**Tìm đường đi**

**2.1 Đường đi (***Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây***)**

Cho đồ thị G = (V, E) gồm n đỉnh biểu diễn dưới dạng ma trận kề và hai đỉnh u, v.

Yêu cầu:

(1) Tìm số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

(2) Tìm đường đi trên G từ đỉnh u đến v sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

- Dòng đầu chứa số nguyên dương t nhận giá trị 1 hoặc 2.

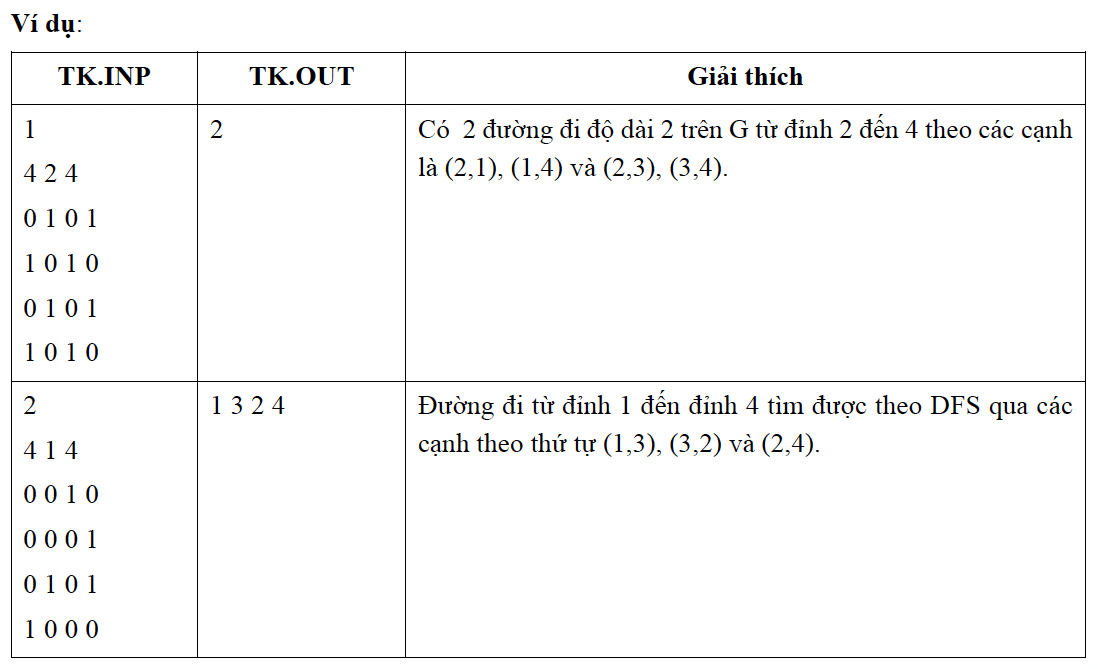
- Dòng thứ hai chứa ba số nguyên dương n, u và v. Trong đó, n là số đỉnh của G, u và v là hai đỉnh của G, với 1*≤*  u, v *≤*  n *≤* 100 và u khác v.

- Trong n dòng tiếp theo, mỗi dòng chứa n số 0 hoặc 1 mô tả ma trận kề của G.

Kết quả: Ghi ra tệp TK.OUT:

- Nếu t = 1 thì ghi ra giá trị là số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

- Nếu t = 2 thì ghi ra trên một dòng gồm dãy các đỉnh mô tả đường đi trên G từ u đến v. Trong trường hợp không có đường đi trên G từ u đến v thì ghi số 0.



2.2 (Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây)

Cho đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh và hai đỉnh u, v.

Yêu cầu:

(1) Tìm số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

(2) Tìm đường đi trên G từ đỉnh u đến v sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

- Dòng đầu chứa số nguyên dương t nhận giá trị 1 hoặc 2.

