LÝ THUYẾT ĐỒ THỊ

TS. Nguyễn Thị Phương Trâm
Email: ntptram@hcmuaf.edu.vn

Nội dung chương trình

- Mục tiêu môn học

  Cung cấp cho sinh viên các khái niệm cơ bản của lý thuyết đồ thị, đồ thị Euler, Hamilton, cây và cây khung bé nhất của đồ thị, bài toán đường đi ngắn nhất và bài toán luồng cực đại trong mạng => Giúp sinh viên có thể sử dụng mô hình lý thuyết đồ thị để mô hình hóa vấn đề bài toán thực tế một cách hiệu quả. Học phần này trang bị những kiến thức toán nền tảng phục vụ cho các chuyên ngành thuộc lĩnh vực CNTT.

Nội dung chương trình

- Thời lượng
  - Lý thuyết : 45 tiết
  - Thực hành: 30 tiết

Nội dung chương trình

1. Các khái niệm cơ bản về đồ thị
2. Tìm kiếm trên đồ thị
3. Đồ thị phẳng
4. Cây và cây bao trùm
5. Bài toán đường đi ngắn nhất
6. Một số bài toán ứng dụng

Kiểm tra đánh giá

- Điểm danh (10%)
- Bài tập thực hành hằng tuần (10%) (Điểm cá nhân)
- Kiểm tra giữa kỳ (30%) (Đồ án nhóm)
- Thi kết thúc môn (50%)

Giáo trình và TLTK

- **Giáo trình**
  - Nguyễn Cam, Chu Đức Khánh, *Lý thuyết Đồ thị*, 1998.
  - Kenneth H. Rosen, Discrete Mathematics and Its Applications
- **Tài liệu tham khảo**
  - Silde bài giảng ThS. Trần Quốc Việt
  - Silde bài giảng ThS. Lê Phi Hùng

Lý thuyết đồ thị

Chương 1: Giới thiệu tổng quan

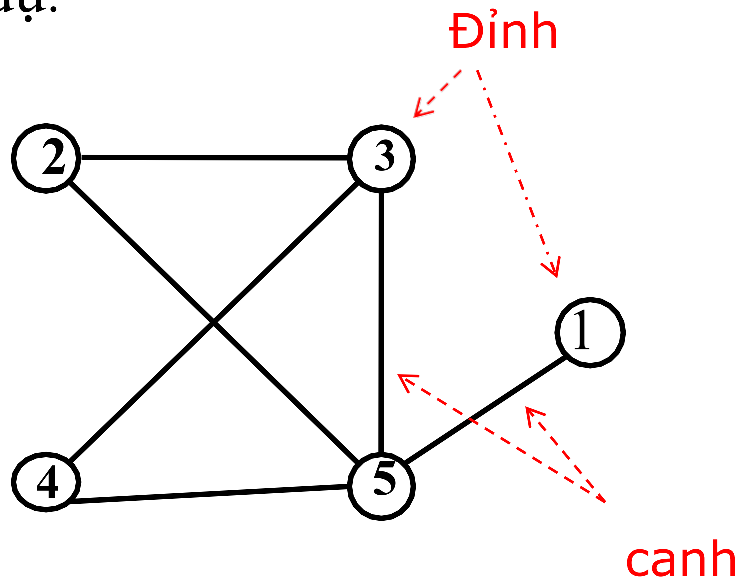
Chương 1: Giới thiệu tổng quan

1. Khái niệm đồ thị, một số lĩnh vực ứng dụng của đồ thị
2. Các loại đồ thị, và định nghĩa
3. Một số đồ thị đặc biệt
4. Biểu diễn đồ thị
5. Đường đi và chu trình
6. Liên thông và thành phần liên thông
7. Một số vấn đề liên quan đến cài đặt đồ thị
8. Tìm kiếm trên đồ thị

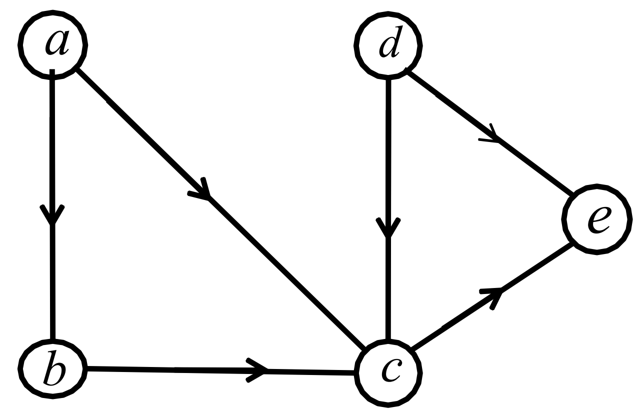
1. Khái niệm

- Đồ thị G là một tập hợp gồm các đỉnh và các cạnh. Ta thường ký hiệu: G = (V, E), trong đó:
  + V: Là tập các đỉnh
  + E: Là tập các cạnh

Ví dụ:



Đồ thị vô hướng

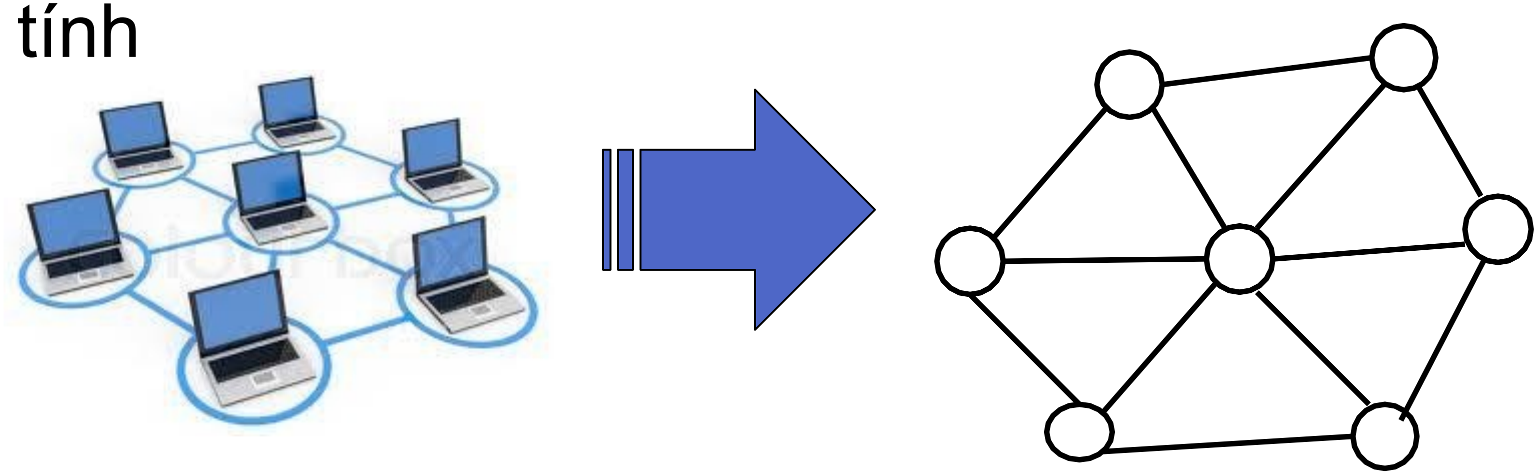


Đồ thị có hướng

1. Một số lĩnh vực ứng dụng

Trong thực tế, rất nhiều bài toán thuộc các lĩnh vực khác nhau được giải bằng đồ thị:

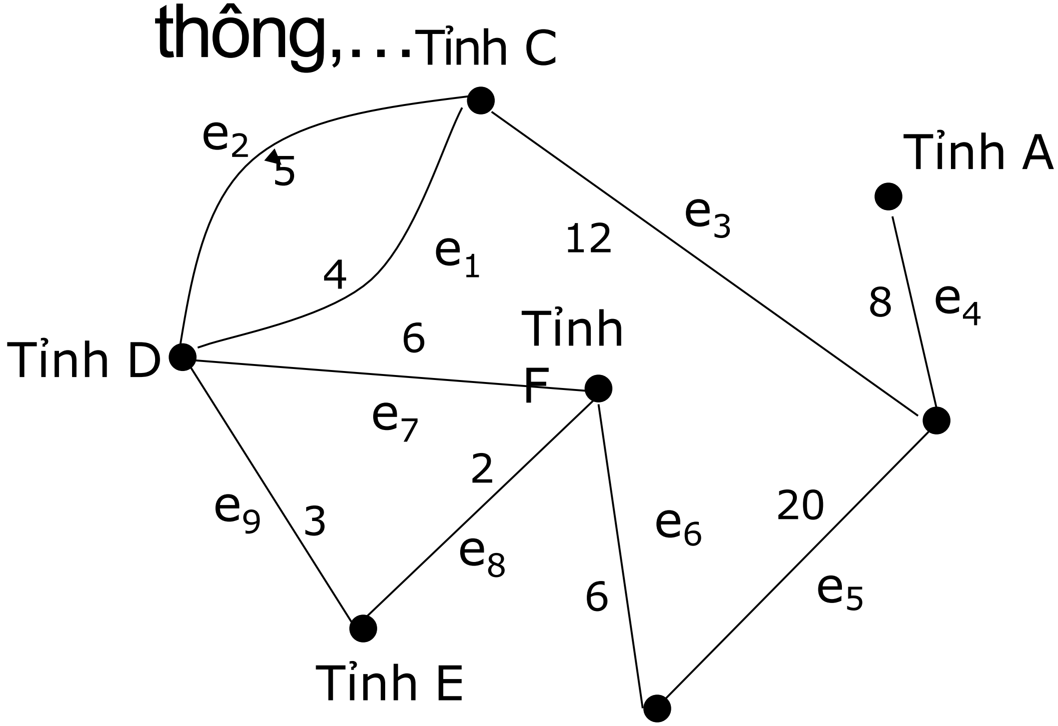
❖ Lĩnh vực mạng máy tính: Biểu diễn mạng máy tính



Xác định 2 máy có thể liên lạc vơi nhau trên một mạng,...

1. Một số lĩnh vực ứng dụng

❖ Lĩnh vực giao thông: Tìm đường đi, đường đi ngắn nhất giữa hai thành phố trong mạng giao thông,…



- Mỗi đỉnh: một tỉnh
- Mỗi cạnh nối 2 đỉnh u,v: Có đường đi trực tiếp giữa 2 tỉnh u,v
- Con số trên mỗi cạnh: Độ dài đường đi trực tiếp giữa 2 tỉnh.

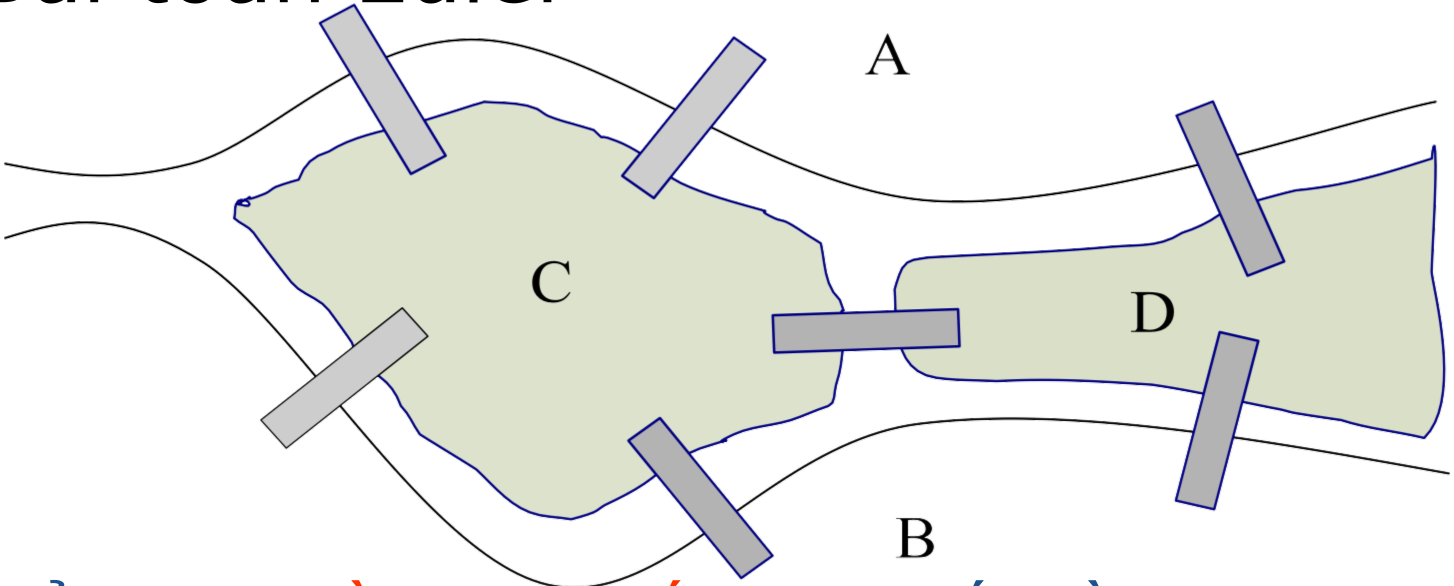
- **Yêu cầu:** Tìm đường đi ngắn nhất từ một tỉnh nào đó đến một tỉnh khác (chẳng hạn từ A đến F)?

1. Một số lĩnh vực ứng dụng

- Giải các bài toán về lập lịch, thời khóa biểu, và phân bố tần số cho các trạm phát thanh và truyền hình
- ….

1. Một số lĩnh vực ứng dụng

❖ Bài toán Euler



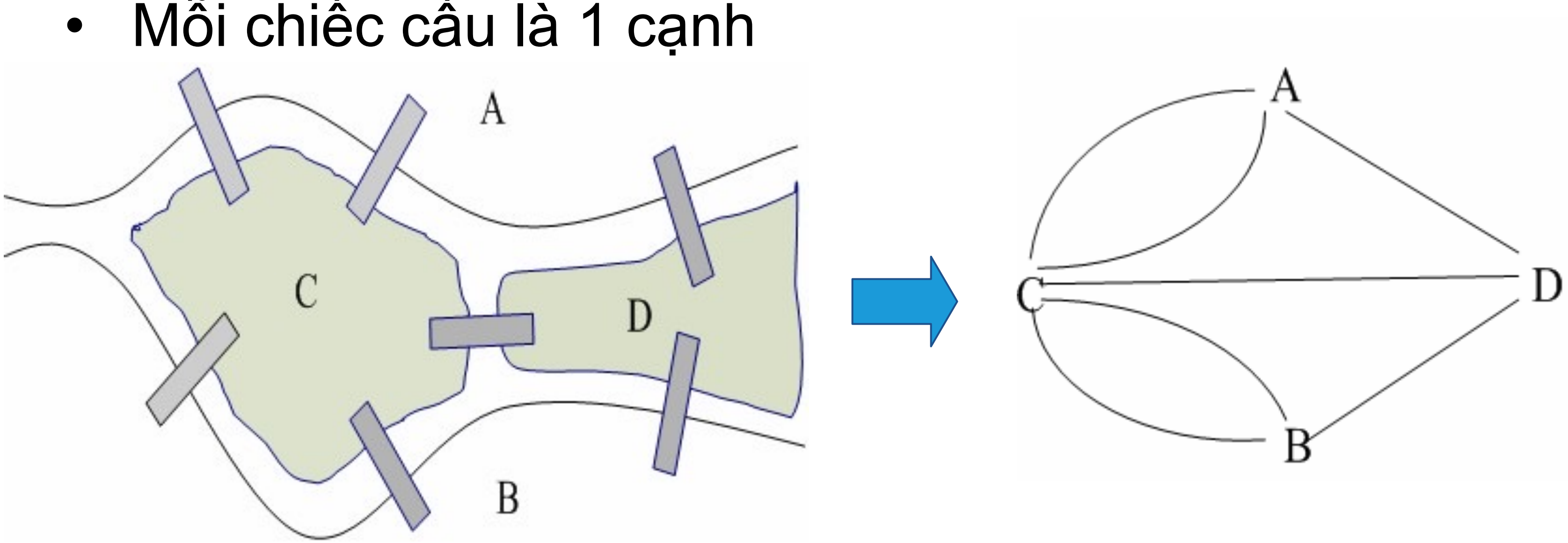
Có thể chỉ <u>một lần</u> đi qua <u>tất cả</u> 7 chiếc cầu này hay không?

Giải bằng đồ thị

1. Một số lĩnh vực ứng dụng

❖ Chuyển bài toán về dạng đồ thị

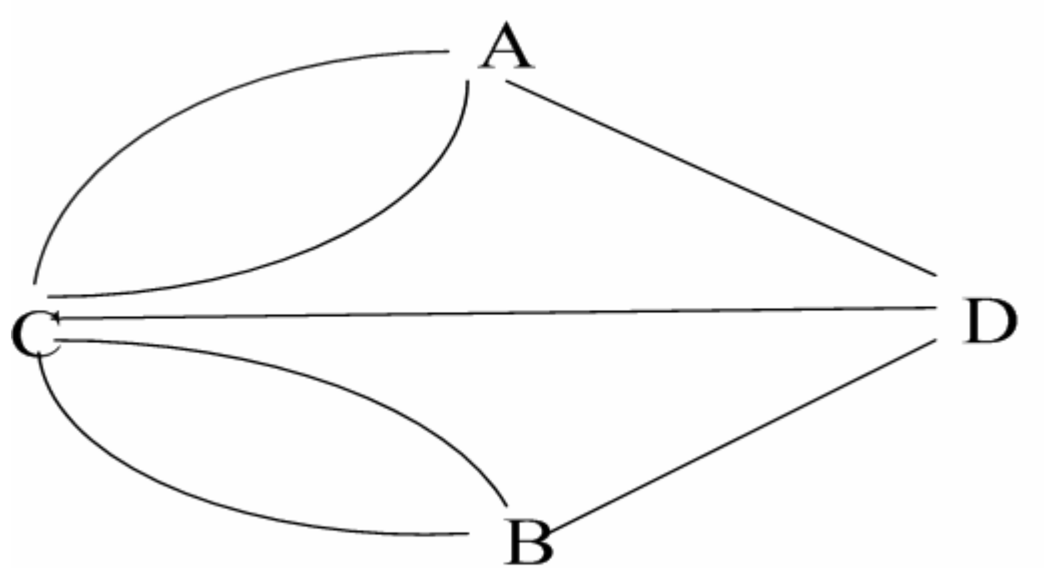
- Mỗi vùng là 1 đỉnh
- Mỗi chiếc cầu là 1 cạnh



1. Một số lĩnh vực ứng dụng

❖ Đồ thị được xây dựng từ bài toán Euler

- Có thể đi qua tất cả các canh của đồ thị, sao cho mỗi cạnh chỉ đi qua đúng một lần được không?



2. Các loại đồ thị

- Đồ thị
  - Đồ thị vô hướng
    - Đơn đồ thị
    - Đa đồ thị
    - Giả đồ thị
  - Đồ thị có hướng
    - Đơn đồ thị
    - Đa đồ thị

2. Một số định nghĩa

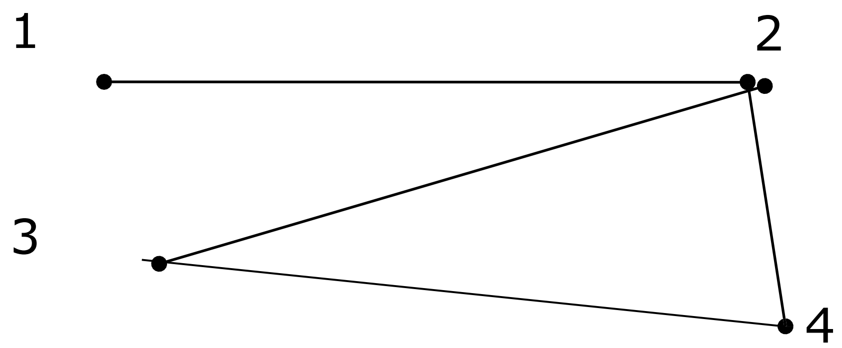
❖ ***Đồ thị vô hướng (undirected graph):***

➢ Đồ thị vô hướng G=(V,E) với:
   - $V \neq \emptyset$ là tập các đỉnh
   - E: Là đa tập hợp với các phần tử có dạng (u,v) với $u,v \in V$ không có thứ tự, gọi là các cạnh của đồ thị

➢ Biểu diễn bằng biểu đồ:
   - Mỗi đỉnh $\leftrightarrow$ một điểm
   - Mỗi cạnh (u,v) $\leftrightarrow$ một cạnh vô hướng nối giữa u và v

2. Một số định nghĩa

❖ Ví dụ: Cho đồ thị G

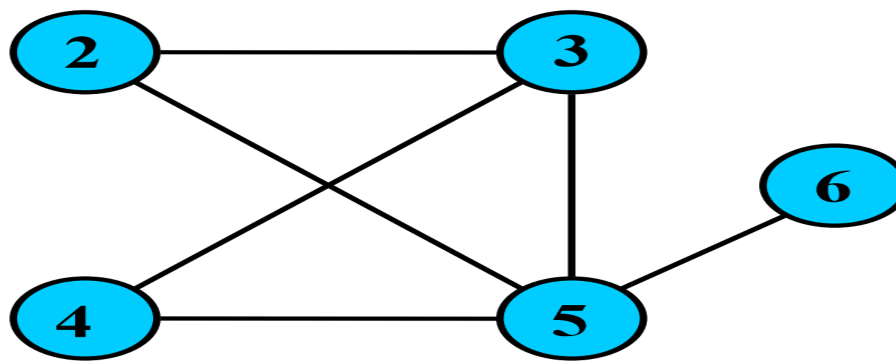


- Tập đỉnh V ={1,2,3,4}
- Tập cạnh E ={(1,2), (2,3), (3,4), (2,4)}
- Kí hiệu: G = (V,E)

Bài tập áp dụng 1

Ví dụ: Cho tập V = {2, 3, 4, 5, 6}. Hãy biểu diễn quan hệ nguyên tố cùng nhau của tập trên (số nguyên tố cùng nhau là nếu chúng có ước số chung lớn nhất là 1). Hãy biểu diễn quan hệ này bằng đồ thị vô hướng.

