnh 6 và đưa nó vào ngăn xếp S.
 Gán num[6] = min_num[6] = k = 5 ; k++;
 Đưa 6 vào S. Kết quả: S = 1,2,8,7,6

Thực hiện đánh số và đưa vào Stack ✓ 15

2. Với các đỉnh kề v của 6: 7 Xét

16

2b. v = '7' duyệt rồi nhưng vẫn còn trên stack

Cập nhật min_num[u] 17

Cập nhật min_num[6] = min(min_num[6], num[7]) =

4 Cập nhật 18

3. Kiểm tra num và min_num của 6

num[6] == min_num[6] num[6] != min_num[6] 19

Cập nhật min_num[7] = min(min_num[7], min_num[6]) =

4 Cập nhật 20

3. Kiểm tra num và min_num của 7

num[7] == min_num[7] num[7] != min_num[7] 21

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 7

Các đỉnh được lấy ra: 6,7

Nội dung còn lại của ngăn xếp: 1,2,8

Cập nhật 22

Cập nhật min_num[8] = min(min_num[8], min_num[7]) = 3

Cập nhật 23

3. Kiểm tra num và min_num của 8

num[8] == min_num[8] num[8] != min_num[8] 24

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 8

Các đỉnh được lấy ra: 8

Nội dung còn lại của ngăn xếp: 1,2

Cập nhật 25

Cập nhật min_num[2] = min(min_num[2], min_num[8]) = 1

26

3. Kiểm tra num và min_num của 2

num[2] == min_num[2] num[2] != min_num[2] 27

Cập nhật min_num[1] = min(min_num[1], min_num[2]) = 1

Cập nhật 28

2a. v = '3' chưa duyệt Duyệt nó 29

SCC(3)

1. Đánh số cho đỉnh 3 và đưa nó vào ngăn xếp S.

Gán num[3] = min_num[3] = k = 6 ; k++;

Đưa 3 vào S. Kết quả: S = 1,2,3

Thực hiện đánh số và đưa vào Stack 30

2. Với các đỉnh kề v của 3: 4,5 Xét 31

2a. v = '4' chưa duyệt Duyệt nó 32

SCC(4)

1. Đánh số cho đỉnh 4 và đưa nó vào ngăn xếp S.

Gán num[4] = min_num[4] = k = 7 ; k++;

Đưa 4 vào S. Kết quả: S = 1,2,3,4

Thực hiện đánh số và đưa vào Stack 33

2. Với các đỉnh kề v của 4: 2,7,8 Xét 34

2b. v = '2' duyệt rồi nhưng vẫn còn trên stack Cập nhật min_num[u] 35

Cập nhật min_num[4] = min(min_num[4], num[2]) = 2

Cập nhật 36

2c. $v = '7'$ duyệt rồi và không còn trên stack X 37

2c. $v = '8'$ duyệt rồi và không còn trên stack X 38

3. Kiểm tra num và min_num của 4

num[4] == min_num[4] num[4] != min_num[4] X 39

Cập nhật min_num[3] = min(min_num[3], min_num[4]) = 2

40

2a. $v = '5'$ chưa duyệt 41

SCC(5)

1. Đánh số cho đỉnh 5 và đưa nó vào ngăn xếp S.

Gán num[5] = min_num[5] = k = 8 ; k++;

Đưa 5 vào S. Kết quả: S = 1,2,3,4,5

42

2. Với các đỉnh kề v của 5: 3,7 43

2b. $v = '3'$ duyệt rồi nhưng vẫn còn trên stack 44

Cập nhật min_num[5] = min(min_num[5], num[3]) = 6

45

2c. $v = '7'$ duyệt rồi và không còn trên stack X 46

3. Kiểm tra num và min_num của 5

num[5] == min_num[5] num[5] != min_num[5] X 47

Cập nhật min_num[3] = min(min_num[3], min_num[5]) = 2

48

3. Kiểm tra num và min_num của 3

num[3] == min_num[3] num[3] != min_num[3] X 49

Cập nhật min_num[1] = min(min_num[1], min_num[3]) = 1 50

3. Kiểm tra num và min_num của 1

num[1] == min_num[1] num[1] != min_num[1] 51

Lấy các đỉnh trong ngăn xếp ra cho đến khi lấy được 1

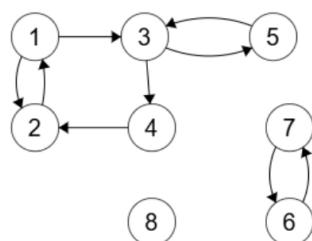
Các đỉnh được lấy ra: 5,4,3,2,1

Nội dung còn lại của ngăn xếp:

52

Vẽ các thành phần liên thông

Help Clear shift Delete Edit Undo Red Black



Test

Got

✓ Test tự động	1. Kiểm tra DFS - Bước 1. SCC(1): 1. Đánh số và đưa u vào ngăn xếp, với u = 1 - Bước 2. SCC(1): 2. Xử lý các đỉnh kề của 1 - Bước 3. SCC(1): 2a. Duyệt 2 - Bước 4. SCC(2): 1. Đánh số và đưa u vào ngăn xếp, với u = 2 - Bước 5. SCC(2): 2. Xử lý các đỉnh kề của 2 - Bước 6. SCC(2): 2b. Cập nhật min_id của 1 - Bước 7. SCC(2): 2b. Cập nhật min_id của 1 - Bước 8. SCC(2): 2a. Duyệt 8 - Bước 9. SCC(8): 1. Đánh số và đưa u vào ngăn xếp, với u = 8 - Bước 10. SCC(8): 2. Xử lý các đỉnh kề của 8 - Bước 11. SCC(8): 2a. Duyệt 7 - Bước 12. SCC(7): 1. Đánh số và đưa u vào ngăn xếp, với u = 7 - Bước 13. SCC(7): 2. Xử lý các đỉnh kề của 7 - Bước 14. SCC(7): 2a. Duyệt 6 - Bước 15. SCC(6): 1. Đánh số và đưa u vào ngăn xếp, với u = 6 - Bước 16. SCC(6): 2. Xử lý các đỉnh kề của 6 - Bước 17. SCC(6): 2b. Cập nhật min_id của 7 - Bước 18. SCC(6): 2b. Cập nhật min_id của 7 - Bước 19. SCC(6): 3b. id[u] != min_id[u] với u = 6 - Bước 20. SCC(7): 2a. Cập nhật min_id của 6 - Bước 21. SCC(7): 3a. id[u] == min_id[u] với u = 7 - Bước 22. SCC(7): 3a. Tìm được SCC của 7 - Bước 23. SCC(8): 2a. Cập nhật min_id của 7 - Bước 24. SCC(8): 3a. id[u] == min_id[u] với u = 8 - Bước 25. SCC(8): 3a. Tìm được SCC của 8 - Bước 26. SCC(2): 2a. Cập nhật min_id của 8 - Bước 27. SCC(2): 3b. id[u] != min_id[u] với u = 2 - Bước 28. SCC(1): 2a. Cập nhật min_id của 2 - Bước 29. SCC(1): 2a. Duyệt 3 - Bước 30. SCC(3): 1. Đánh số và đưa u vào ngăn xếp, với u = 3 - Bước 31. SCC(3): 2. Xử lý các đỉnh kề của 3 - Bước 32. SCC(3): 2a. Duyệt 4 - Bước 33. SCC(4): 1. Đánh số và đưa u vào ngăn xếp, với u = 4 - Bước 34. SCC(4): 2. Xử lý các đỉnh kề của 4 - Bước 35. SCC(4): 2b. Cập nhật min_id của 2 - Bước 36. SCC(4): 2b. Cập nhật min_id của 2 - Bước 37. SCC(4): 2c. Bỏ qua đỉnh 7 - Bước 38. SCC(4): 2c. Bỏ qua đỉnh 8 - Bước 39. SCC(4): 3b. id[u] != min_id[u] với u = 4 - Bước 40. SCC(3): 2a. Cập nhật min_id của 4 - Bước 41. SCC(3): 2a. Duyệt 5 - Bước 42. SCC(5): 1. Đánh số và đưa u vào ngăn xếp, với u = 5 - Bước 43. SCC(5): 2. Xử lý các đỉnh kề của 5 - Bước 44. SCC(5): 2b. Cập nhật min_id của 3 - Bước 45. SCC(5): 2b. Cập nhật min_id của 3 - Bước 46. SCC(5): 2c. BỎ QUA ĐỈNH 7 - Bước 47. SCC(5): 3b. id[u] != min_id[u] với u = 5 - Bước 48. SCC(3): 2a. Cập nhật min_id của 5 - Bước 49. SCC(3): 3b. id[u] != min_id[u] với u = 3 - Bước 50. SCC(1): 2a. Cập nhật min_id của 3 - Bước 51. SCC(1): 3a. id[u] == min_id[u] với u = 1 - Bước 52. SCC(1): 3a. Tìm được SCC của 1 2. Kiểm tra các thành phần liên thông Các thành phần liên thông Ok.	✓
---	---	--------------------------------------

Passed all tests! ✓

Đúng

Marks for this submission: 1,00/1,00. Accounting for previous tries, this gives 0,90/1,00.

Hoàn thành việc xem lại

◀ * Tự học - for + DFS = Duyệt
tất cả các đỉnh của đồ thị

Chuyển tới...

* Tự học - Áp dụng thuật toán
Tarjan tìm các BPLT của đồ thị có
hướng (ngẫu nhiên) ►

Bạn đang đăng nhập với tên Duc Luu Chau Minh (Thoát)

CT175HK1 2025 2026

Data retention summary

Get the mobile app