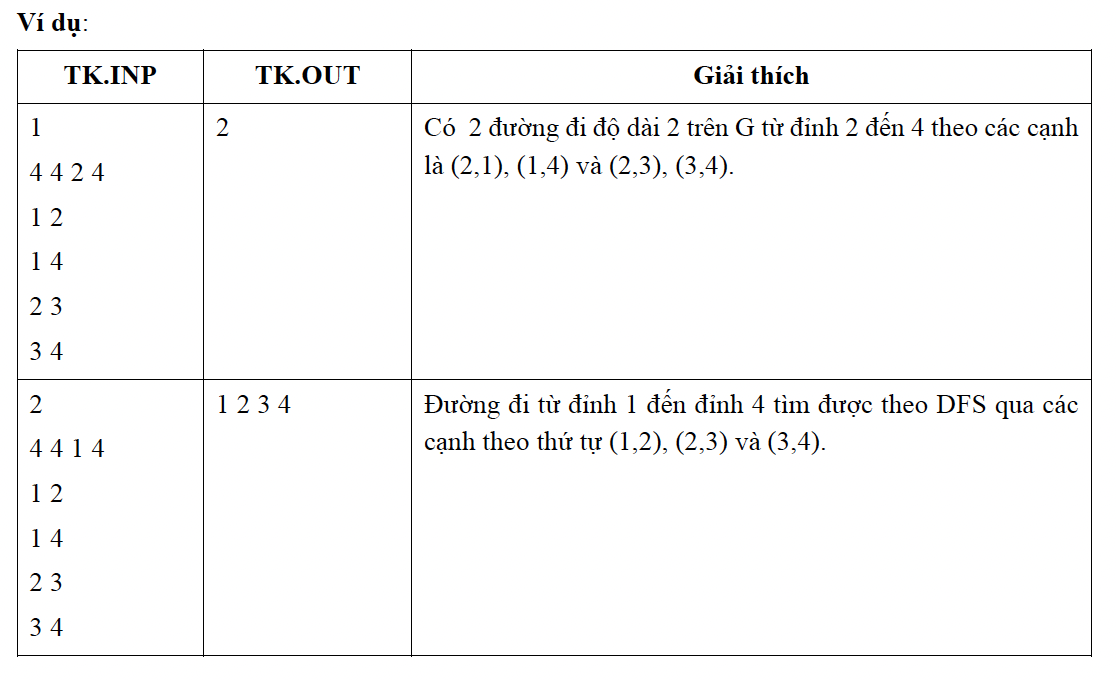
- Dòng thứ hai chứa bốn số nguyên dương n, m, u và v. Trong đó, n là số đỉnh, m là số cạnh của G, u và v là hai đỉnh của G, với 1 ≤ u, v ≤ n ≤ 100, m ≤ n(n-1)/2 và u khác v.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 ≤ i ≤ m) chứa hai số nguyên ui, vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1  ui < vi  n.

Kết quả: Ghi ra tệp TK.OUT:

- Nếu t = 1 thì ghi ra giá trị là số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

- Nếu t = 2 thì ghi ra trên một dòng gồm dãy các đỉnh mô tả đường đi trên G từ u đến v. Trong trường hợp không có đường đi trên G từ u đến v thì ghi số 0.



2.3. Đường đi. *(Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây)*

Cho đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh và hai đỉnh u, v.

Yêu cầu:

(1) Tìm số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

(2) Tìm đường đi trên G từ đỉnh u đến v sử dụng thuật toán tìm kiếm theo chiều rộng (BFS).

Dữ liệu: Vào từ tệp TK.INP:

- Dòng đầu chứa số nguyên dương t nhận giá trị 1 hoặc 2.