Quan hệ được biểu diễn bằng đồ thị như sau



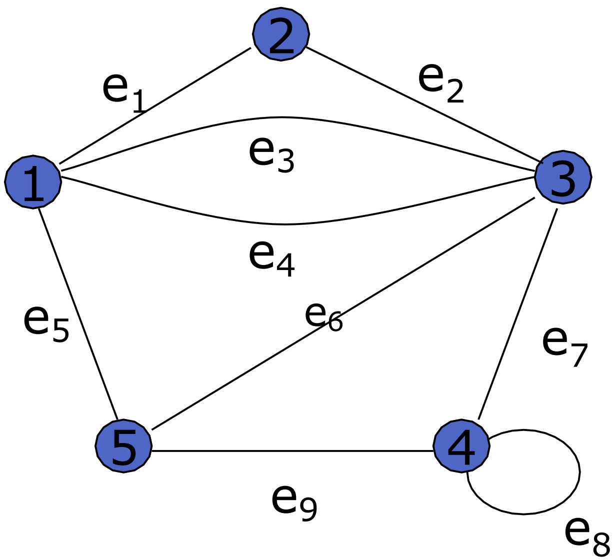
2. Một số định nghĩa

❖ Cho đồ thị vô hướng G=(V,E)

- Với cạnh e=(u,v)$\in$E, u,v gọi là 2 đỉnh kề nhau, e gọi là cạnh liên thuộc với 2 đỉnh u,v
- Hai cạnh e_1, e_2 liên kết cùng một cặp đỉnh khác nhau được gọi là 2 cạnh song song (paralell edges).
- Một cạnh trên cùng một đỉnh gọi khuyên (loop).

2. Một số định nghĩa

❖ ***Ví dụ:***

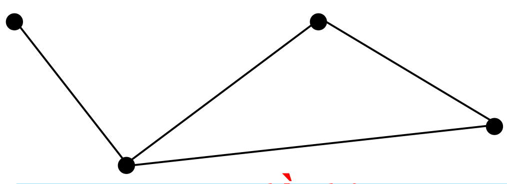


- Đỉnh 1 kề với đỉnh 2
- Đỉnh 2 kề với đỉnh 3
- Đỉnh 5 kề với đỉnh 4
- Đỉnh 1 không kề với đỉnh 4
- …
- e_3, e_4: Các cạnh song song
- e_8: Khuyên

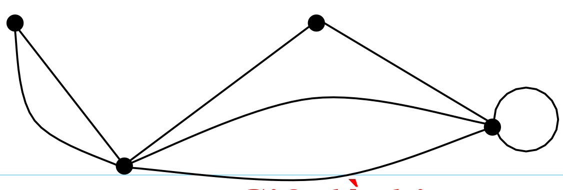
2. Một số định nghĩa

❖ Cho đồ thị vô hướng G=(V,E):

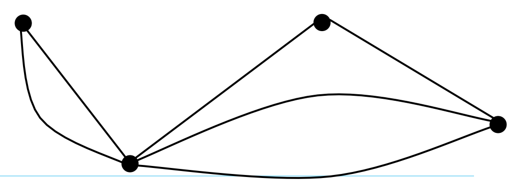
- **G là đồ thị đơn (Simple graph)** nếu G không có khuyên và không có cạnh song song
- **G gọi là đa đồ thị (multigraphs)**nếu G không có khuyên và có thể có các cạnh song song
- **G gọi là giả đồ thị (pseudographs)** nếu G có thể có cả khuyên và các cạnh song song.



Đơn đồ thị



Giả đồ thị



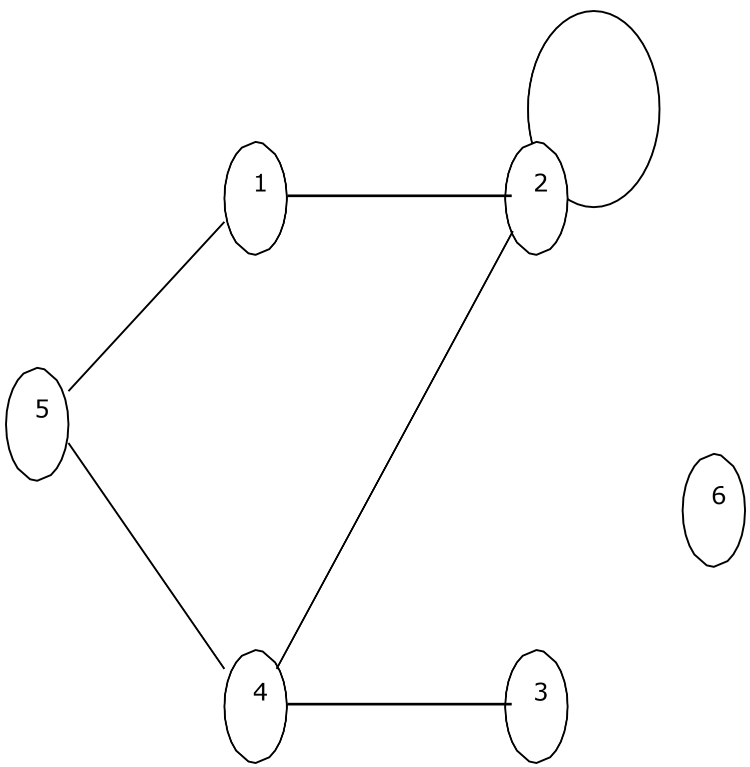
Đa đồ thị

2. Một số định nghĩa

- **Bậc của đỉnh trong đồ thị vô hướng**: Bậc của đỉnh v trong đồ thị vô hướng là số cạnh tới đỉnh v, kí hiệu **deg(v).** Mỗi vòng tại một đỉnh sẽ được xem như 2 cạnh tới đỉnh đó.
  - Đỉnh có bậc 0 gọi là đỉnh cô lập (isolated vertex)
  - Đỉnh có bậc 1 gọi là đỉnh treo (pendant vertex)

2. Một số định nghĩa

❖ Ví dụ:



deg(1)=deg(5)=2,

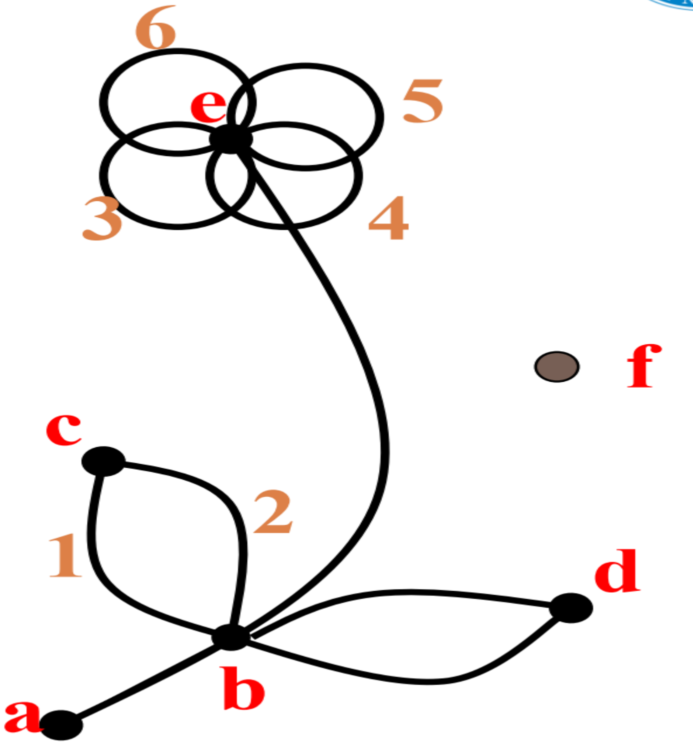
deg(4)=3, deg(3)=1, deg(6)=0, deg (2)=???

3: Đỉnh treo, 6: Đỉnh cô lập

Bài tập áp dụng 2

❖ Ví dụ: xác định Bậc của đỉnh trong hình sau

- deg(a) = ?
- deg(b) = ?
- deg(c) = ?
- deg(d) = ?
- deg(e) = ?
- deg(f) = ?



2. Một số định nghĩa

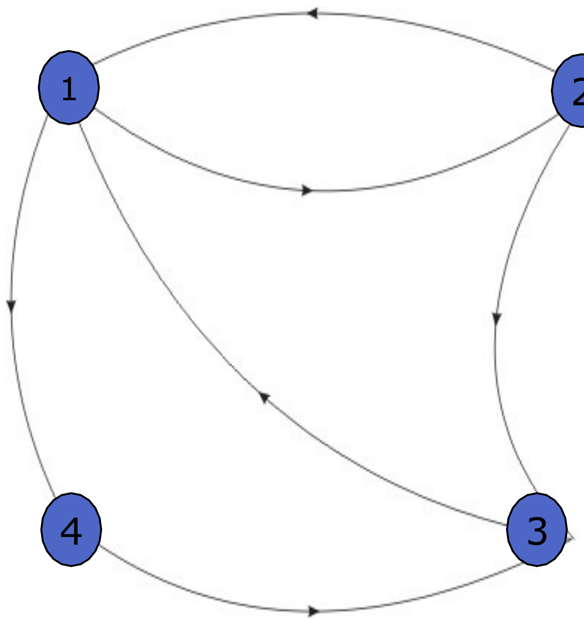
- **Đồ thị có hướng (***directed graph)* Đồ thị có hướng G = (V,E), V≠∅ là tập các đỉnh, E là tập các cặp (u,v) có **thứ tự** trong V gọi là các cung.
  - Với (u,v)∈E, u gọi là đỉnh đầu, v gọi là đỉnh cuối của cung (u,v) và v gọi là đỉnh kề của u.
  - Hai cung e_1, e_2 liên kết cùng một cặp đỉnh được gọi là 2 cung song song (paralell edges).
  - Cung từ một đỉnh đến chính nó gọi là khuyên (loop).

2. Một số định nghĩa

- Cho đồ thị có hướng G=(V, E)
  - **G là đơn đồ thị có hướng (Simple directed Graphs)** nếu G không có khuyên và không có cạnh song song cùng chiều.
  - **G là đa đồ thị có hướng (Directed multigraphs)** nếu G có thể có các khuyên, các cạnh song song cùng chiều
- Đồ thị hỗn hợp (Mixed Graph): là đồ thị mà có chứa cả cạnh vô hướng và cạnh có hướng

2. Một số định nghĩa

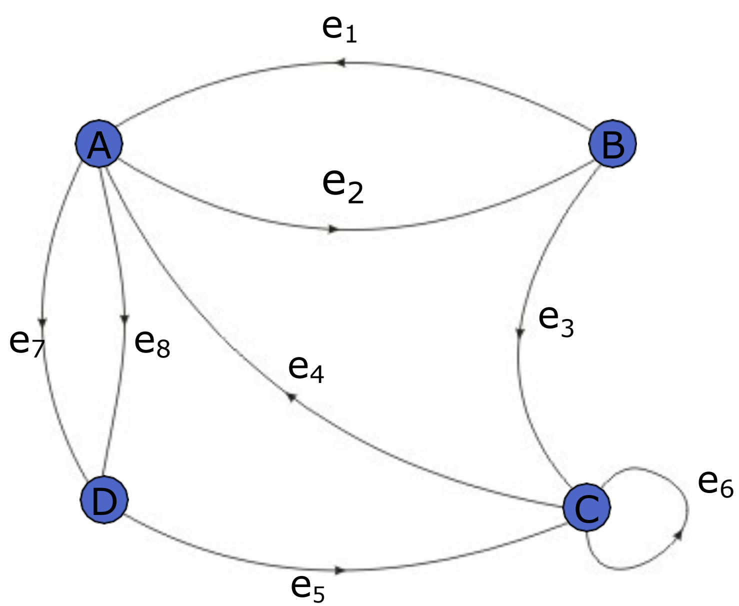
❖ ***Ví dụ***



Đơn đồ thị có hướng

2. Một số định nghĩa

❖ Ví dụ đa đồ thị có hướng:



- A,B,C,D: Các đỉnh
- $e_1,e_2,e_3,e_4,e_5,e_6,e_7,e_8$: Các cung
- e_1,e_2: Song song ngược chiều
- e_7,e_8: Song song cùng chiều
- e_6: Khuyên

2. Một số định nghĩa

- Đồ thị
  - Đồ thị vô hướng — Không có thứ tự
    - Đơn đồ thị — Không cạnh lặp, không khuyên
    - Đa đồ thị — Có cạnh lặp, không khuyên
    - Giả đồ thị — Có cạnh lặp, Có khuyên
  - Đồ thị có hướng — Có thứ tự
    - Đơn đồ thị — Không cung lặp, không khuyên
    - Đa đồ thị — Có cung lặp, có thể có khuyên
  - **Đồ thị hỗn hợp**
    - -Đồ thị vô hướng
    - - Đồ thị có hướng

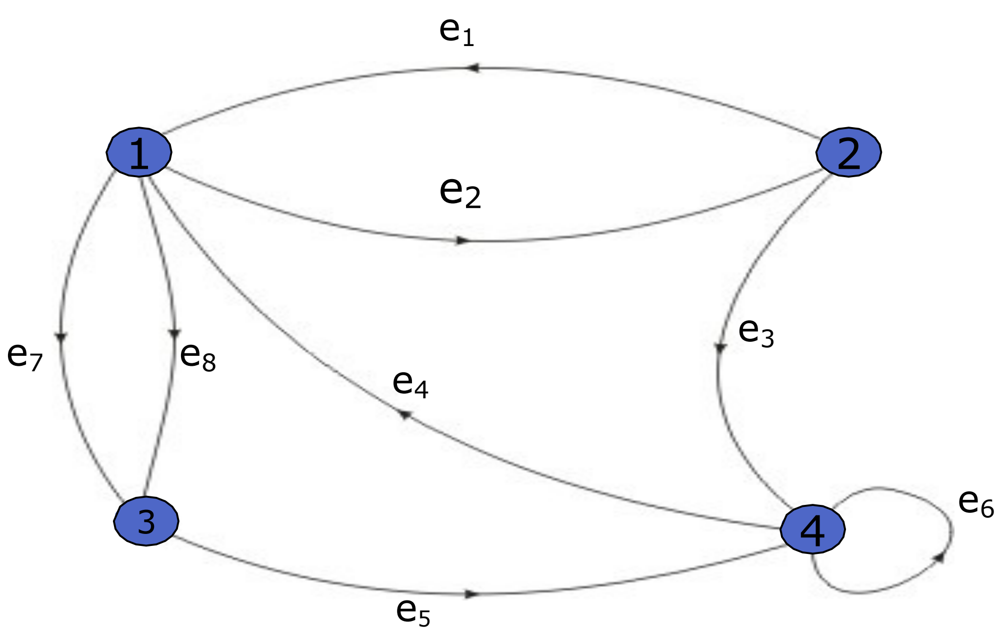
2. Một số định nghĩa

❖ **Bậc của đỉnh trong đồ thị có hướng**: Cho đồ thị có hướng G = (V,E) và $v \in V$.

- Nửa bậc trong của v, kí hiệu $deg^{-}(v)$ là số cung đến đỉnh v.
- Nửa bậc ngoài của v, kí hiệu $deg^{+}(v)$ là số cung xuất phát từ v.

2. Một số định nghĩa

❖ **Ví dụ: Cho đồ thị**



$deg^+(1)=?$

$deg^-(1)=?$

$deg^+(2)=?$

$deg^-(2)=?$

$deg^+(4)=?$

$deg^-(4)=?$

deg(1)?

deg(2)?

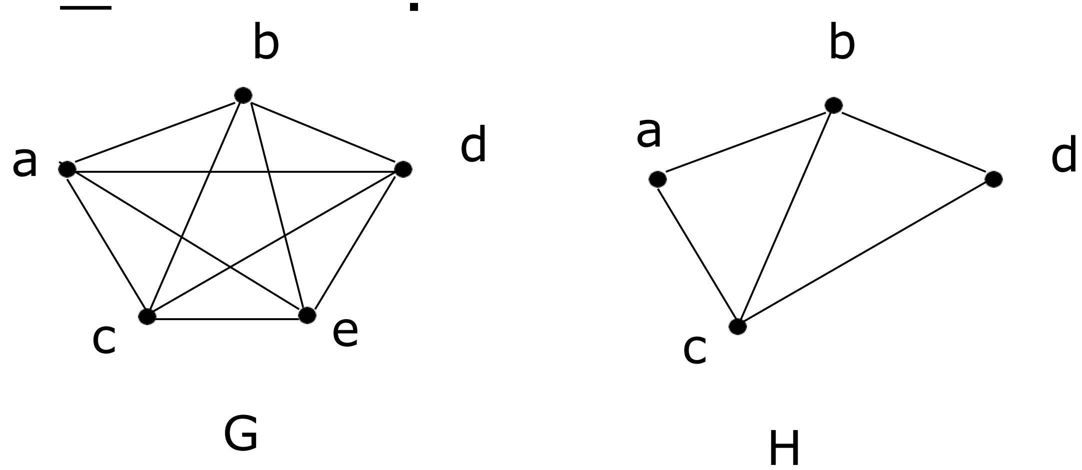
Luyện tập

1. Hãy biểu diễn quan hệ ước chung lớn nhất bằng 2 của các cặp hai số trong tập hợp V = {1, 2, 3, 4, 5, 6,7, 8}.
2. Vẽ đồ thị vô hướng G = (V, E) cho bởi: V = {A, B, C, D, E, F} và E = {(E, F), (B, F), (D, C), (D, F), (F, B), (C, F), (A, F), (E, D)}
3. Trong trận đấu vòng tròn, đội Hổ thắng đội Giẻ cùi xanh, Chim giáo chủ và Chim vàng anh. Đội Giẻ cùi xanh thắng các đội Chim giáo chủ và Chim vàng anh. Chim giáo chủ thắng đội Chim vàng anh. Hãy

2. Một số định nghĩa

- **Đồ thị con (***subgraph* **)**: Cho 2 đồ thị (cùng có hướng hoặc cùng vô hướng) G=(V,E) và H=(X,U). H được gọi là đồ thị con của G nếu $X \subseteq V$ và $U \subseteq E$. Kí hiệu $H \leq G$

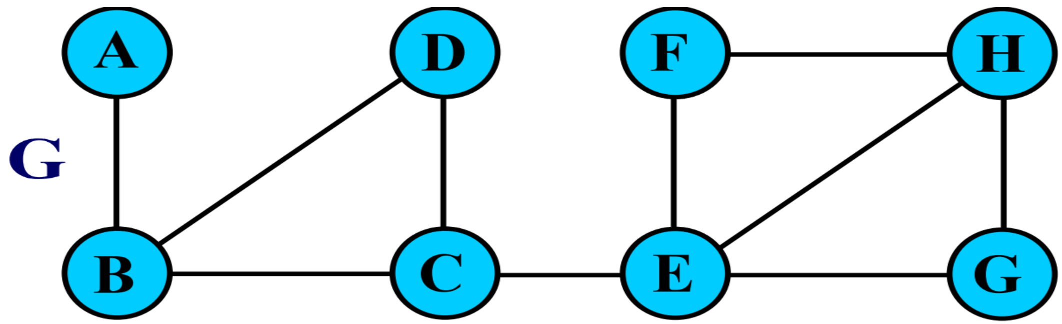
***<u>Ví dụ</u>*:**

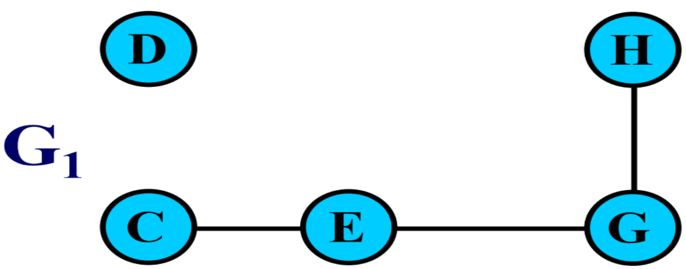


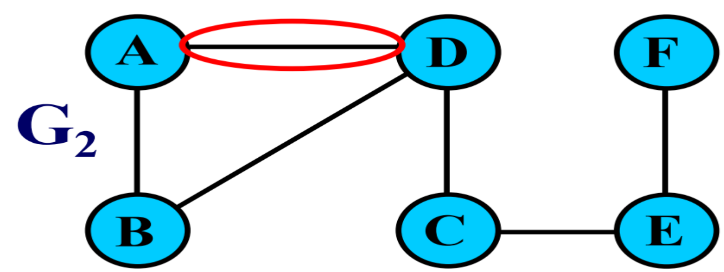
H là đồ thị con của G

Ví dụ đồ thị con

❖ **Ví dụ 2:** G1 là đồ thị con của G; G2 không là đồ thị con của G



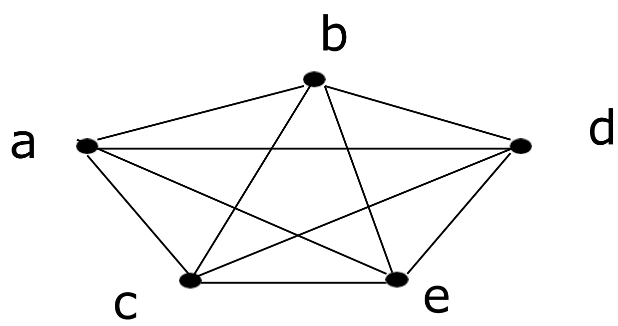




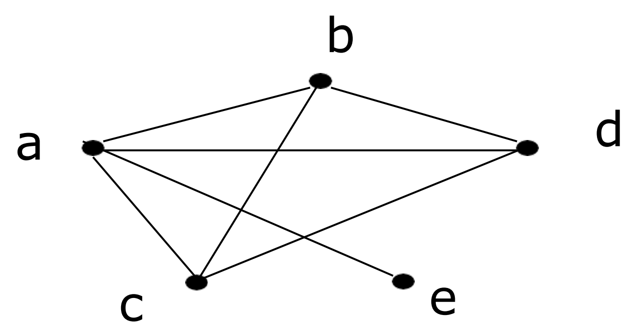
2. Một số định nghĩa

❖ **Đồ thị khung (*spanning subgraph*):** Cho 2 đồ thị G=(V,E) và H=(X,U), H≤G. Nếu X=V thì H gọi là đồ thị khung của G

<u>Ví dụ</u>:



G



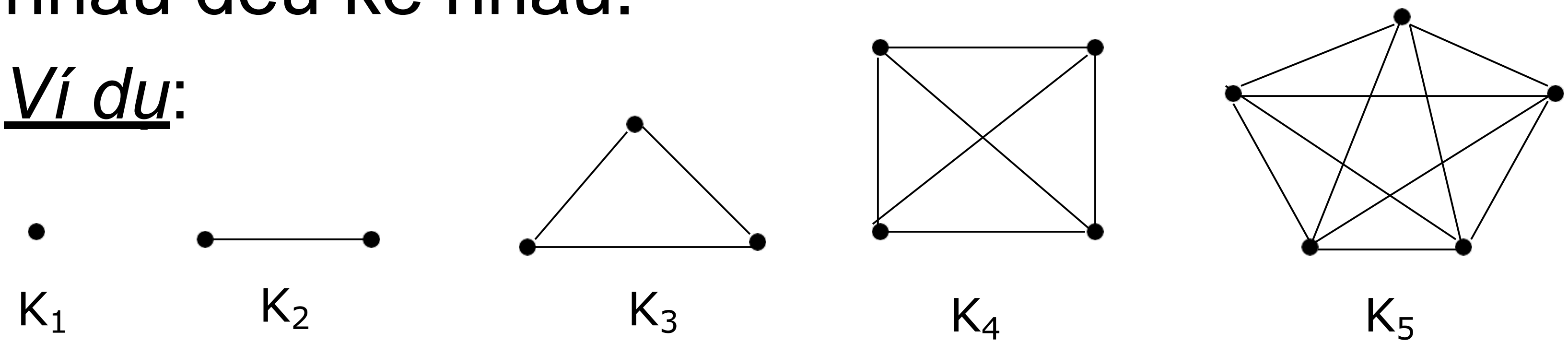
H

H là đồ thị khung của G

3. Một số đồ thị đặc biệt

❖ ***Đồ thị đủ (Complete Graph):*** Một đơn đồ thị vô hướng G=(V,E) với |V|=n, được gọi là đồ thị đủ cấp n(kí hiệu K_n) nếu với mỗi cặp đỉnh khác nhau đều kề nhau.