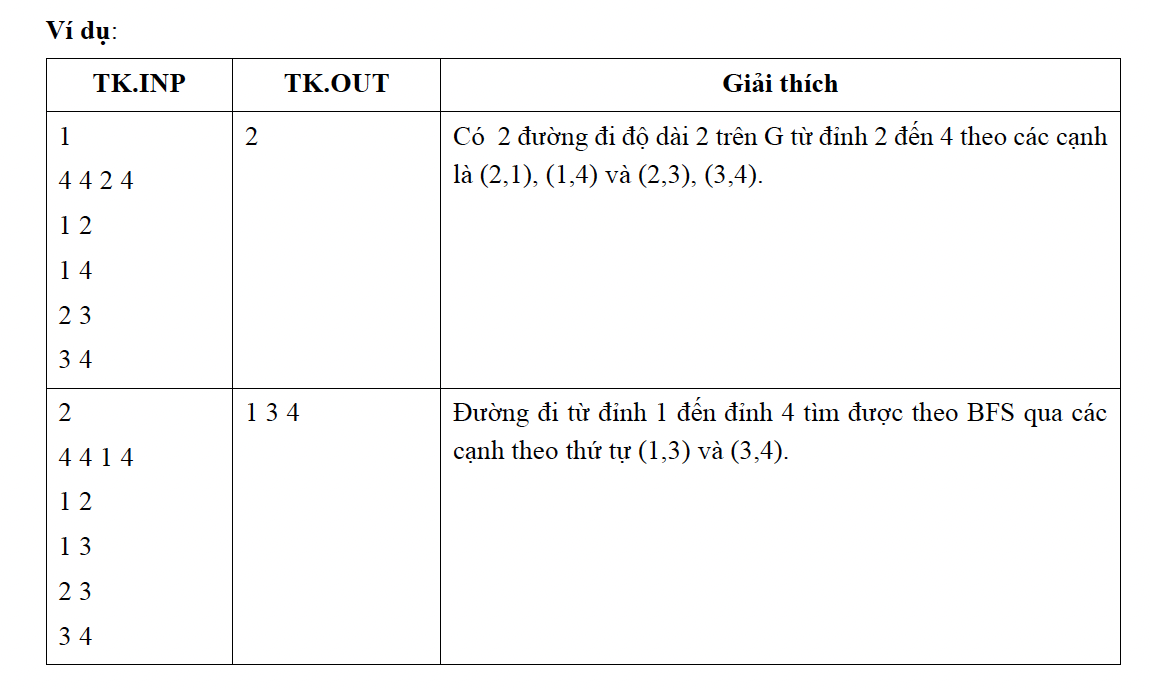
- Dòng thứ hai chứa bốn số nguyên dương n, m, u và v. Trong đó, n là số đỉnh, m là số cạnh của G, u và v là hai đỉnh của G, với 1 *≤* u, v *≤* n *≤* 100, m *≤* n(n-1)/2 và u khác v.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* m) chứa hai số nguyên ui, vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1 *≤* ui < vi *≤* n.

Kết quả: Ghi ra tệp TK.OUT:

- Nếu t = 1 thì ghi ra giá trị là số lượng đường đi độ dài 2 trên G từ đỉnh u đến v.

- Nếu t = 2 thì ghi ra trên một dòng gồm dãy các đỉnh mô tả đường đi trên G từ u đến v. Trong trường hợp không có đường đi trên G từ u đến v thì ghi số 0.



2.4. **Liên thông** (*Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây*)

Cho đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng ma trận kề.

Yêu cầu: Tìm các thành phần liên thông của G sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

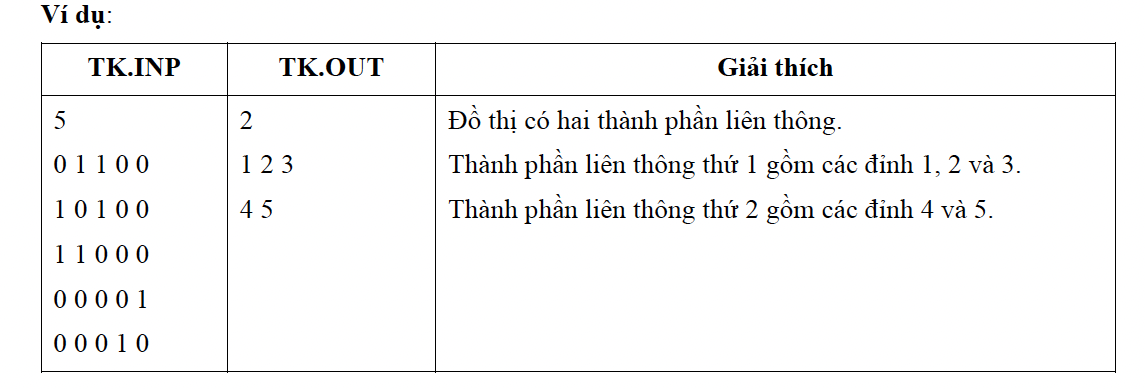
- Dòng đầu chứa số nguyên dương n là số đỉnh của G, n *≤* 100.

- Trong n dòng tiếp theo, mỗi dòng chứa n số 0 hoặc 1 mô tả ma trận kề của G.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra giá trị lt là số lượng các thành phần liên thông của G.

- Trong lt dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* lt) ghi các đỉnh thuộc thành phần liên thông thứ i theo thứ tự tăng.



2.5. **Liên thông** (Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây)

Cho trước đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng ma trận kề.

Yêu cầu: Tìm các thành phần liên thông của G sử dụng thuật toán tìm kiếm theo chiều rộng (BFS).

Dữ liệu: Vào từ tệp TK.INP:

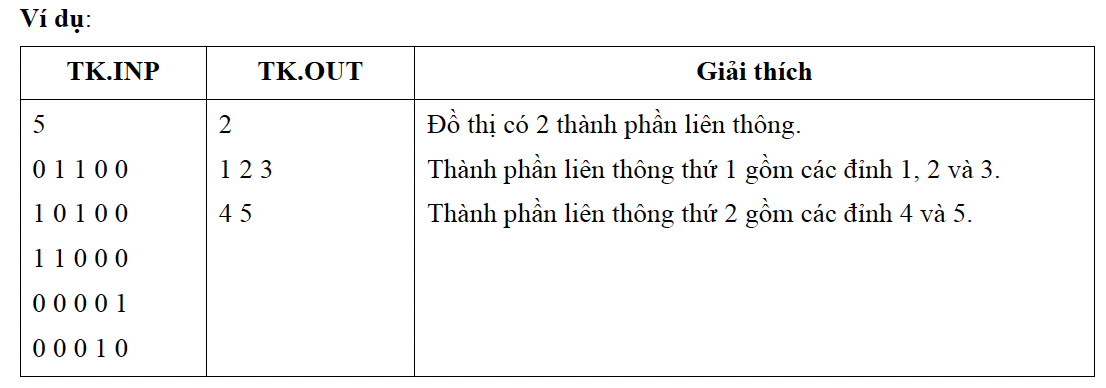
- Dòng đầu chứa số nguyên dương n là số đỉnh của G, n ≤ 100.

- Trong n dòng tiếp theo, mỗi dòng chứa n số 0 hoặc 1 mô tả ma trận kề của G.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra giá trị lt là số lượng các thành phần liên thông của G.

- Trong lt dòng tiếp theo, mỗi dòng thứ i (1 ≤ i ≤ lt) ghi các đỉnh thuộc thành phần liên thông thứ i theo thứ tự tăng.



**2.6. Đỉnh trụ** (Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây)

Cho trước đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách kề.

Yêu cầu: Tìm các đỉnh trụ của G sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

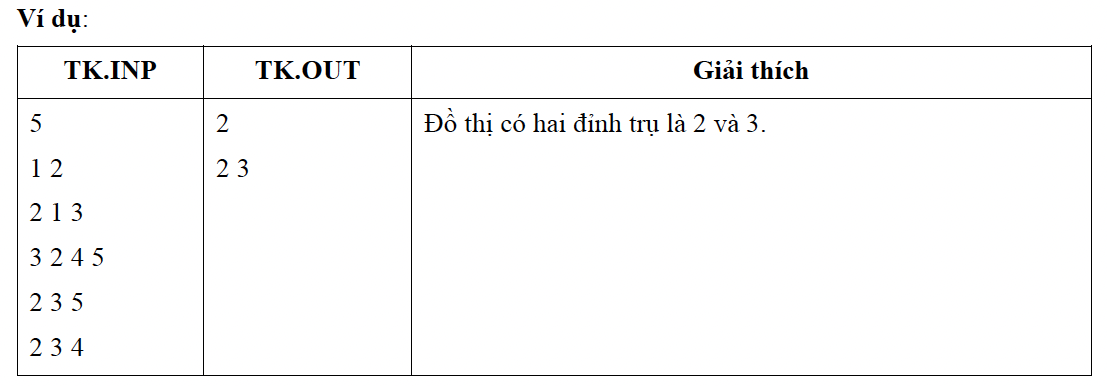
- Dòng đầu chứa số nguyên dương n số đỉnh của G, với n ≤ 100.

- Trong n dòng tiếp theo, mỗi dòng thứ i (1 ≤ i ≤ n) chứa số tự nhiên k (1 ≤ k < n) là số lương đỉnh kề với đỉnh i và k số tự nhiên theo thứ tự tăng v1, …, vk là số hiệu các đỉnh kề tương ứng.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra giá trị t là số lượng các đỉnh trụ của G.

- Trong trường hợp t > 0, dòng tiếp theo ghi các đỉnh trụ tìm được theo thứ tự tăng.



2.7. **Cạnh cầu** (*Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây*)

Cho trước đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh.