Ví dụ:

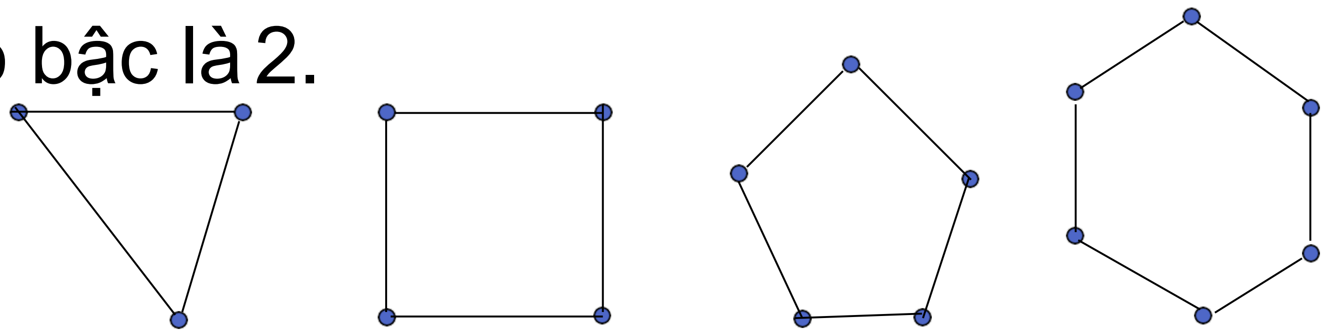


Một số đồ thị K_n (n=1,2,...,5)

Một đồ thị đủ cấp n thì có số cạnh là n(n-1)/2

3. Một số đồ thị đặc biệt

❖ **ĐN1: Đồ thị vòng** *(Cycles)* ***là*** Đơn đồ thị n đỉnh v_1, v_2, ..., v_n ($n \geq 3$) với n cạnh (v_1,v_2), (v_2,v_3), ..., (v_{n-1},v_n), (v_n,v_1) được gọi là đồ thị vòng, ký hiệu là C_n. Như vậy, mỗi đỉnh của C_n có bậc là 2.

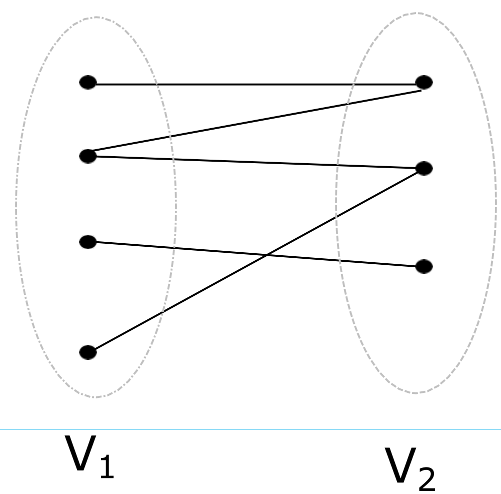


3.Một số đồ thị đặc biệt

- **ĐN2: Đồ thị vòng là** Đồ thị G=(V,E) được gọi là đồ thị vòng khi số lượng đỉnh của đồ thị >=3, bậc của các đỉnh đều bằng 2 và các cạnh nối với nhau thành 1 vòng khép kính, ký hiệu C_n

3.Một số đồ thị đặc biệt

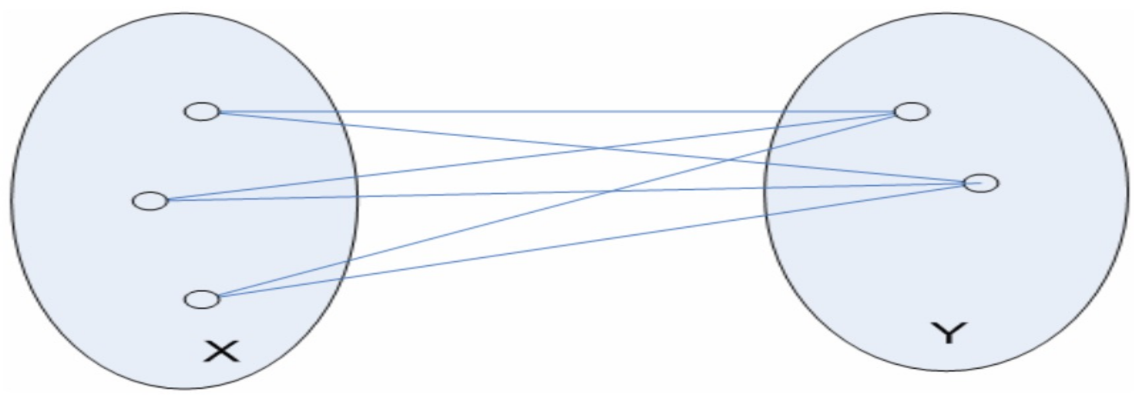
❖ **Đồ thị lưỡng phân/hai phía *(Bipartite Graphs):***
Đơn đồ thị G=(V,E) gọi là lưỡng phân nếu $V=V_1 \cup V_2$, với $V_1 \cap V_2=\varnothing$, $V_1 \neq \varnothing$, $V_2 \neq \varnothing$ và mỗi cạnh trong E đều nối một đỉnh trong V_1 với một đỉnh trong V_2.



3.Một số đồ thị đặc biệt

❖ **Đồ thị lưỡng phân đầy đủ (Complete Bipartite Graphs):** Đơn đồ thị G = (X ∪ Y, E) được gọi là đồ thị hai phía đầy đủ nếu: Mỗi đỉnh thuộc X sẽ được nối với mỗi đỉnh thuộc Y. Nếu |X| = m và |Y| = n thì ta sẽ ký hiệu là: $K_{m, n}$

Số cạnh của Đồ thị hai phía đầy đủ ?



3.Một số đồ thị đặc biệt

- **Định lý:** Đơn đồ thị G = (V, E) là đồ thị hai phía khi và chỉ khi nó không chứa chu trình độ dài lẻ

- <u>Chứng minh:</u>

$\forall$ Đồ thị hai phía

$\Rightarrow$ Không chứa chu trình độ dài lẻ

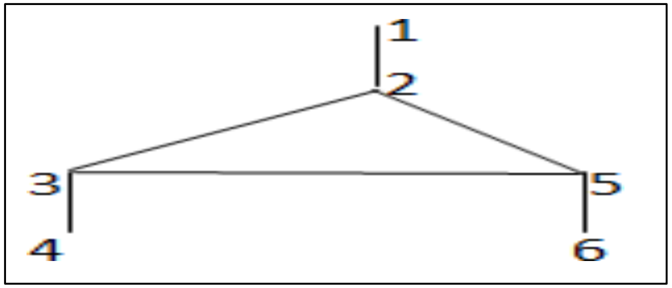
$\forall$ Đồ thị, không chứa chu trình độ dài lẻ

$\Rightarrow$ hai phía

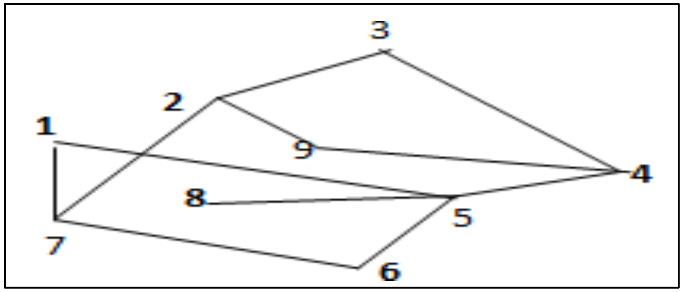
3. Một số đồ thị đặc biệt

❖ **Định lý**: Một đơn đồ thị là lưỡng phân nếu và chỉ nếu có thể dùng 1 trong 2 màu khác nhau cho trước để gán cho mỗi đỉnh sao cho không có 2 đỉnh kề nhau có chung một màu

Ví dụ: Đồ thị nào sau đây là lưỡng phân?



G

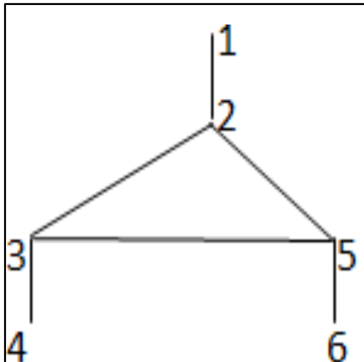


H

THUẬT TOÁN KIỂM TRA ĐỒ THỊ LIÊN THÔNG LƯỠNG PHÂN

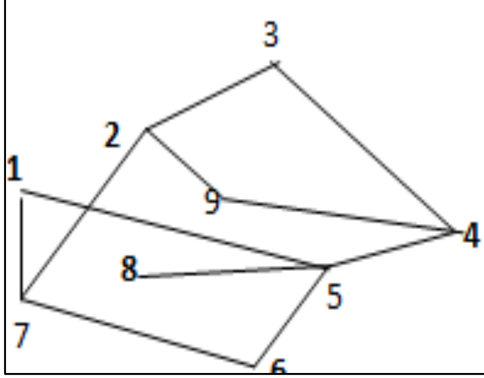
- B1: Chọn v là một đỉnh bất kì của đồ thị. Đặt X={v};
- B2: Tìm Y là tập đỉnh kề của các đỉnh trong X. Nếu X giao Y khác rỗng thì đồ thị không phải là lưỡng phân. Kết thúc. Ngược lại xuống B3.
- B3: Tìm T là tập các đỉnh kề của các đỉnh trong Y. Nếu T giao Y khác rỗng thì đồ thị không phải là lưỡng phân. Kết thúc. Nếu T=X thì đồ thị là lưỡng phân, kết thúc. Ngược lại gán X=T và lặp lại B2.

VÍ DỤ KIỂM TRA ĐỒ THỊ LIÊN THÔNG LƯỠNG PHÂN



X	Y (Các đỉnh kề X)	T (Các đỉnh kề Y)
1	2	1,3,5
1,3,5	2,3,4,5,6	

$X \cap Y \neq \phi$: đồ thị không phải là hai phía



X	Y (Các đỉnh kề X)	T (Các đỉnh kề Y)
1	5,7	1,2,4,6,8
1,2,4,6,8	3,5,7,9	1,2,4,6,8

T=X : đồ thị là hai phía

X={1,2,4,6,8}; Y={3,5,7,9}

THUẬT TOÁN KIỂM TRA ĐỒ THỊ LIÊN THÔNG LƯỠNG PHÂN

1. Chọn v là đỉnh bất kỳ. Đặt $X = \{v\}$
2. $Y = \{\, u \mid u$ kề với $v, \forall\, v \in X\}$
3. Nếu $X \cap Y \neq \varnothing \Rightarrow$ G không là đồ thị hai phía
4. Ngược lại, đặt $X := Y$ Quay trở lại 2.
5. Nếu tất cả các đỉnh được xét hết mà không xảy ra 3. thì G là đồ thị hai phía. Ngược lại G không là đồ thị hai phía.

VÍ DỤ KIỂM TRA ĐỒ THỊ LIÊN THÔNG LƯỠNG PHÂN

- Ví dụ:

  $X = \{1\}$

  $Y = \{5\}$, $X \cap Y = \emptyset$, $\Rightarrow X := Y$

  $Y = \{1, 2\}$, $X \cap Y = \emptyset$, $\Rightarrow X := Y$

  $Y = \{5, 6, 7\}$, $X \cap Y = \emptyset$, $\Rightarrow X := Y$

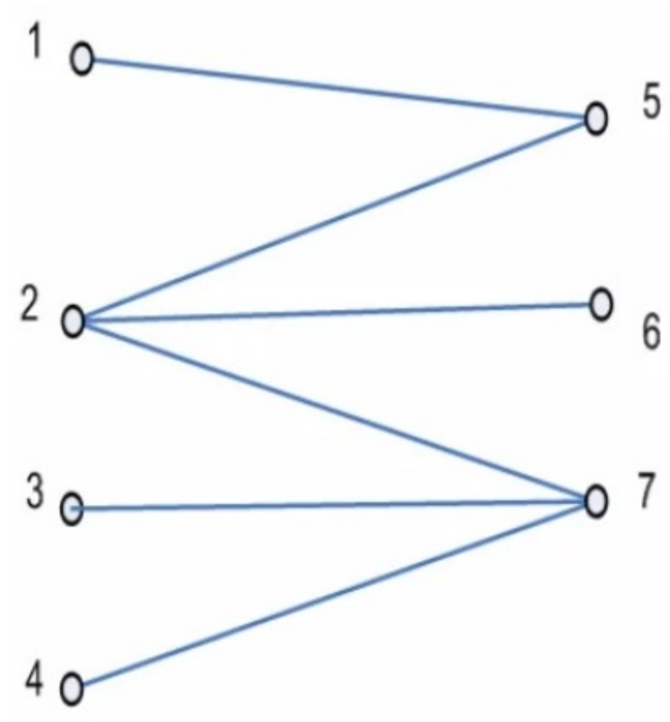
  $Y = \{1, 2, 3, 4\}$

  DỪNG

  Khi đó đồ thị là hai phía:

  $X = \{1, 2, 3, 4\}$

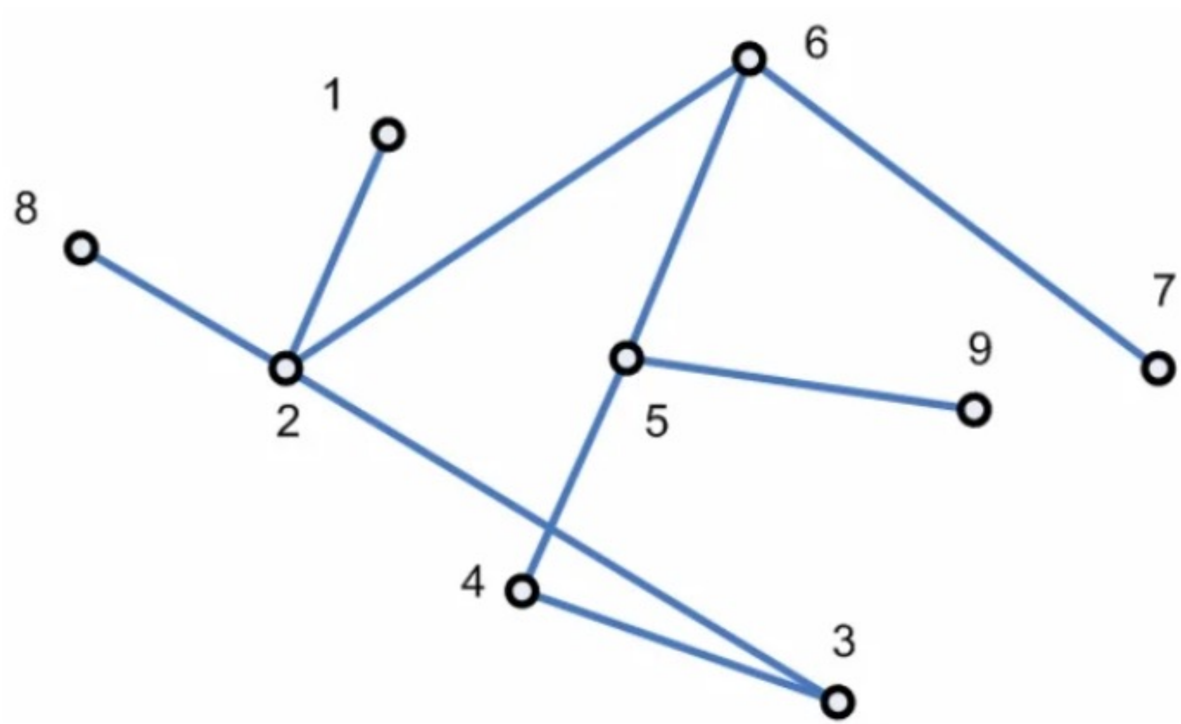
  $Y = \{5, 6, 7\}$



BÀI TẬP KIỂM TRA ĐỒ THỊ LIÊN THÔNG LƯỠNG PHÂN

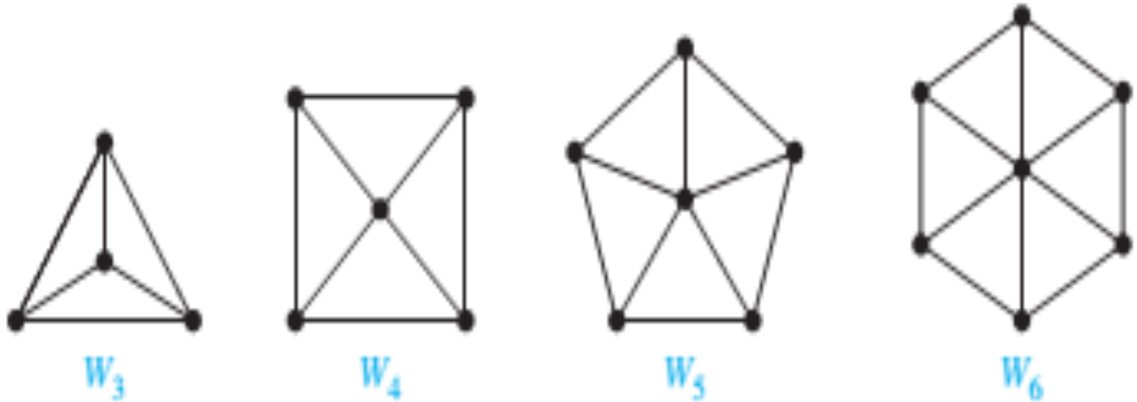
❖ Bài tập:

Kiểm tra đồ thị sau có phải là đồ thị hai phía hay không?



3. Một số đồ thị đặc biệt

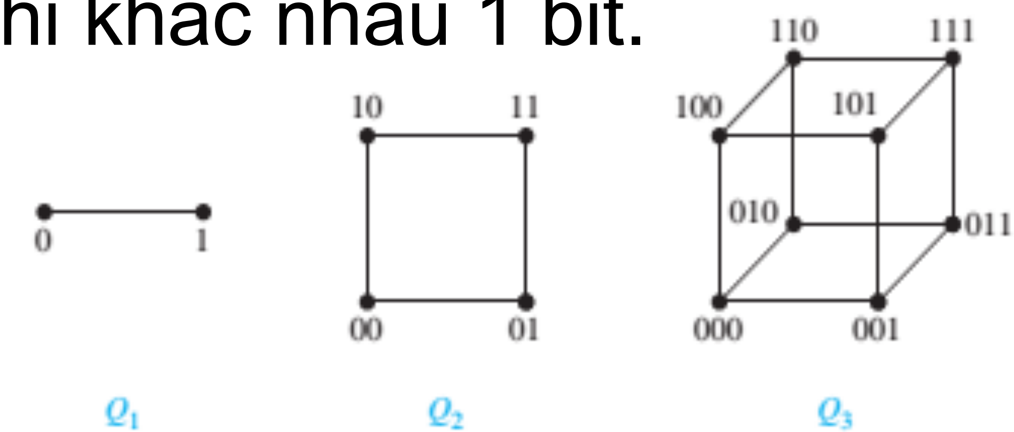
- **Đồ thị bánh xe *(Wheels)*:** Kí hiệu W_n , nhận được từ đồ thị C_n ($n \geq 3$) bằng cách thêm một đỉnh mới và bổ sung các cạnh nối đỉnh vừa thêm với các đỉnh trong C_n.
- Ví dụ:



Một số đồ thị W_n, ($3 \leq n \leq 6$)

3. Một số đồ thị đặc biệt

- **Đồ thị siêu khối (*n*-Cubes):** Đồ thị siêu khối $k=2^n$ đỉnh là đồ thị có các đỉnh được đánh số bằng các chuỗi nhị phân độ dài n.
- Ký hiệu: Q_n
- Hai đỉnh kề nhau nếu 2 chuỗi nhị phân tương ứng chỉ khác nhau 1 bit.
- Ví dụ:



3. Một số đồ thị đặc biệt

- Các dạng đồ thị đặc biệt
  - Đồ thị đầy đủ (K_n)
  - Đồ thị vòng (C_n)
    - Đồ thị bánh xe (W_n)
  - Đồ thị siêu khối (Q_n)
  - Đồ thị hai phía
    - Đồ thị hai phía đầy đủ ($K_{m,n}$)
  - Đồ thị phẳng

4. Định lý bắt tay (The handshaking Theorem)

- **Định lý**: Cho đồ thị vô hướng G=(V,E) với m cạnh. *Khi đó tổng tất cả các bậc của đỉnh trong V bằng 2m.*

$$\sum_{v \in V} \deg(v) = 2m$$

- **Ví dụ**: Đồ thị G có 6 đỉnh và tất cả các đỉnh có bậc là 6. Tính số cạnh của G?

4. Định lý 1: Định lý bắt tay

Hệ quả:

i)Tổng bậc của các đỉnh bậc lẻ trong một đồ thị vô hướng G là một số chẵn

ii)Mọi đồ thị vô hướng đều có một số chẵn các đỉnh bậc lẻ

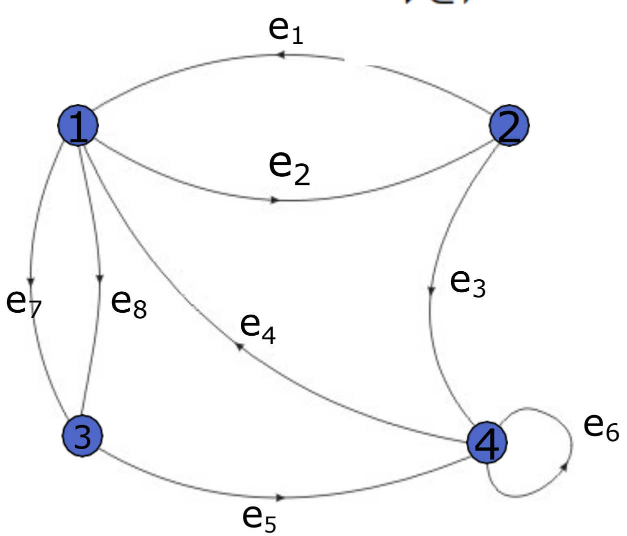
iii) Đồ thị K_n có $\frac{1}{2}n(n-1)$ cạnh

Định lý 2

- **Định lý**: G=(V,E) là đồ thị có hướng có m cạnh, ta có:

$$\sum_{v\in V} \deg^+(v) = \sum_{v\in V} \deg^-(v) = m$$

- **Ví dụ:**



m=|E|=8

$$\sum_{v\in V} \deg^+(v) = \deg^+(1)+\deg^+(2)+\deg^+(3)+\deg^+(4)$$
$$= 3+2+1+2 = 8$$

$$\sum_{v\in V} \deg^-(v) = \deg^-(1)+\deg^-(2)+\deg^-(3)+\deg^-(4)$$
$$= 2+1+2+3 = 8$$

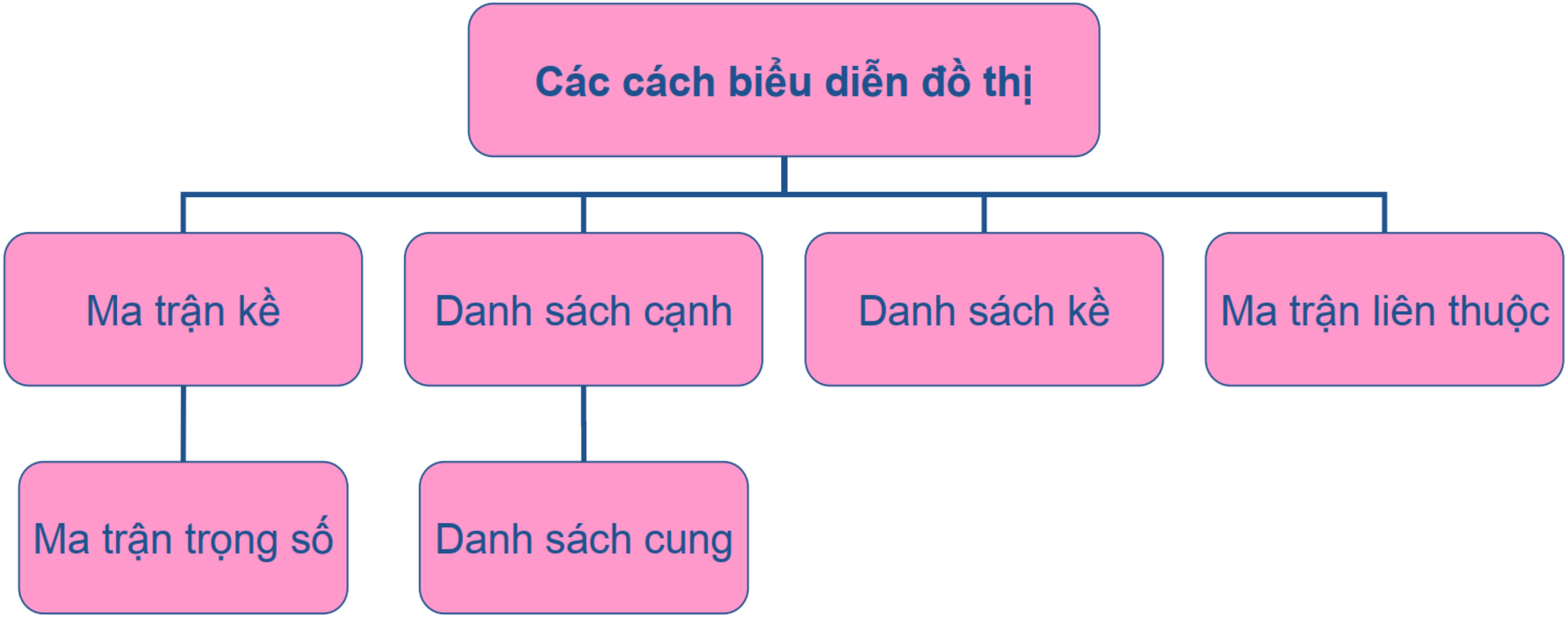
BÀI TẬP

- Bài 1: Vẽ đơn đồ thị khi biết bậc của các đỉnh lần lượt là:
  1. 1,2,2,3
  2. 4,3,3,2,2
- Bài 2: Vẽ đơn đồ thị có 6 đỉnh, trong đó có:
  1. 3 đỉnh bậc 3 và 3 đỉnh bậc 1
  2. Bậc các đỉnh lần lượt là 1,2,3,3,4,5
  3. Bậc các đỉnh lần lượt là 2,2,4,4,4,4
- Bài 3:Tìm số đỉnh của đồ thị G, biết rằng G có:
  1. 12 cạnh và bậc của tất cả các đỉnh =2
  2. 15 cạnh, trong đó 3 đỉnh bậc 4 và các đỉnh còn lại bậc 3,

4. BIỂU DIỄN ĐỒ THỊ TRÊN MÁY TÍNH

1. Biểu diễn đồ thị bằng danh sách kề
2. Biểu diễn đồ thị bằng ma trận kề
3. Sự đẳng cấu của các đồ thị

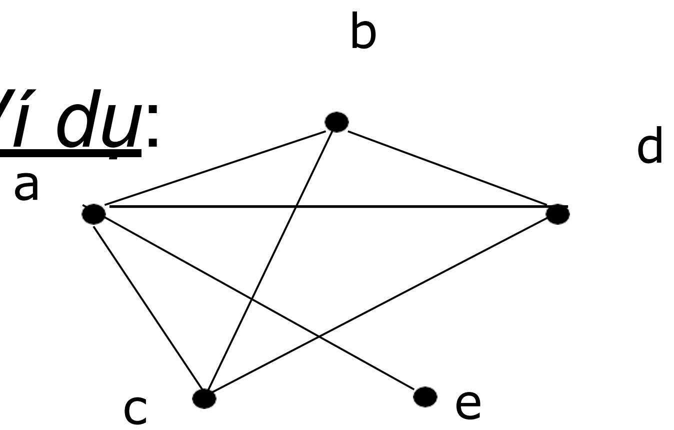
4. BIỂU DIỄN ĐỒ THỊ TRÊN MÁY TÍNH



4.1 Biểu diễn đồ thị bằng danh sách kề

❖ G=(V,E) không có cạnh song song (G không có cạnh song song cùng chiều nếu G có hướng). G có thể được biểu diễn bằng cách liệt kê tất cả các đỉnh của G, mỗi đỉnh liệt kê các đỉnh kề với nó

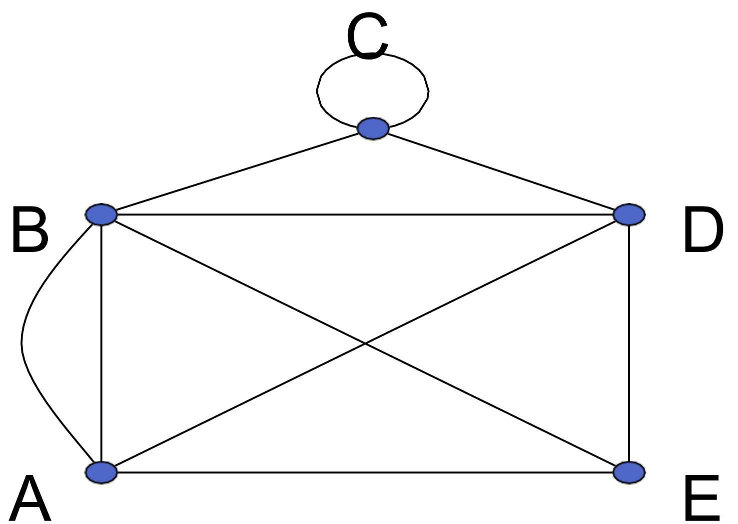
❖ <u>*Ví dụ*</u>:



Đỉnh	Các đỉnh kề
a	b,d,e,c
b	a,c,d
c	a,b,d
d	a,b,c
e	a

4.1 Biểu diễn đồ thị bằng danh sách kề

Lưu ý: Biểu diễn bằng danh sách kề khá cồng kềnh, đặc biệt khi G có nhiều cạnh ít được dùng trong các thuật toán về đồ thị



A: B B D E

B: A A C D E

C: C C B D

D: A B C E

E: A B D

4.1 Biểu diễn đồ thị bằng ma trận kề (Adjacency Matrix)

- Cho đồ thị G=(V,E), tập đỉnh $V=\{v_1, v_2, \ldots, v_n\}$ và tập cạnh/cung $E=\{e_1, e_2, \ldots, e_m\}$. Ma trận kề của G ứng với thứ tự các đỉnh $v_1, v_2, \ldots, v_n$ là ma trận vuông cấp n được định nghĩa như sau:

$$A = (a_{ij})_{1 \le i,j \le n}$$

Với a_{ij}=số cạnh/cung nối từ đỉnh v_i đến đỉnh v_j

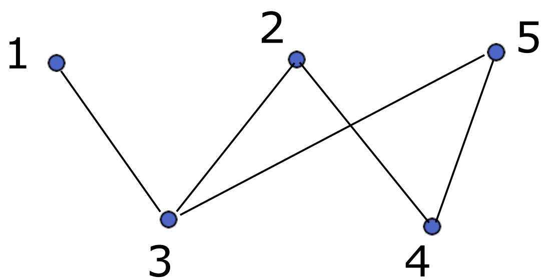
$$a_{i,j} = \begin{cases} 0, & if \ (i,j) \notin E \\ 1, & if \ (i,j) \in E \end{cases}$$

✍ ***Trong ma trận kề một khuyên tại 1 điểm được biểu diễn là 1***

✍ ***Nếu G là đồ thị vô hướng thì A đối xứng***

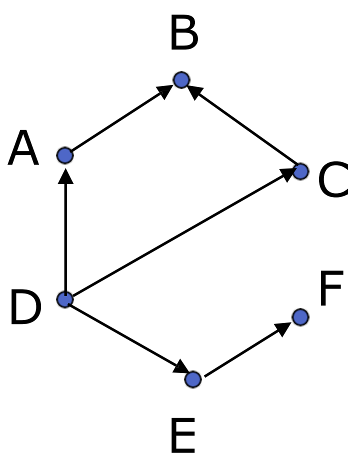
4.1 Biểu diễn đồ thị bằng ma trận kề

Ví dụ:



Ma trận kề

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$



$$\begin{array}{c c} & \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \\ E \\ F \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

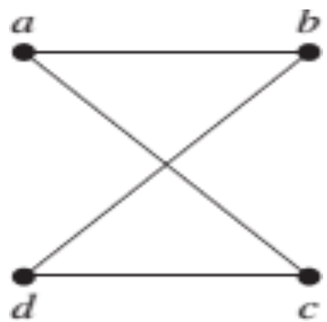
4.1 Biểu diễn đồ thị bằng ma trận kề

<u>*Ví dụ*</u>: Cho G=(V,E) với ma trận kề như sau:

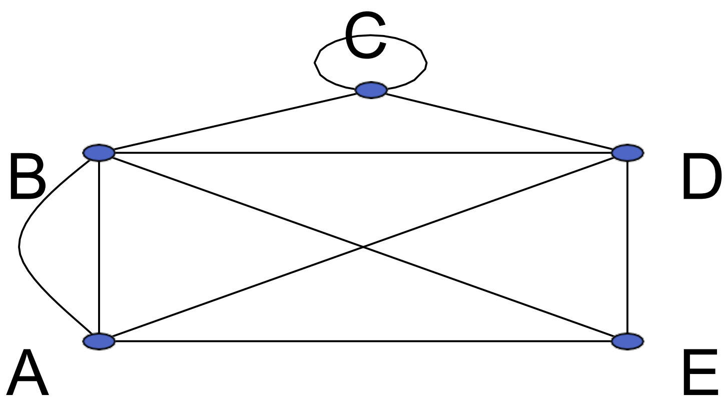
$$M=\begin{array}{cc} & \begin{array}{ccccc}A & B & C & D & E\end{array} \\ \begin{array}{c}A\\B\\C\\D\\E\end{array} & \begin{bmatrix}0 & 0 & 1 & 0 & 0\\ 0 & 0 & 2 & 1 & 0\\ 1 & 2 & 0 & 0 & 1\\ 0 & 1 & 0 & 0 & 1\\ 0 & 0 & 1 & 1 & 0\end{bmatrix}\end{array}$$

- Đỉnh A có bậc 1
- Đỉnh B có bậc 3
- Đỉnh C có bậc 4
- Đỉnh D có bậc 2
- Đỉnh E có bậc 2

4.1 Biểu diễn đồ thị bằng ma trận kề



$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$



$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} 0 & 2 & 0 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

4.2 Biểu diễn đồ thị bằng ma trận liên thuộc (Incidence Matrix)

- Cho **đồ thị vô hướng** $G=(V,E)$, $V=\{v_1,v_2,\ldots,v_n\}$, $E=\{e_1,e_2,\ldots,e_m\}$. Ma trận liên thuộc của G là ma trận cấp $n \times m$ được định nghĩa như sau:

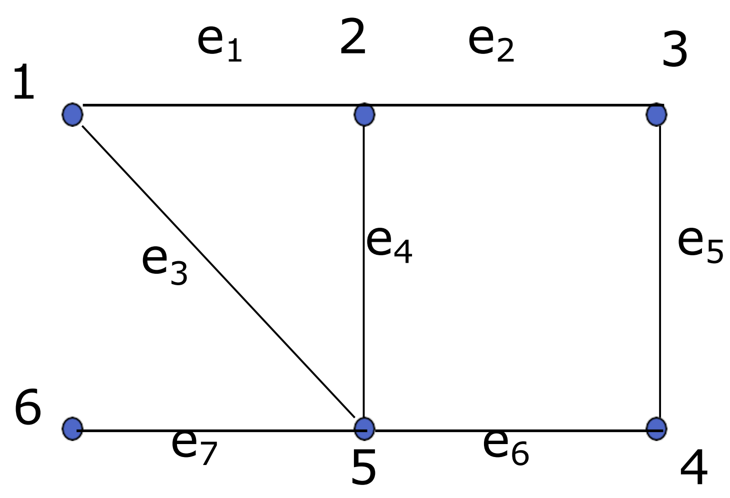
$$M = (m_{ij})_{1 \le i \le n,\, 1 \le j \le m}$$

$$m_{ij} = \begin{cases} \textbf{1 nếu } e_j \textbf{ liên thuộc với } v_i \\ \textbf{0 nếu } e_j \textbf{ không liên thuộc với } v_i \end{cases}$$

Cụ thể, M là ma trận kề, trong đó: i (dòng) là các đỉnh, j (cột) tương ứng với các cạnh

4.2 Biểu diễn đồ thị bằng ma trận liên thuộc

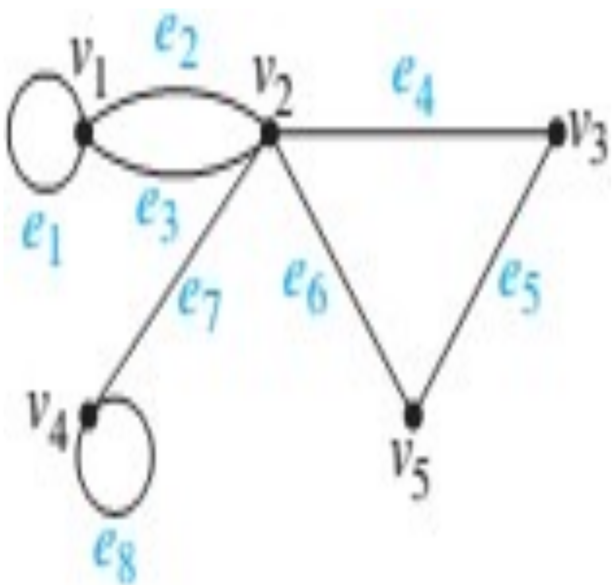
❖ Ví dụ:



$$
\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{array} \\
\begin{bmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\end{array}
$$

4.2 Biểu diễn đồ thị bằng ma trận liên thuộc

Đồ thị



Ma trận liên thuộc

$$
\begin{array}{c}
\\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{bmatrix}
\end{array}
$$

4.2. Biểu diễn đồ thị bằng ma trận liên thuộc

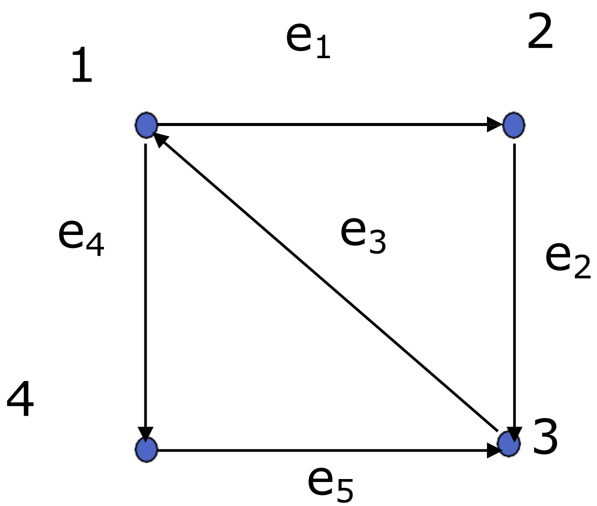
❖ Cho **đồ thị có hướng** G=(V,E), $V=\{v_1,v_2,\ldots,v_n\}$, $E=\{e_1, e_2,\ldots, e_m\}$. Ma trận liên thuộc của G là ma trận cấp $n\times m$ được xác định như sau:

$$M = (m_{ij})_{1\le i\le n, 1\le j\le m}$$

$$m_{ij}= \begin{cases} 1 \text{ nếu } e_j \text{ rời khỏi đỉnh } i \\ 0 \text{ nếu } e_j \text{ không liên thuộc với } v_i \\ -1 \text{ nếu } e_j \text{ đến đỉnh } i \end{cases}$$

4.2 Biểu diễn đồ thị bằng ma trận liên thuộc

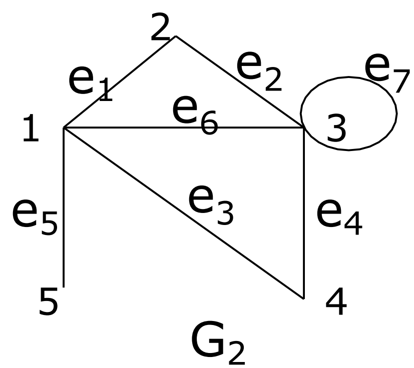
Ví dụ:

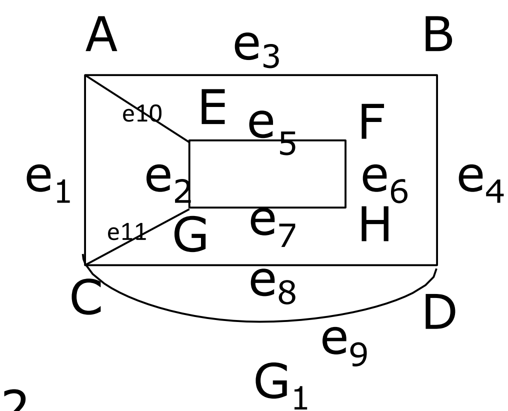


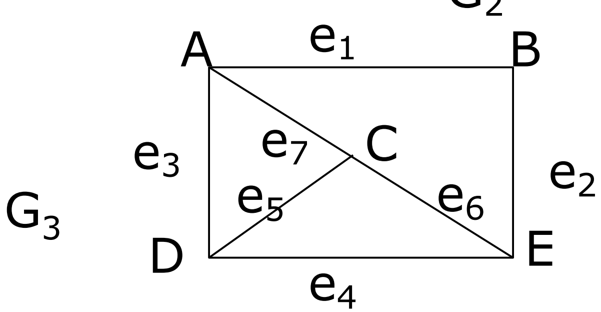
$$
\begin{array}{c c}
 & \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 \end{matrix} \\
\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 1 & 0 & -1 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}
\end{array}
$$

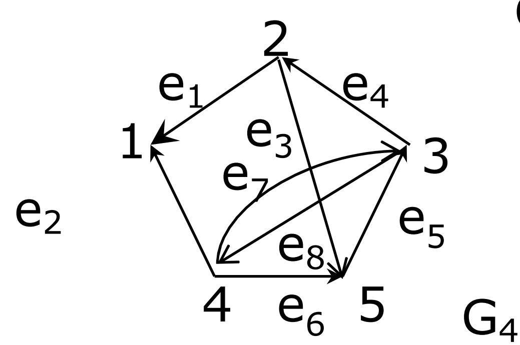
Bài tập

❖ Biểu diễn các đồ thị sau bằng ma trận kề, ma trận liên thuộc









4.3 Đồ thị đẳng cấu (Graph Isomorphism)

❖ **Định nghĩa**: Hai đơn đồ thị $G=(V_1,E_1)$ và $G'=(V_2,E_2)$ gọi là đẳng cấu với nhau nếu có:

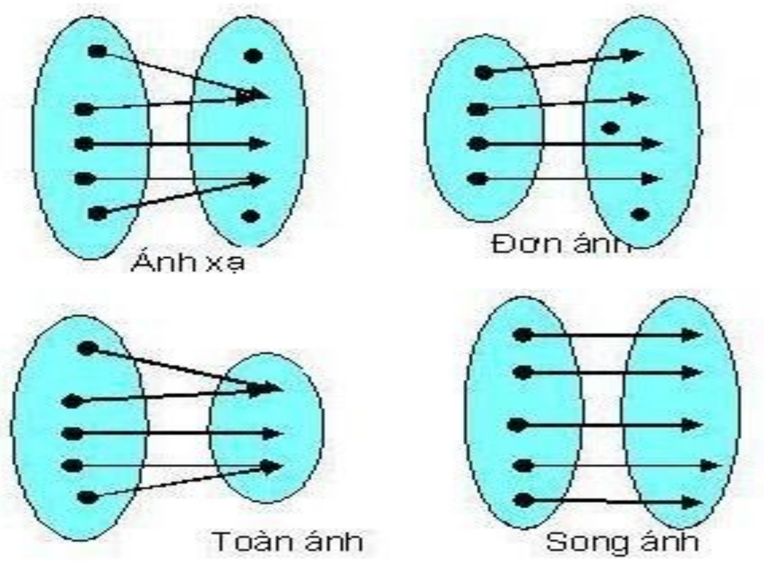
- **Điều kiện cần:**
  - Cùng số lượng đỉnh
  - Cùng số lượng đỉnh bậc k, $\forall$k nguyên và ≥ 0
  - Cùng số cạnh
  - Cùng số thành phần

4.3 Đồ thị đẳng cấu

❖ **Điều kiện đủ**

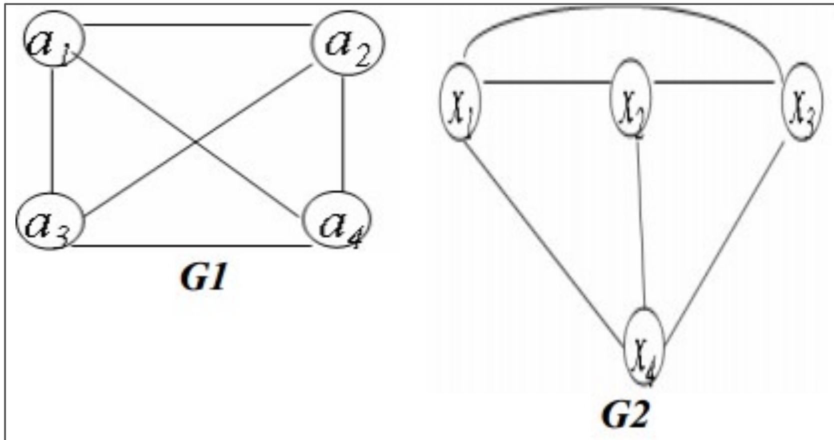
- **Tồn tại song ánh $f: V_1 \leftrightarrow V_2$ sao cho:**

uv là cạnh của G $\Leftrightarrow$ f(u)f(v) là cạnh của G'



4.3 Đồ thị đẳng cấu

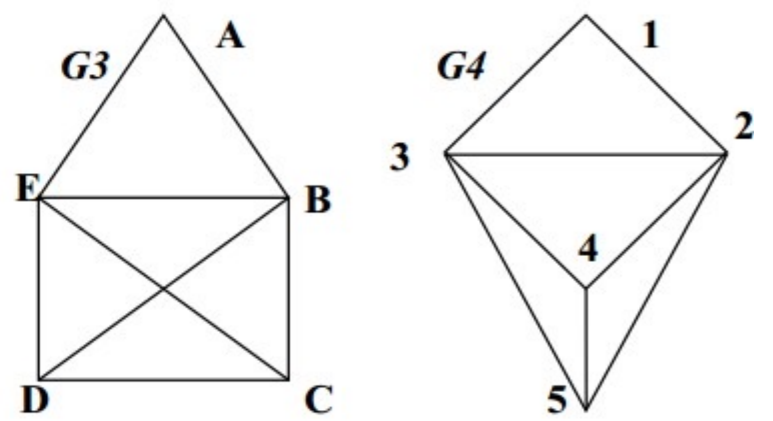
❖ Ví dụ



Hình 1.28. G1 và G2 đẳng hình

hay

a1⇔x1	a1⇔x2
a2⇔x2	a2⇔x3
a3⇔x3	a3⇔x1
a4⇔x4	a4⇔x4



Hình 1.29. G3 và G4 đẳng hình

A ⇔ 1
B ⇔ 2
C ⇔ 4
D ⇔ 5
E ⇔ 3

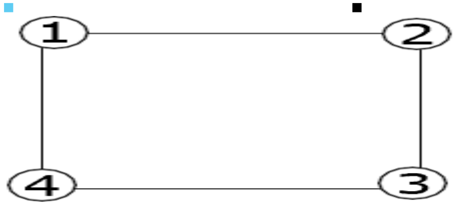
4.3 Đồ thị đẳng cấu

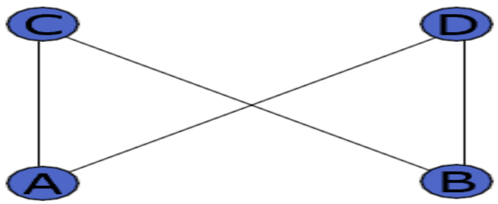
❖ Các bước để chứng minh 2 đồ thị đẳng cấu:

- **Bước 1.** Lập bảng và sắp xếp bậc các đỉnh của đồ thị tăng dần, số đỉnh, số cạnh, số thành phần.
- **Bước 2:** Định nghĩa hàm f
- **Bước 3:** Lập ma trận kề đối chứng song ánh (thứ tự đỉnh theo bước 2)

4.3 Đồ thị đẳng cấu

Ví dụ: hai đồ thị sau đây có đẳng cấu hay ko? Giải thích?





- BƯỚC 1: XÁC ĐỊNH BẬC CỦA CÁC ĐỈNH

ĐỒ THỊ	BẬC 2	SỐ CẠNH	SỐ THÀNH PHẦN	Số đỉnh
G1	1,2,3,4	4	1	4
G2	C,B,D,A	4	1	4

- BƯỚC 2: **Định nghĩa hàm f**

F(G1)	1	2	3	4
G2	C	B	D	A

4.3 Đồ thị đẳng cấu

- **Bước 3: Lập ma trận kề đối chứng song ánh theo bước 2**

	1	2	3	4
C	0	1	0	1
B	1	0	1	0
D	0	1	0	1
A	1	0	1	0

HÌNH 1

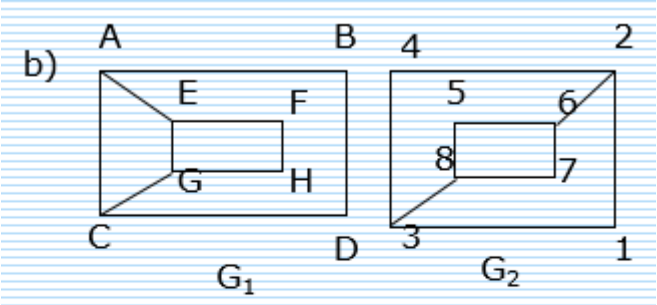
	C	B	D	A
1	0	1	0	1
2	1	0	1	0
3	0	1	0	1
4	1	0	1	0

HÌNH 2

- **Kết luận: Hình 1 và Hình 2 đẳng cấu**

4.3 Đồ thị đẳng cấu

❖ Ví dụ: Chứng minh 2 đồ thị sau có đẳng cấu hay không?



❖ BƯỚC 1: XÁC ĐỊNH BẬC CỦA CÁC ĐỈNH

ĐỒ THỊ	BẬC 2	BẬC 3	SỐ CẠNH	SỐ THÀNH PHẦN	Số đỉnh
G1	B,D,F,H	A,C,E,G	10	1	8
G2	1,4,5,7	2,3,6,8	10	1	8

4.3 Đồ thị đẳng cấu

- BƯỚC 2: Định nghĩa hàm f→ không định nghĩa được

F (G1)	B	D	F	H	A	C	E	G
F (G2)								

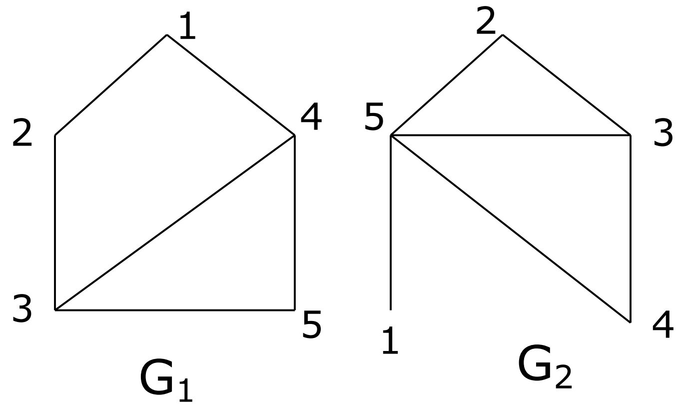
Bởi vì, deg(B)=2, đỉnh B kề với đỉnh deg(A)=3 và deg(D)=2; tuy nhiên trong G2 có 4 đỉnh bậc 2 (4,1,5,7) nhưng trong 4 đỉnh này không có bất kỳ đỉnh nào có 2 đỉnh kề với nó là bậc 2 và bậc 3. Nên không thể định nghĩa được ánh xạ cho đỉnh B

- BƯỚC 3: Không lập ma trận kề đối chứng song ánh theo bước 2.
- Kết luận: Hình G1 và Hình G2 KHÔNG đẳng cấu

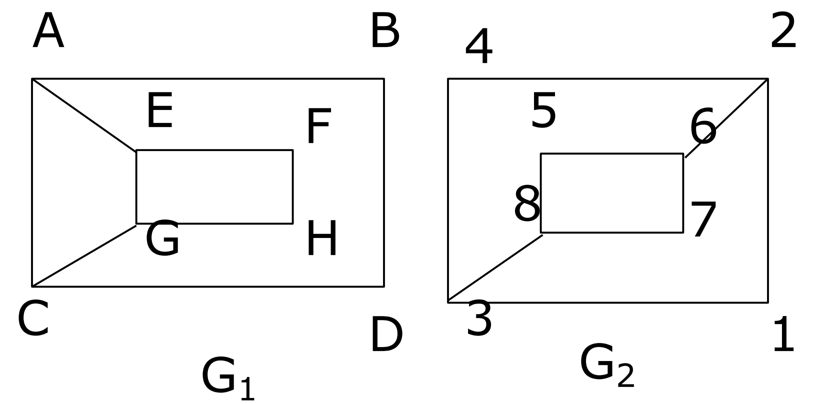
4.3 Đồ thị đẳng cấu

Ví dụ: Các cặp đồ thị sau đây có phải đẳng cấu không?

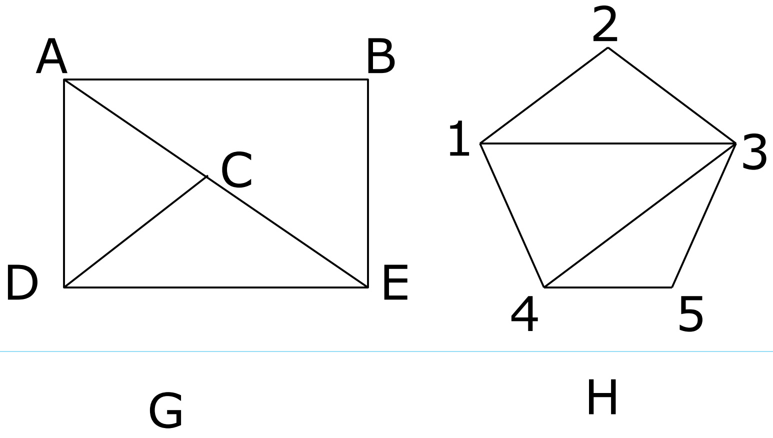
a)



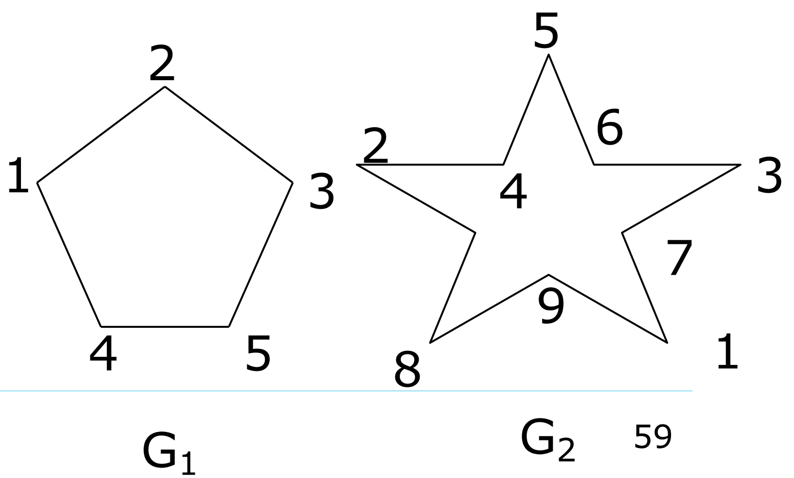
b)



c)

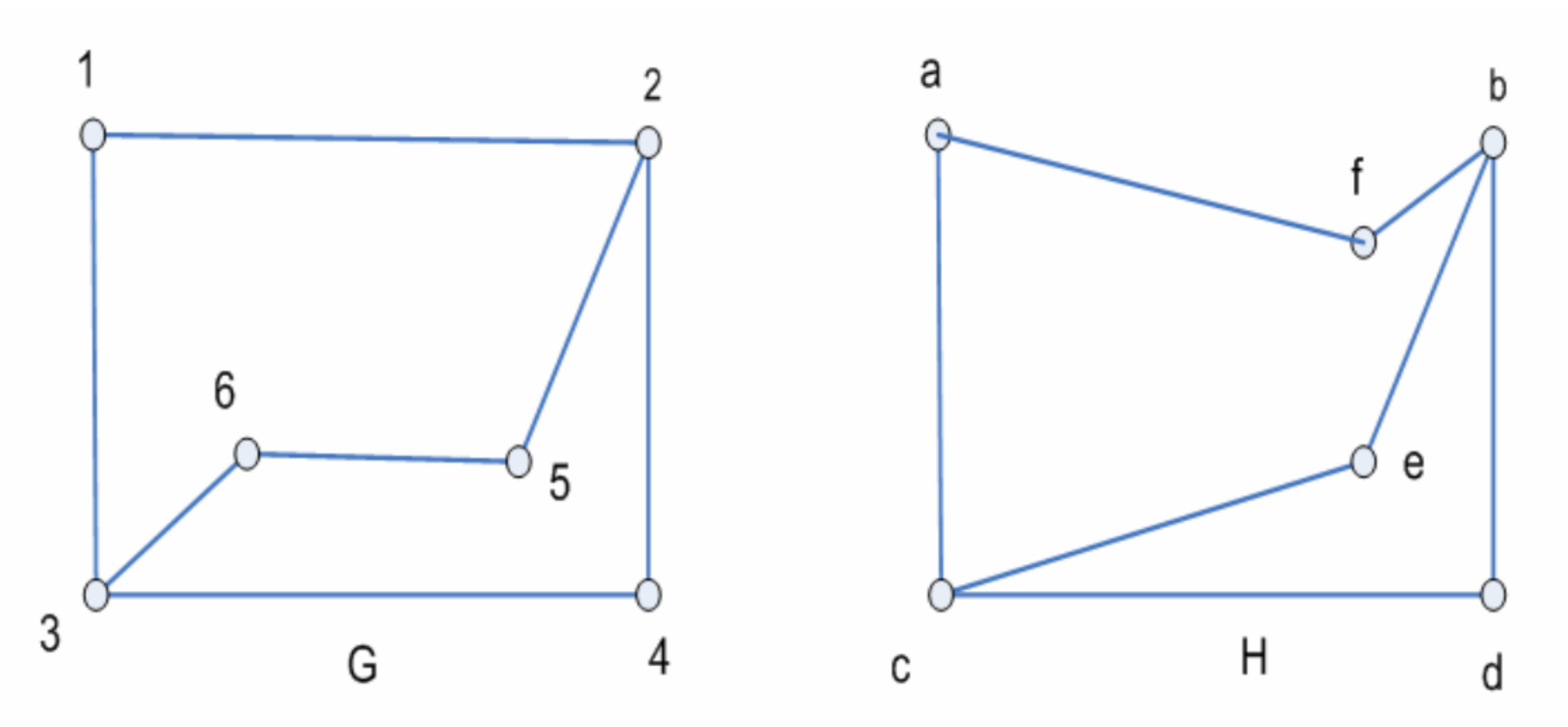


d)



4.3 Đồ thị đẳng cấu

Ví dụ: H và G có đẳng cấu hay không? chứng minh?



5. Đường đi và chu trình

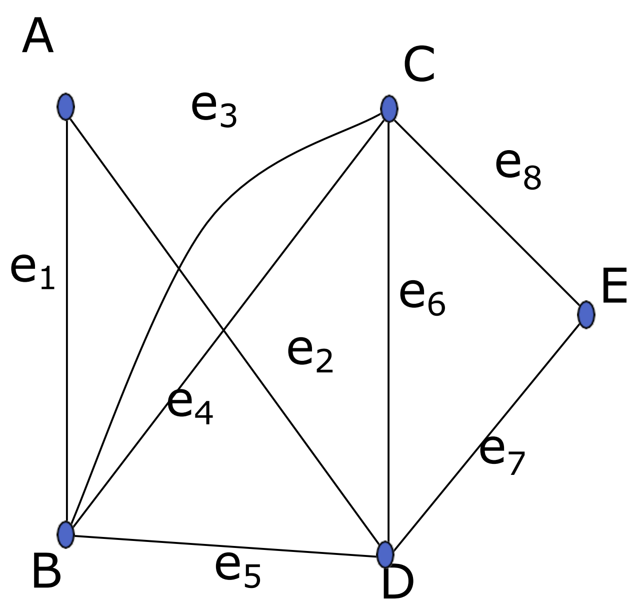
- **Đường đi** (Path) có độ dài k từ đỉnh u đến đỉnh v của đồ thị G=(V,E) là dãy các đỉnh $x_0, x_1, x_2, \ldots, x_k$; $x_0=u$, $x_k=v$ và (x_i, x_{i+1}) là một cạnh/cung của G. Có thể biểu diễn đường đi bởi dãy các đỉnh cạnh/cung liên tiếp:

$$\mathbf{P=(x_0, e_1, x_1, e_2, \ldots, x_{k-1}, e_k, x_k)}$$

Với: $x_0=u$, $x_k=v$, $e_i=(x_{i-1}, x_i) \in E$

5. Đường đi và chu trình

❖ **Ví dụ:**



- (A,e_1,B,e_4,C,e_6,D) là một đường đi có độ dài 3 từ đỉnh A và đỉnh D
- $(E,e_7,D,e_6,C,e_4,B,e_1,A)$ là đường đi từ E đến A có độ dài 4

5. Đường đi và chu trình

- Đường đi không có lặp lại các cạnh/cung gọi là đường đi đơn
- Đường đi không có lặp lại đỉnh gọi là đường sơ cấp

Ví dụ:

- (A,e_1,B,e_4,C,e_6,D) là một đường đi sơ cấp có độ dài 3 từ đỉnh A và đỉnh D
- $(A,e_1,B,e_5,D,e_5,B,e_4,C)$ không phải là đường đi đơn
- $(A,e_1,B,e_4,C,e_3,B,e_5,D)$ là đường đi đơn từ A đến D nhưng không phải là là đường đi sơ cấp

- Mọi đường đi sơ cấp đều là đường đi đơn

5. Đường đi và chu trình

❖ ***Chu trình:*** Đường đi có đỉnh đầu trùng với đỉnh cuối gọi là chu trình.

•Chu trình gọi là **đơn** nếu không có sự lặp lại các cạnh (hay cung).

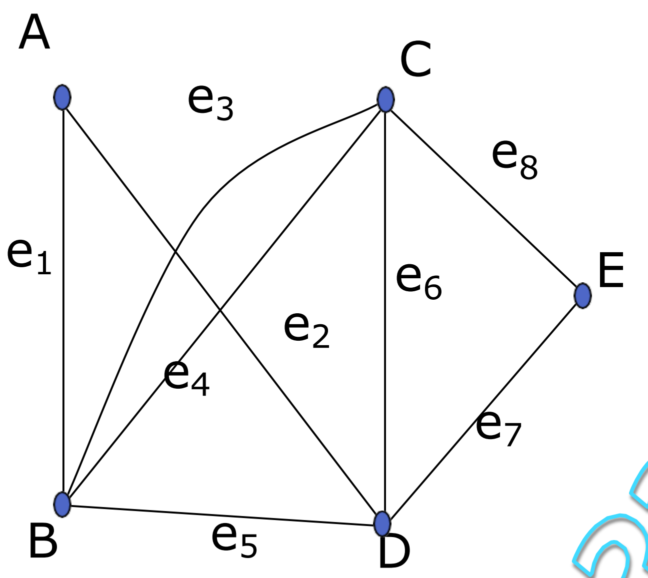
•Chu trình gọi là **sơ cấp** nếu không có sự lặp lại các đỉnh.