Yêu cầu: Tìm các cạnh cầu của G sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

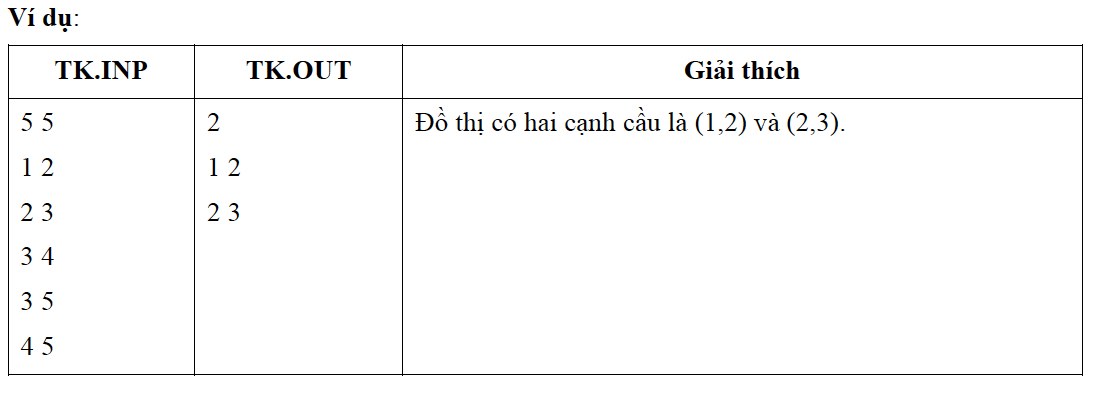
- Dòng đầu chứa hai số nguyên dương n và m. Trong đó, n là số đỉnh và m là số cạnh của G, với n *≤* 100 và m *≤* n(n-1)/2.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* m) chứa hai số ui và vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1 *≤* ui < vi *≤* n.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra giá trị c là số lượng các cạnh cầu của G.

- Trong trường hợp c > 0, trong c dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* c) ghi hai số nguyên dương ui và vi là đỉnh đầu và đỉnh cuối của cạnh cầu thứ i tìm được. Các cạnh cầu được ghi ra theo thứ tự từ điển.



2.8. **Đỉnh thắt** (*Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây*)

Cho đồ thị vô hướng liên thông G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh và hai đỉnh u, v. Một đỉnh s của G gọi là đỉnh thắt của cặp u, v nếu mọi đường đi trên G từ u đến v đều phải đi qua đỉnh s.

Yêu cầu: Tìm các đỉnh thắt của cặp đỉnh u và v sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

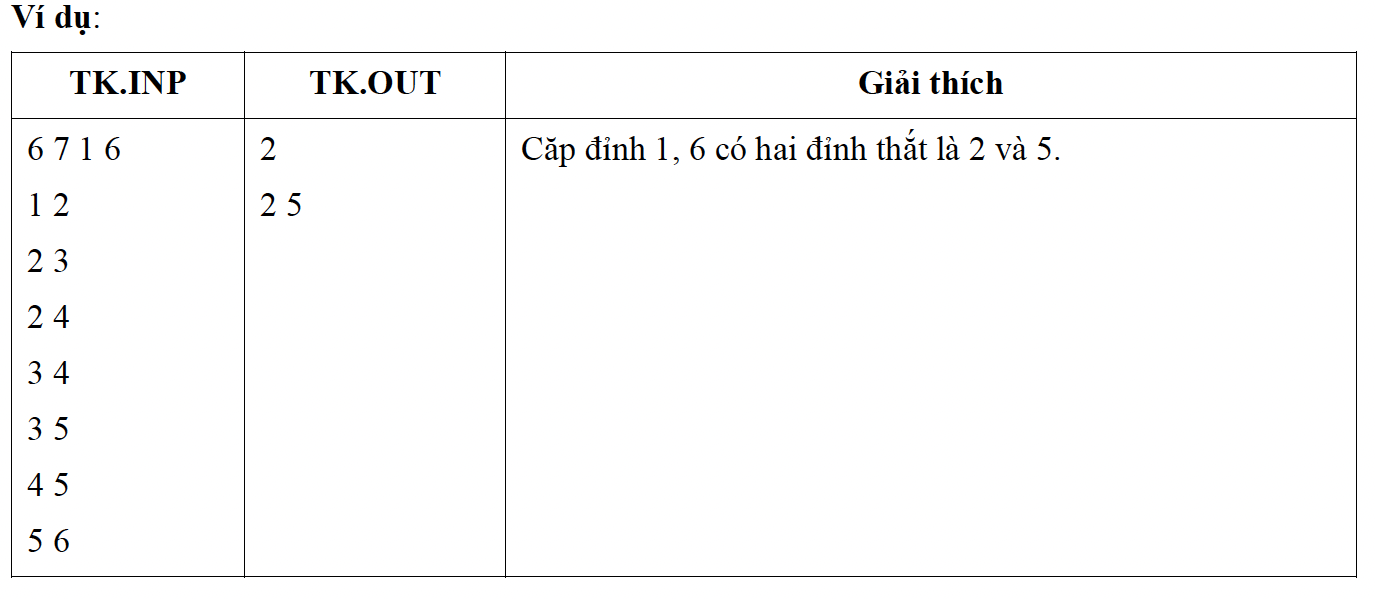
- Dòng đầu chứa bốn số nguyên dương n, m, u và v. Trong đó, n là số đỉnh, m là số cạnh, u và v là hai đỉnh khác nhau của G, với 1 *≤* u, v *≤* n *≤* 100 và m *≤* n(n-1)/2.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* m) chứa hai số ui và vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1 *≤* ui < vi *≤* n.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra giá trị t là số lượng các đỉnh thắt của cặp đỉnh u và v.