5. Đường đi và chu trình (tt)

- ***Định lý***: Cho đồ thị G=(V,E) có ma trận kề là A. Số đường đi khác nhau có độ dài r từ đỉnh i đến đỉnh j của đơn đồ thị G là giá trị của phần tử a_{ij} trong ma trận A^r

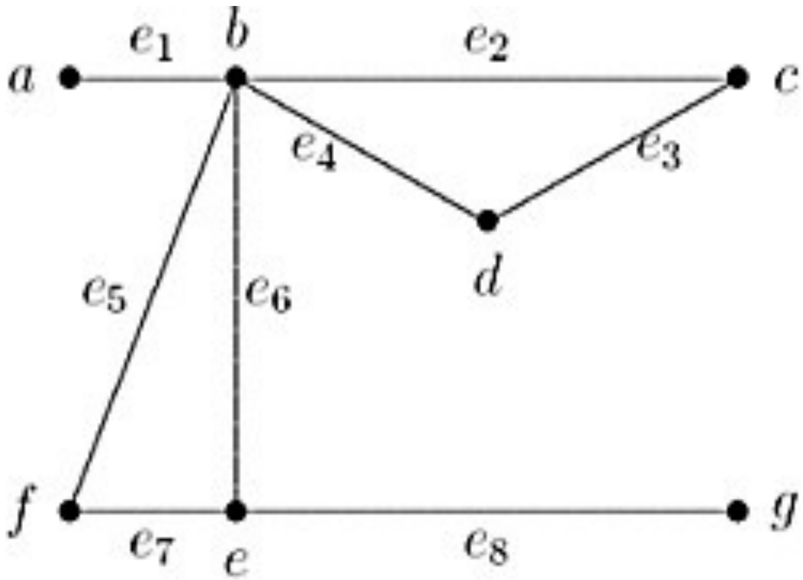
5. Đường đi và chu trình

❖ Ví dụ: Cho đồ thị G như hình dưới. Số đường đi có độ dài là 3 đi từ A đến D?



		A	B	C	D	E
A=	A	0	1	0	1	0
	B	1	0	2	1	0
	C	0	2	0	1	1
	D	1	1	1	0	1
	E	0	0	1	1	0

5. Đường đi và chu trình



- Chu trình sơ cấp:
  (b,c,d,b)
  (b,f,e,b)

Chu trình sơ cấp nào không?

nếu không có sự lặp lại các đỉnh

5. Đường đi và chu trình

Bài 1: Các danh sách đỉnh sau đây có tạo nên đường đi trong đồ thị bên dưới hay không? Đường đi nào là đơn? Đường đi nào là chu trình? Độ dài của các đường đi này là bao nhiêu?

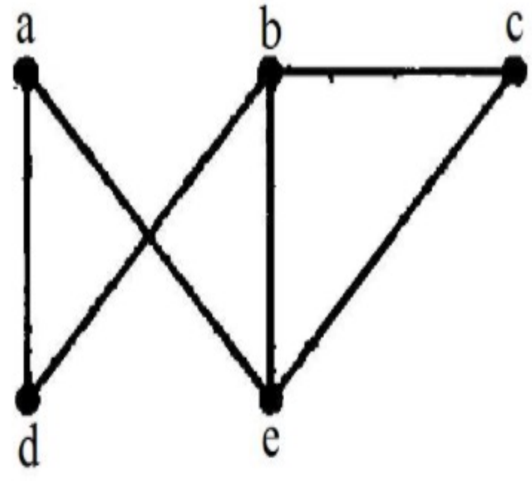
a) (a, e, b, c, b)→ có đường đi với độ dài là 4, ko chu trình, không đơn

b) (a, e, a, d, b, c, a)→ không phải đường

c) (e, b, a, d, b, e) → không phải đường

d) (c, b, d, a, e, c)

→ Chu trình đón với độ dài là 5



5. Đường đi và chu trình

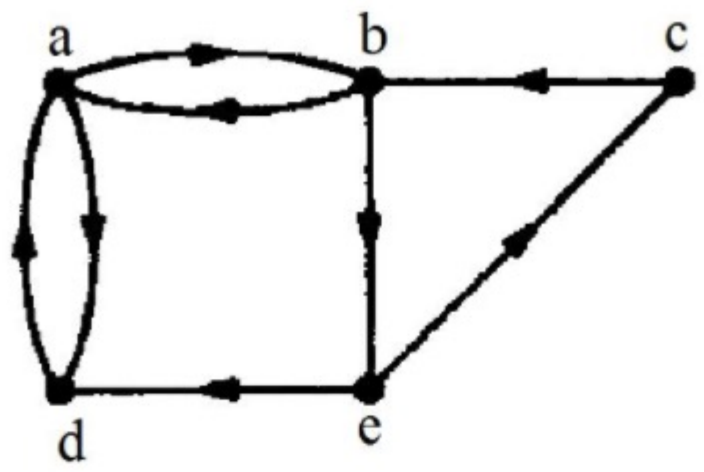
Bài 2. Các danh sách đỉnh sau đây có tạo nên đường đi trong đồ thị bên dưới hay không? Đường đi nào là đơn? Đường đi nào là chu trình? Độ dài của các đường đi này là bao nhiêu?

a) (a, b, e, c, b)

b) (a, d, a, d, a)

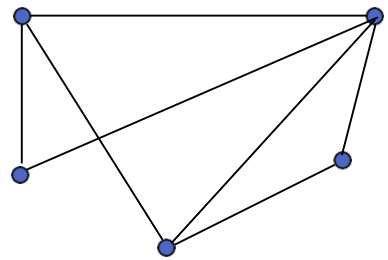
c) (a, d, b, e, a)

d) (a, b, e, c, b, d, a)

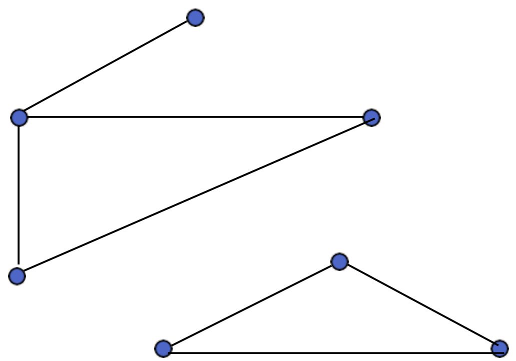


6. Sự liên thông – thành phần liên thông

- **Định nghĩa:** Đồ thị vô hướng G=(V,E) gọi là liên thông nếu luôn tồn tại đường đi giữa 2 đỉnh u, v bất kỳ trong V.



G_1: Liên thông



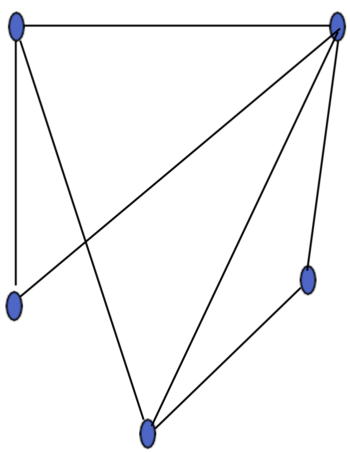
G_2: Không liên thông

6. Sự liên thông–thành phần liên thông

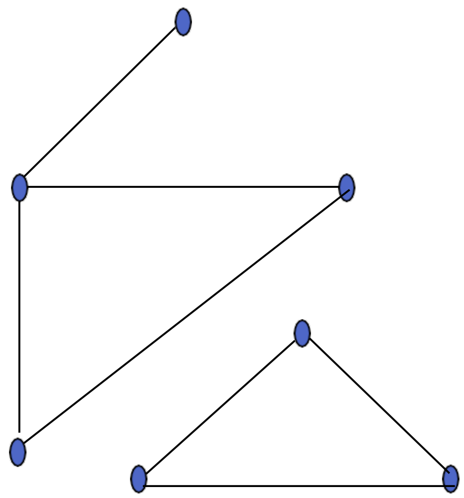
❖ ***Định nghĩa*:** Cho đồ thị vô hướng G=(V,E). Trong trường hợp đồ thị G là không liên thông, nó sẽ rã ra thành một số đồ thị con liên thông không có đỉnh chung. Những đồ thị con liên thông như vậy ta sẽ gọi là các thành phần liên thông của đồ thị.

6. Sự liên thông–thành phần liên thông

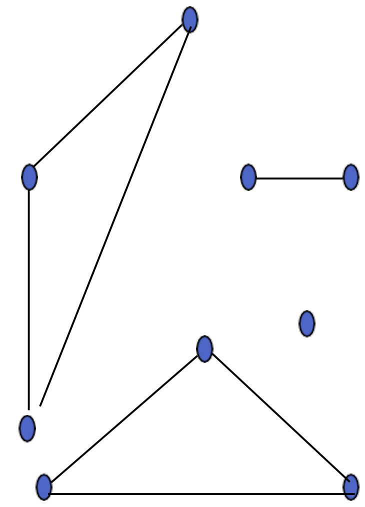
❖ **Ví dụ:**



G_1: có 1 thành phần liên thông



G_2: có 2 thành phần liên thông



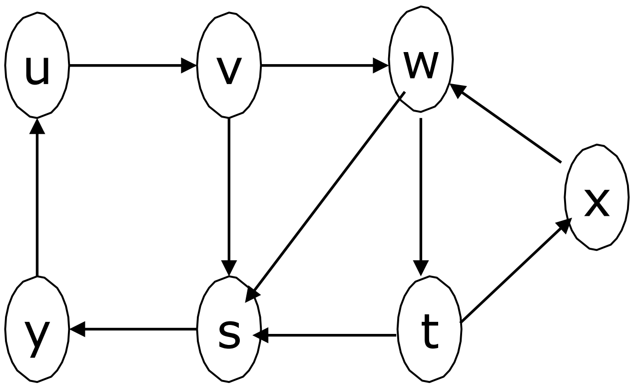
G_3: có 4 thành phần liên thông

6. Sự liên thông–thành phần liên thông

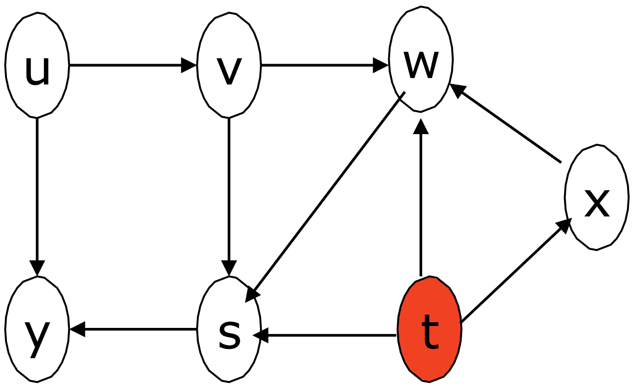
- **Đồ thị có hướng** G gọi là **<u>liên thông yếu</u>** *(Weakly connected)* nếu đồ thị vô hướng tương ứng của nó là liên thông
- **Đồ thị có hướng** G gọi là **<u>liên thông mạnh</u>** (*strongly connected*) nếu với mọi cặp đỉnh khác nhau u,v luôn có đường đi từ đỉnh x đến đỉnh y và ngược lại.

6. Sự liên thông–thành phần liên thông

❖ **Ví dụ:**



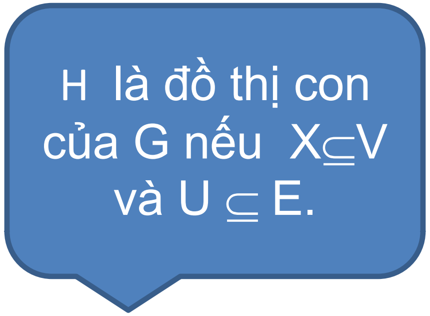
G:liên thông mạnh

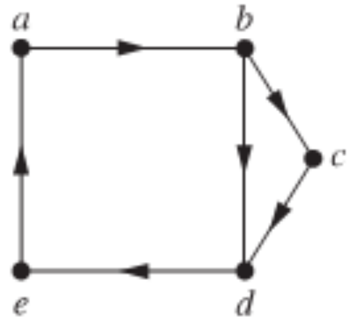


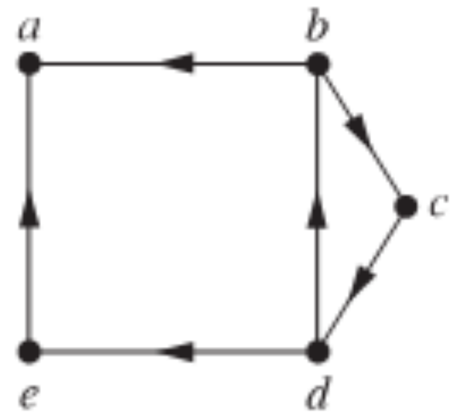
G’ là liên thông yếu (không lt mạnh)

6. Sự liên thông–thành phần liên thông

- Một thành phần liên thông mạnh của đồ thị có hướng G là một đồ thị con liên thông mạnh của G và không là đồ thị con của bất kỳ đồ thị con liên thông mạnh nào khác của G.
- Ví dụ: Tìm các thành phần liên thông mạnh của các đồ thị có hướng sau:





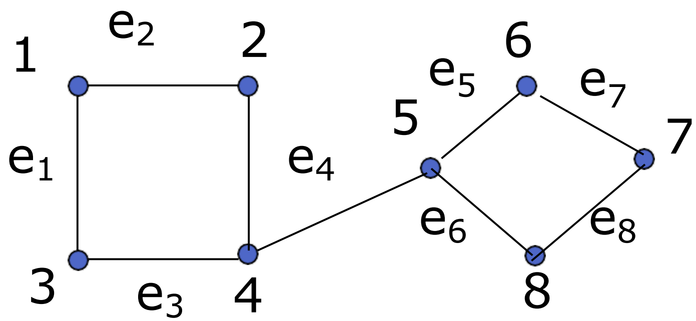


6. Sự liên thông–thành phần liên thông

❖ *<u>Định nghĩa</u>*: Cho G liên thông

- Cạnh e của G gọi là cầu nếu sau khi loại bỏ e, G không còn liên thông
- Đỉnh v trong G gọi là đỉnh nối (đỉnh cắt/vertex cut) nếu sau khi loại bỏ v cùng với các cạnh liên thuộc với nó thì G không còn liên thông.

Ví dụ:



➢ Các đỉnh 4,5 là đỉnh nối

➢ Cạnh e_4 là cầu

6. Sự liên thông–thành phần liên thông

- ***Mệnh đề 1***: Giữa mọi cặp đỉnh phân biệt của một đồ thị vô hướng liên thông luôn có đường đi sơ cấp (Đường đi không có lặp lại đỉnh)
- **Mệnh đề 2:** Mọi đơn đồ thị vô hướng n đỉnh ($n \geq 2$) có **tổng bậc của hai đỉnh tuỳ ý không nhỏ hơn n đều là đồ thị liên thông**.

Hệ quả: Đơn đồ thị mà bậc của mỗi đỉnh của nó không nhỏ hơn một nửa số đỉnh là đồ thị liên thông.

6. Sự liên thông–thành phần liên thông

- **Mệnh đề 3:** Nếu một đồ thị có đúng hai đỉnh bậc lẻ thì hai đỉnh này phải liên thông, tức là có một đường đi nối chúng.
- **Mệnh đề 4:** Cho G=(V,E) là một đồ thị liên thông. Khi đó một đỉnh của G là điểm khớp khi và chỉ khi trong G tồn tại hai đỉnh u và v sao cho mỗi đường đi nối u và v đều phải đi qua đỉnh này.

6. Thuật giải tìm các thành phần liên thông của đồ thị

- **Bước 1:** Tại thời điểm khởi tạo tất cả các đỉnh **i** chưa viếng thăm của đồ thị đều có **Nhan[i]=0 và SoThanhPhanLT=0**
- **Bước 2:** Chọn 1 đỉnh **i** bất kỳ chưa được viếng thăm (Nhan[i]=0)
  - Tăng giá trị của **SoThanhPhanLT** lên 1(SoThanhPhanLT++)
  - Sử dụng hàm **DiTimCacDinhLienThong** để duyệt đỉnh **i** và tất cả các đỉnh **j** chưa được viếng thăm (**Nhan[j]=0**) có nối với đỉnh **i**. Kết thúc mỗi lần duyêt, ta được 1 thành phần liên thông. Để đánh dấu đỉnh này đã viếng thăm ta gán **Nhan[j]= SoThanhPhanLT**.
- **Bước 3:** Lặp lại bước 2 cho đến khi không chọn được đỉnh **i**

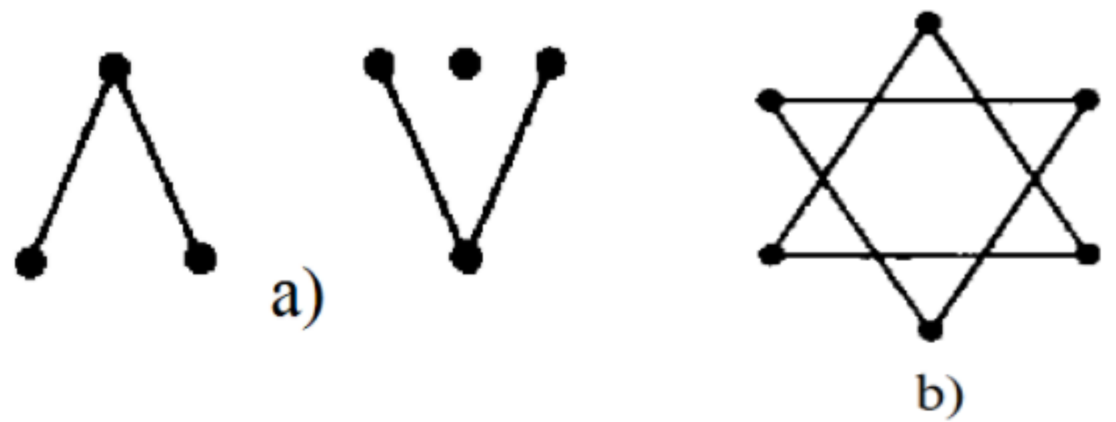
6. Sự liên thông–thành phần liên thông

❖ **Bài 1**: Cho đồ thị vô hướng, đơn G có 7 đỉnh trong đó có một đỉnh bậc 6. Hỏi G có liên thông không?

Giải. Đỉnh bậc 6 nối với 6 đỉnh còn lại. Do đó hai đỉnh bất kỳ đều có một đường đi qua đỉnh bậc 6 → **Đồ thị G thiên thông**

6. Sự liên thông–thành phần liên thông

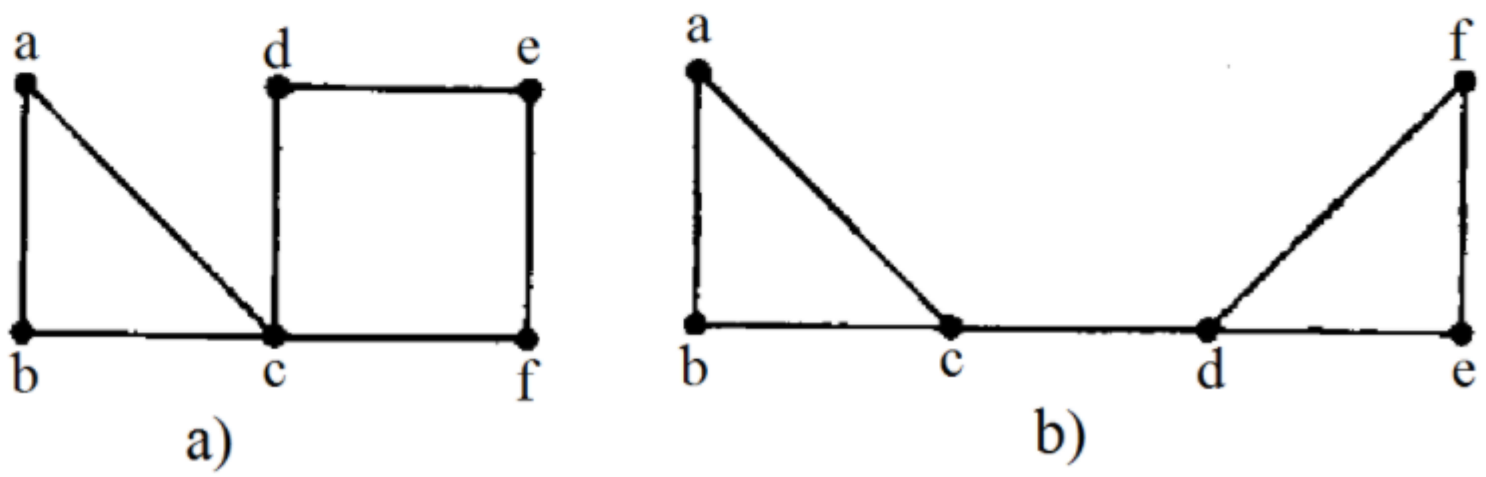
❖ **Bài 2:** Xác định xem các đồ thị đã cho có liên thông không.



Có bao nhiêu thành phần liên thông trong các đồ thị ở trên, tìm các thành phần liên thông đó.

6. Sự liên thông–thành phần liên thông

❖ **Bài 3:** Tìm tất cả các đỉnh cắt và cạnh cắt của đồ thị.



7. Hướng dẫn cài đặt đồ thị

- Đọc các loại đồ thị từ file txt
- Xuất ra ma trận kề/ma trện liên thuộc
- Cài đặt các phương thức cần thiết để xét tính liên thông/ tìm số thành phần liên thông của đồ thị

8. CÁC THUẬT TOÁN TÌM KIẾM TRÊN ĐỒ THỊ

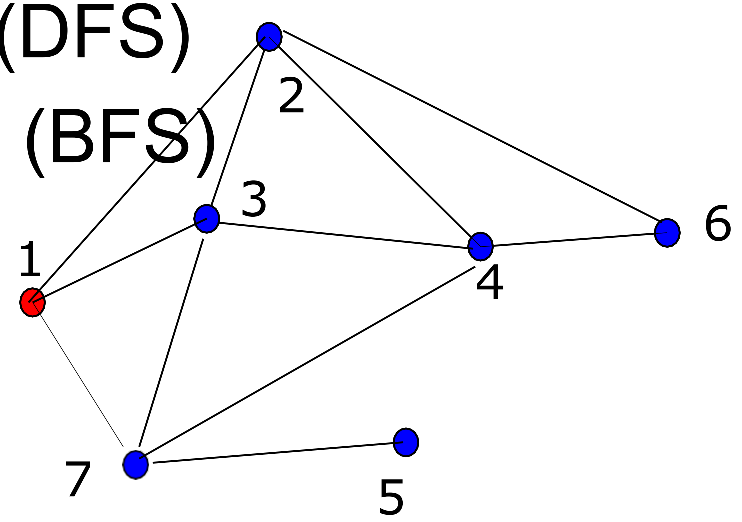
1. **Duyệt đồ thị theo chiều sâu (DFS: Depth First Search)**
2. **Duyệt đồ thị theo chiều rộng (BFS: Breadth First Search)**
3. **Tìm đường đi**
4. **Kiểm tra tính liên thông**

8. Duyệt đồ thị

- là thăm qua tất cả các đỉnh của đồ thị
- Thường dùng một trong 2 cách để duyệt một đồ thị liên thông:
  - Duyệt theo chiều sâu (DFS)
  - Duyệt theo chiều rộng (BFS)

Ví dụ: Duyệt đồ thi sau

bắt đầu từ đỉnh 1



8.1 Duyệt đồ thị theo chiều sâu (DFS: Depth First Search)

Ý tưởng

- ❖ Bắt đầu tìm kiếm từ một đỉnh v_0 nào đó của đồ thị, chọn v là một đỉnh tuỳ ý kề với v_0, ta sẽ thăm đỉnh này và lặp lại quá trình tìm kiếm đối với v.
- ❖ Ở bước tổng quát, giả sử ta đang xét đỉnh v.
  - Nếu như trong số các đỉnh kề với v tìm được đỉnh w là chưa được xét thì ta sẽ xét đỉnh này (trở thành đã xét) và bắt đầu từ đỉnh đó ta sẽ bắt đầu quá trình tìm kiếm,
  - Còn nếu như không còn đỉnh nào kề với v là chưa xét thì ta nói rằng đỉnh v đã duyệt xong và quay trở lại tiếp tục tìm kiếm từ đỉnh mà trước đó ta đến được đỉnh v.

8.1 Thuật toán tìm kiếm theo chiều sâu

Thuật toán dùng 2 danh sách:

STACK: Ngăn xếp kết nạp các đỉnh mở cần được thăm(LIFO).

CLOSED: danh sách các đỉnh đã được xét

```
DFS(v) {
    CLOSED = ∅;  STACK = ∅;
    STACK ← v;    // kết nạp v vào đầu ngăn xếp
    While ( STACK <> ∅ ) {
        p ← STACK;    // lấy p từ đầu ngăn xếp
        if ( p ∉ CLOSED ){// nếu đỉnh p chưa được xét
            Thamdinh(p);
            CLOSED ← p;    // đỉnh p đã được xét
            for ( u ∈ Ke(p) )
                if ( u ∉ CLOSED ) STACK ← u;
        }
    }
}
```

8.1 Thuật toán tìm kiếm theo chiều sâu

Cài đặt bằng đệ quy

Tim kiếm theo chiều sâu bắt đầu từ đỉnh v

B1: Lấy s là một đỉnh của đồ thị

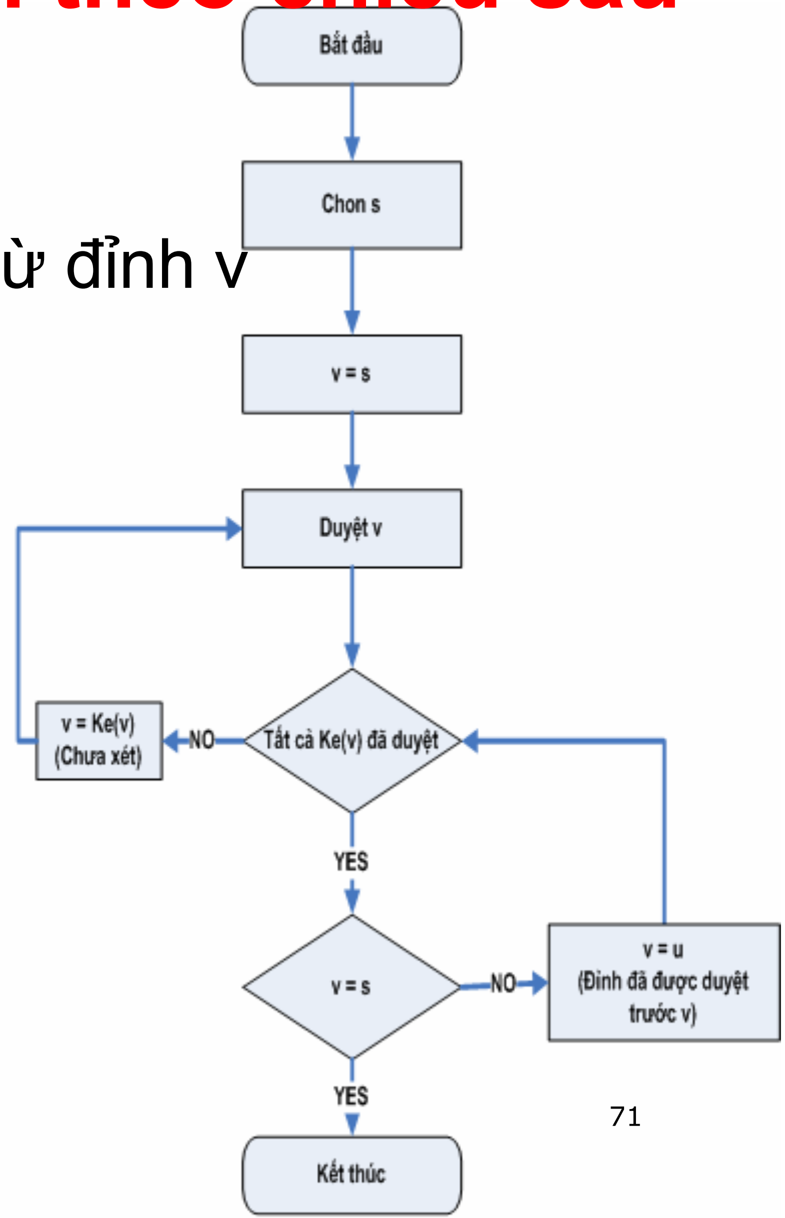
B2: Đặt v = s

B3: Duyệt đỉnh v

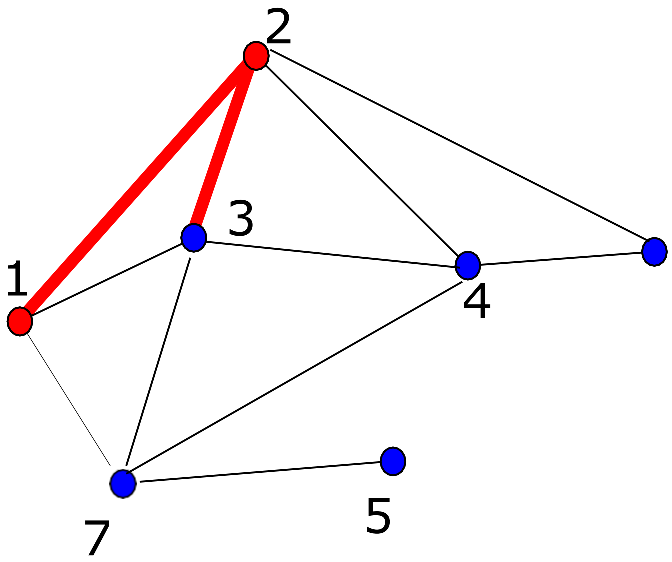
B4: Nếu ∀ đỉnh kề của v đều được duyệt, đặt v=đỉnh đã được duyệt trước đỉnh v, Nếu v = s thì đi đến Bước 6, ngược lại trở lại Bước 3.

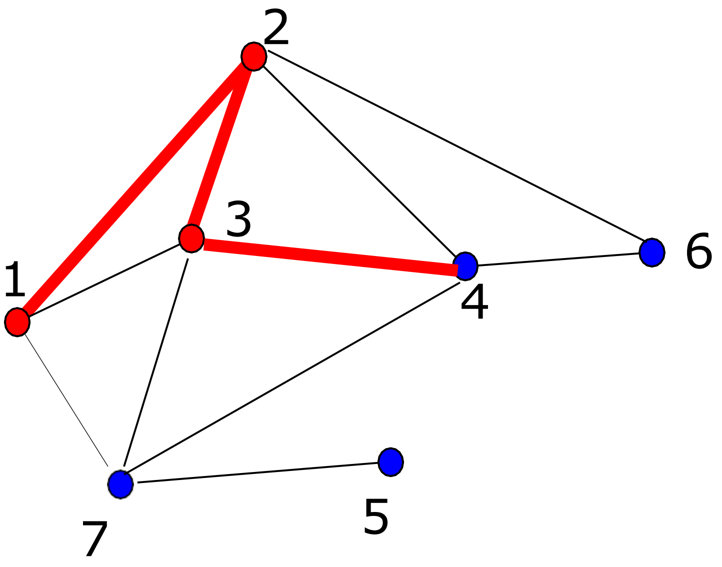
B5: Chọn u là đỉnh kề chưa được duyệt của v, đặt v = u, trở lại Bước 3

B6: Kết thúc

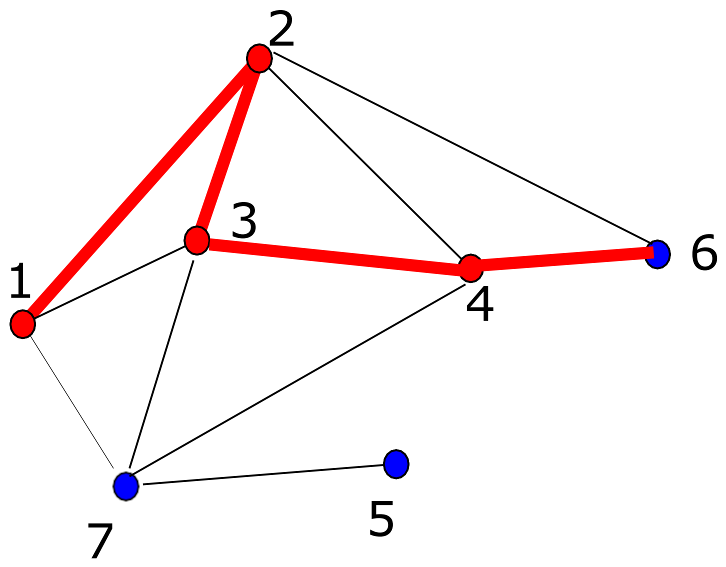


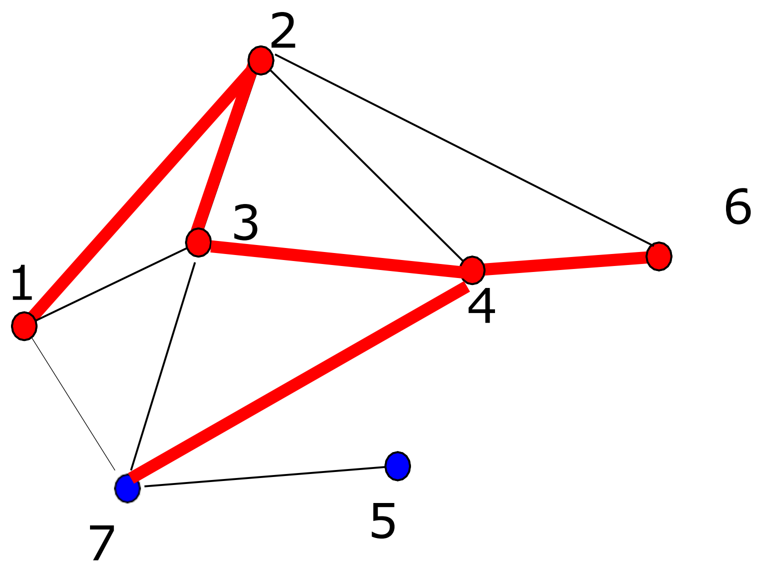
8.1 Thuật toán tìm kiếm theo chiều sâu



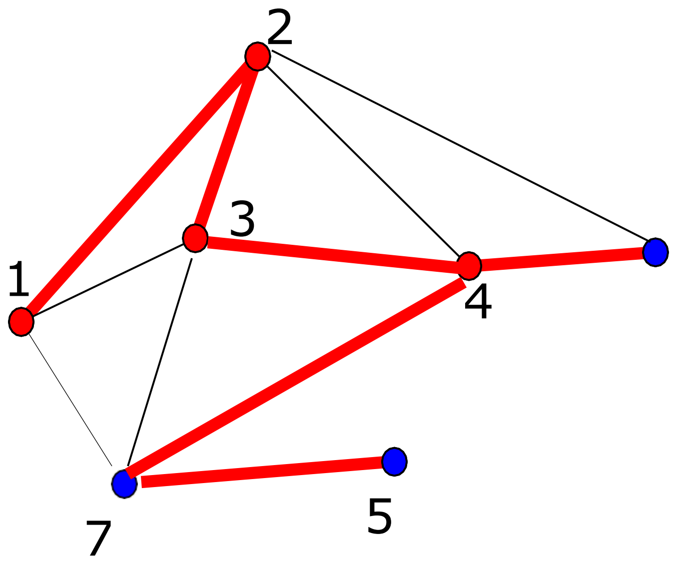


8.1 Thuật toán tìm kiếm theo chiều sâu





8.1 Thuật toán tìm kiếm theo chiều sâu



8.1 Thuật toán tìm kiếm theo chiều sâu

DFS không đệ quy

B1: Lấy s là một đỉnh của đồ thị

B2: Đặt s vào STACK

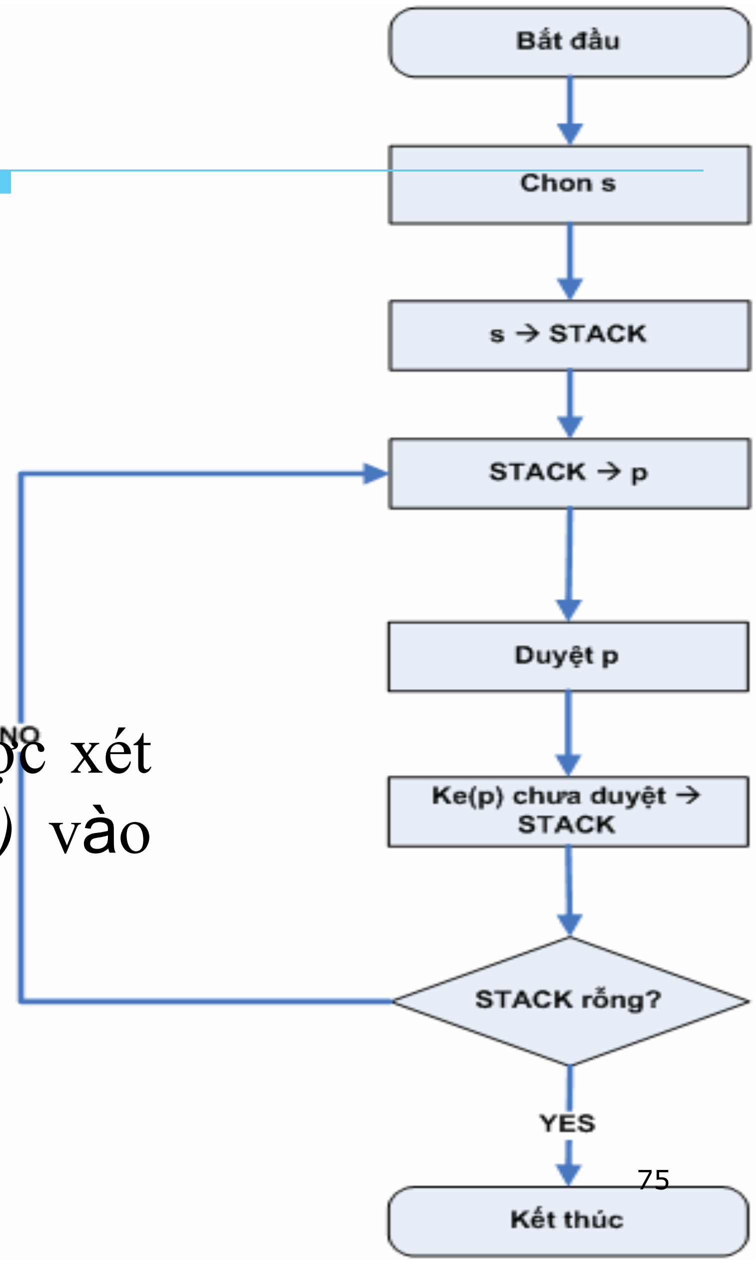
B3: Nếu STACK rỗng đi đến 7.

B4: Lấy đỉnh p từ STACK

B5: Duyệt đỉnh p

B6: Đặt các đỉnh kề của p chưa được xét *(chưa từng có mặt trong STACK)* vào STACK, trở lại 3.

B7: Kết thúc.



8.1 Thuật toán tìm kiếm theo chiều sâu

- Input: G vô hướng, liên thông
- B_1: Khởi tạo:

  stack$\leftarrow\varnothing$;

  visited[i]=0 $\forall$i, visited[1]=true

  v=1; stack.push(v);
- B_2: v=stack.peek();

  Tìm một đỉnh kề với v mà chưa được thăm

  (Tìm j: visited[j]=false và a[v][j]>0)

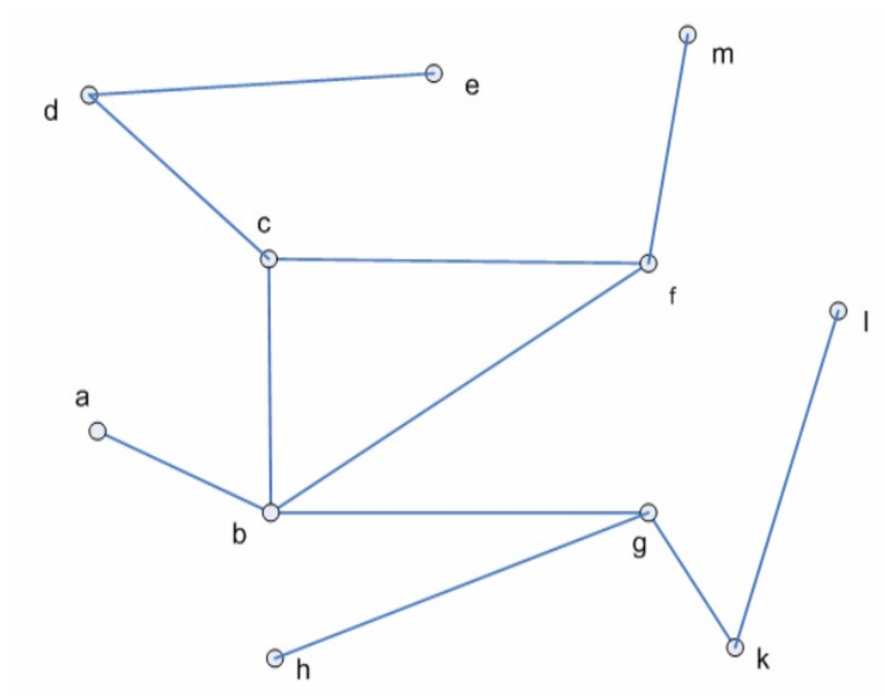
  Nếu tìm thấy:

  - stack.push(v);
  - v$\leftarrow$j
- Ngược lại (nếu không tìm thấy):

8.1 Thuật toán tìm kiếm theo chiều sâu

- Bài tập: ghi ra thứ tự các đỉnh duyệt theo chiều s

Thứ tự duyệt:
d c b a
f m
g h
k l
e



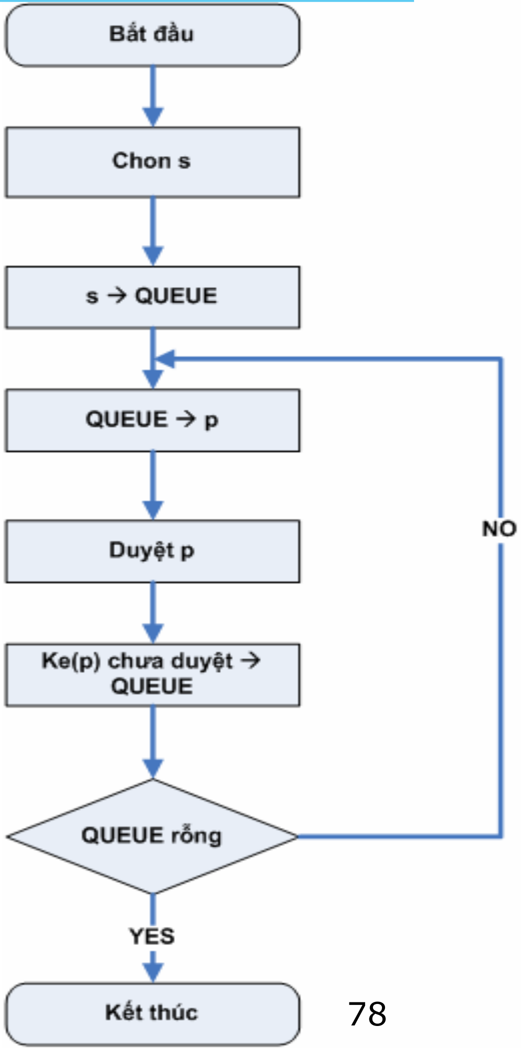
8.2 Duyệt đồ thị theo chiều rộng (BFS: Breadth First Search)

❖ **Nguyên lý**

- Tìm kiếm theo chiều rộng trên đồ thị, được xây dựng trên cơ sở thay thế ngăn xếp STACK bởi một cấu trúc hàng đợi(QUEUE).
- Với sự thay đổi này, đỉnh được thăm càng sớm sẽ càng sớm duyệt xong.