- Trong trường hợp t > 0, dòng tiếp theo ghi các đỉnh thắt tìm được theo thứ tự tăng.



**2.9. Định chiều đồ thị** (*Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây*)

Cho đồ thị vô hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh. Khi đó, G gọi là định chiều được nếu có thể biến đổi G thành đồ thị có hướng liên thông mạnh G’ = (V, E’) bằng cách định chiều mỗi cạnh vô hướng thành một cung (cạnh) có hướng. Phép biến đổi như trên gọi là phép định chiều.

Yêu cầu:

(1) Kiểm tra G có phải là đồ thị định chiều được hay không?

(2) Tìm một phương án định chiều G sử dụng phép định chiều dựa trên thuật toán tìm kiếm theo chiều sâu (DFS) bắt đầu tại đỉnh 1.

Dữ liệu: Vào từ tệp TK.INP:

- Dòng đầu chứa số nguyên t nhận giá trị 1 hoặc 2.

- Dòng sau chứa hai số nguyên dương n và m. Trong đó, n là số đỉnh, m là số cạnh của G, với n *≤*  100 và m *≤*  n(n-1)/2.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤*  i *≤*  m) chứa hai số ui và vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1 *≤* ui < vi *≤*  n.

Kết quả: Ghi ra tệp TK.OUT:

- Nếu t = 1 thì ghi ra giá trị 1 hoặc 0 tùy thuộc G là đồ thị định chiều được hoặc không.

- Nếu t = 2 thì ghi ra danh sách cạnh của đồ thị G’ theo quy cách:

+ Dòng đầu ghi ra hai số nguyên dương n và m là số đỉnh và số cạnh của G’;

+ Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤*  i *≤*  m) chứa hai số ui và vi là đỉnh đầu và đỉnh cuối của cạnh có hướng ei. Các cạnh được liệt kê theo thứ tự từ điển.

|  |  |  |
| --- | --- | --- |
| TK.INP | TK.OUT | Giải thích |
| 1  4 5  1 2  1 3  1 4  2 4  3 4 | 1 | Đồ thị vô hướng đã cho liên thông và không chứa cạnh cầu nên định chiều được. |
| 2  4 5  1 2  1 3  1 4  2 4  3 4 | 4 5  1 2  2 4  3 1  4 1  4 3 | Các cung thuận được định chiều là (1,2), (2,4), (4,3).  Các cung nghịch được định chiều là (3,1), (4,1). |

**2.10. Thành phần liên thông mạnh** (*Tệp chương trình: TK.CPP; Thời gian chạy chương trình ≤ 1 giây*)

Cho đồ thị có hướng G = (V, E) gồm n đỉnh biểu diễn dưới dạng danh sách cạnh.

Yêu cầu: Tìm các thành phần liên thông mạnh của đồ thị G sử dụng thuật toán tìm kiếm theo chiều sâu (DFS).

Dữ liệu: Vào từ tệp TK.INP:

- Dòng đầu chứa hai số nguyên dương n và m. Trong đó, n là số đỉnh, m là số cạnh của G, với n *≤* 100 và m *≤* n(n-1)/2.

- Trong m dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* m) chứa hai số nguyên dương ui và vi là đỉnh đầu và đỉnh cuối của cạnh ei, với 1 *≤* ui, vi *≤* n.

Kết quả: Ghi ra tệp TK.OUT:

- Dòng đầu ghi ra ghi ra giá trị k là số thành phần liên thông mạnh của G.

- Trong k dòng tiếp theo, mỗi dòng thứ i (1 *≤* i *≤* k) ghi các đỉnh thuộc thành phần liên thông thứ i theo thứ tự tăng.