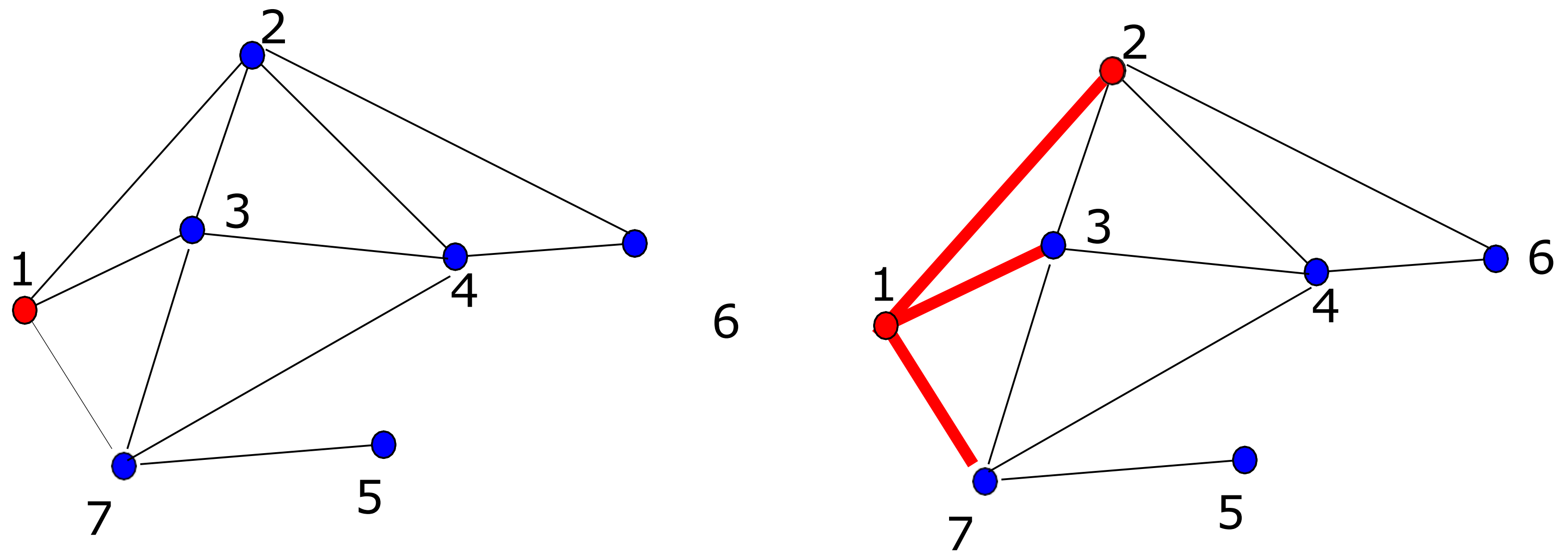
8.2 Thuật toán tìm kiếm theo chiều rộng

```
Procedure BFS(u)
Begin     Queue:= ∅;
          Queue.push(u);
          visited[u]:=True;
          While Queue<>∅ do
                Begin  v=Queue.pop();
                       visit(v);
                for each vertex w adjacent to v do
                  If not visited[w] then
                       Begin
                       Queue.push(w); visited[w]=true;
                       End;
                End;
End;
```



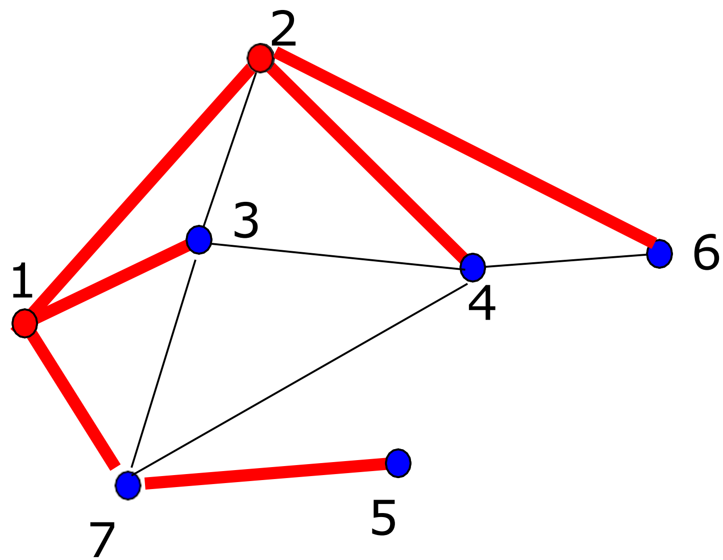
8.2 Thuật toán tìm kiếm theo chiều rộng

❖ Duyệt theo chiều rộng



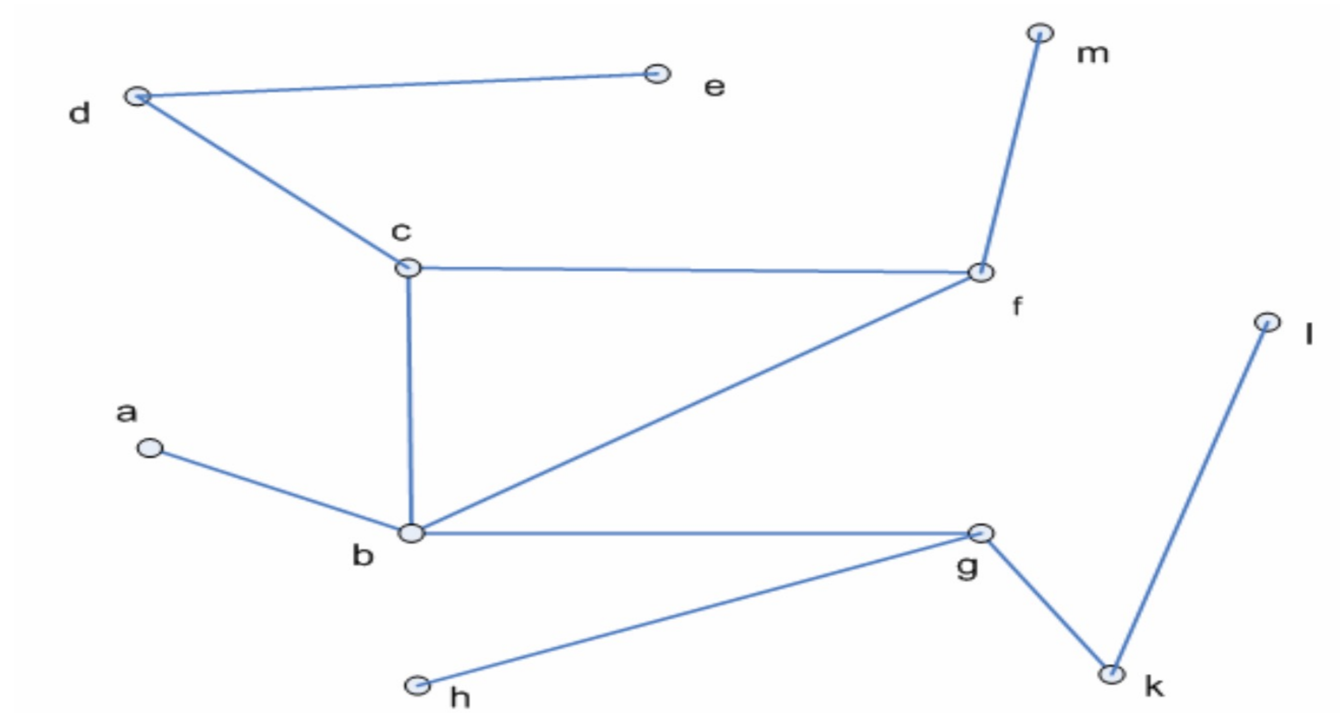
8.2 Thuật toán tìm kiếm theo chiều rộng

❖ Duyệt theo chiều rộng



8.2 Thuật toán tìm kiếm theo chiều rộng

❖ Bài tập: ghi ra thứ tự các đỉnh duyệt theo chiều rộng (BFS) cho đồ thị sau



8.3 Tìm đường đi

❖ Bài toán

- Cho đồ thị G, s và t là hai đỉnh tùy ý của đồ thị. Hãy tìm đường đi từ s đến t.

❖ Phương pháp

- Bắt đầu từ đỉnh s, Sử dụng DFS hoặc BFS để duyệt
  đồ thị.
  - Tìm thấy đường đi từ s đến t: ChuaXet(t) = 0 ⇔ đã được xét
  - Không tìm thấy đường đi từ s đến t: ChuaXet(t) = 1 ⇔chưa dc xét
- Sử dụng thêm mảng Truoc[] để lưu vết

8.3 Tìm đường đi- Tìm đường đi theo chiều sâu

```
/* Khai báo các biến ChuaXet, Ke */
DFS(v);
{
   Duyệt đỉnh (v);
   ChuaXet[v] = 0;
     for ( u ∈ Ke(v) )
        if ( ChuaXet[u] )
             {
                    Truoc[u] = v; /* Lưu vết*/
                     DFS(u);
              }
}
```

8.3 Tìm đường đi- Tìm đường đi theo chiều rộng

```
/* Khai báo các biến ChuaXet, Ke , QUEUE */
BFS(v){
   QUEUE = ∅; QUEUE ⇐ v; ChuaXet[v] = 0;
   while ( QUEUE ≠ ∅ ){
        p ⇐ QUEUE;
        Duyệt đỉnh p;
        for ( u ∈ Ke(p) )
              if ( ChuaXet[u] ){
                        QUEUE ⇐ u;
                        ChuaXet[u] = 0;
                        Truoc[u] = p;/*Lưu vết*/
               }
   }
}
```

8.3 Tìm đường đi- Tìm đường đi theo chiều rộng

Khôi phục đường đi từ s đến t

s → x1 → x2 → … → xn → t

Cài đặt:

```
v = t;
while (v != s) {
printf (v);
v = Truoc[v];
}
```

8.4. Kiểm tra tính liên thông

- Bài toán
  - Tính số thành phần liên thông của đồ thị, và xác định những đỉnh thuộc cùng một thành phần liên thông.
- Phương pháp
  - Sử dụng DFS và BFS
  - Biến inconnect đếm số thành phần liên thông của đồ thị.
  - Mảng index[] lưu chỉ số của các thành phần liên thông.

8.4. Kiểm tra tính liên thông- DFS

```
/* Khai báo các biến ChuaXet, Ke, index*/
DFS(v);{
   Duyệt đỉnh (v);
    index[v] = inconnect;  ChuaXet[v] = 0;
   for ( u ∈ Ke(v) )
     if ( ChuaXet[u] )  DFS(u);
}
     main() {
     /* Nhập đồ thị, tạo biến Ke */
     for ( v ∈ V ) ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
     inconnect = 0;
     for ( v ∈ V )
     if ( ChuaXet[v] ) {
     inconnect ++; DFS(v);
     }
     }
```

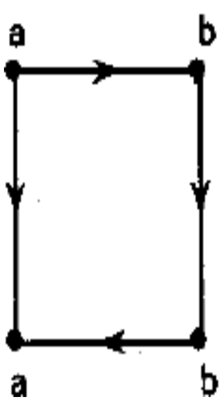
8.4. Kiểm tra tính liên thông-BFS

```
/* Khai báo các biến toàn cục ChuaXet, Ke,
QUEUE, index */
BFS(v) {
QUEUE = 0; QUEUE ⇐ v; ChuaXet[v] = 0;
        while ( QUEUE ≠ 0 ) {
        p ⇐ QUEUE; Duyệt đỉnh p;
        index[p] = inconnect;
        for ( u ∈ Ke(p) )
        if ( ChuaXet[u] ) {
                QUEUE ⇐ u;ChuaXet[u] = 0;
        }
    }
}
```

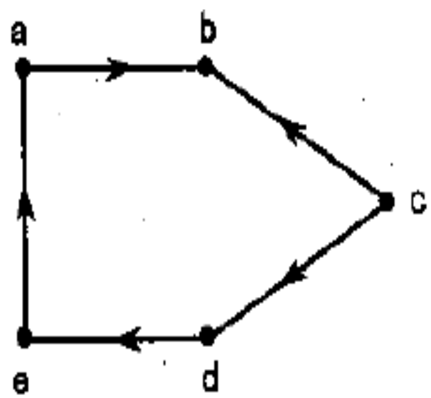
```
main() {
for ( v ∈ V ) ChuaXet[v] = 1;
inconnect = 0;
for ( v ∈ V )
if ( ChuaXet[v] ) {
inconnect + + ; BFS(v);
}
}
```

Bài tập chương 01

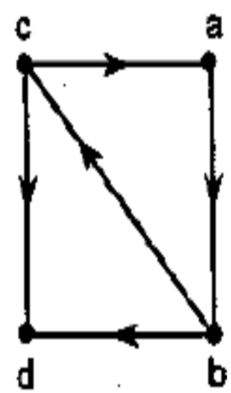
1) Viết ma trận kề và mà trận liên thuộc của các đồ thị sau:



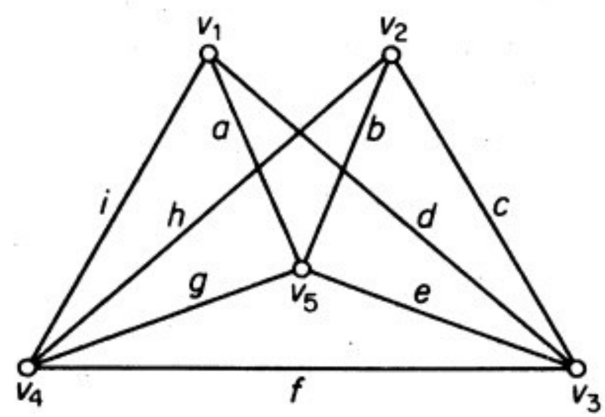
H1



H2



H3



H4

Bài tập chương 01

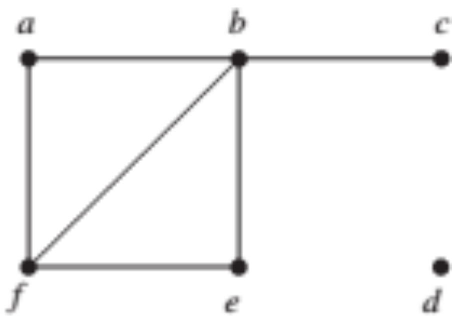
2) Tìm số đỉnh của đồ thị vô hướng G. Biết:
   a) G có 12 cạnh và mọi đỉnh đều có bậc là 2
   b) G có 15 cạnh, 3 đỉnh bậc 4 và các đỉnh còn lại bậc 3.
   c) G có 6 cạnh và mọi đỉnh đều có bậc bằng nhau.

3) Một đồ thị vô hướng G có 19 cạnh và mọi đỉnh đều có bậc >=3. G có tối đa bao nhiêu đỉnh?

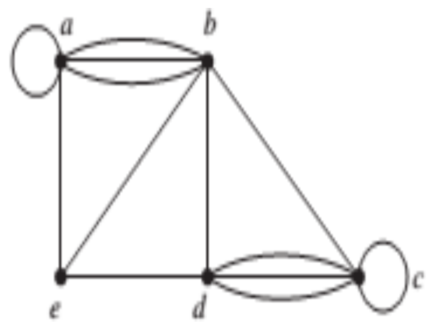
Bài tập chương 01

4) Với các đồ thị vô hướng sau đây, tính bậc của từng đỉnh, chỉ ra các đỉnh treo, các đỉnh cô lập, sau đó tính tổng bậc của tất cả các đỉnh, áp dụng định lý bắt tay tính số cạnh của từng đồ thị:

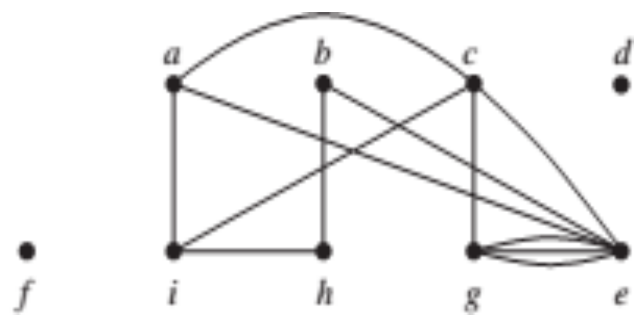
1.



2.

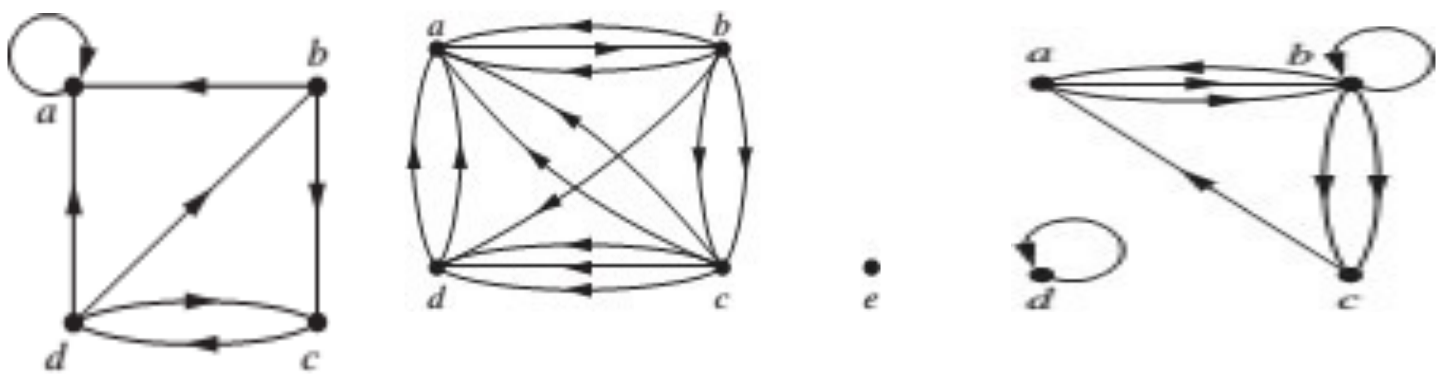


3.



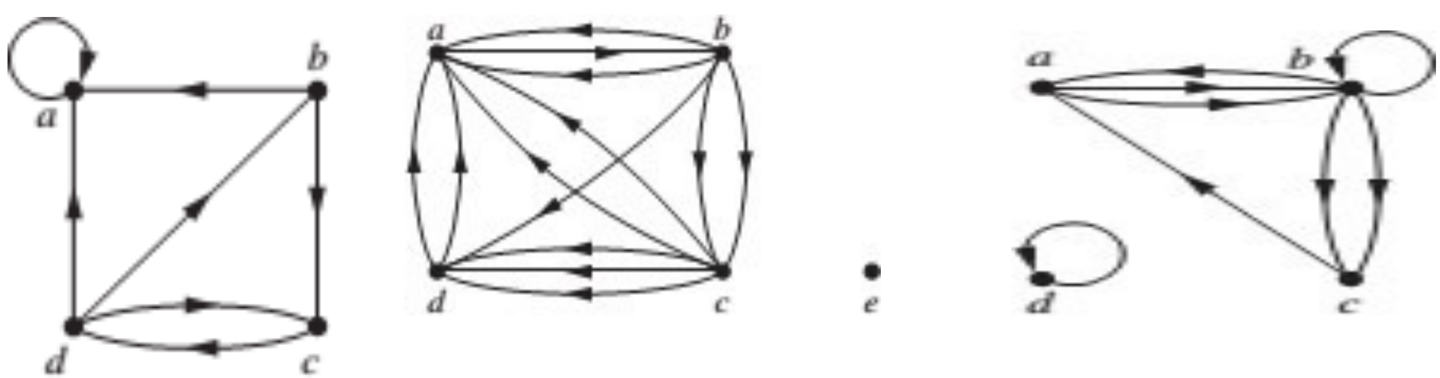
Bài tập chương 01

5) Với các đồ thị có hướng sau đây, tính nữa bậc trong, nữa bậc ngoài của từng đỉnh, tính số cung của từng đồ thị:



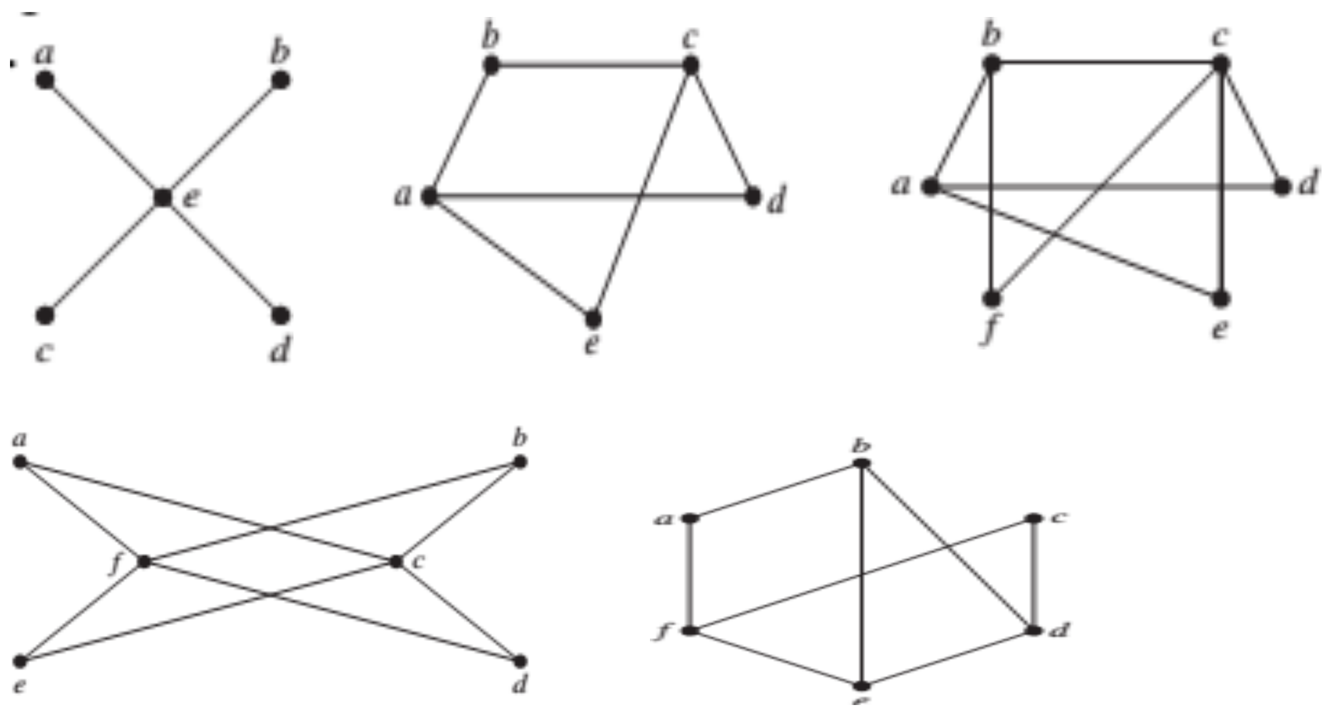
Bài tập chương 01

6) Với các đồ thị có hướng sau đây, tính nữa bậc trong, nữa bậc ngoài của từng đỉnh, tính số cung của từng đồ thị:



Bài tập chương 01

7) Các đồ thi sau đây, đồ thị nào là lưỡng phân



Thực hành/Tự học chương 1

- Cài đặt đồ thị (vô hướng/ có hướng):
  - Sử dụng ma trận kề, ma trận liện thuộc để biểu diễn đồ thị
  - Các phương thức:
    - Thêm một đỉnh
    - Thêm một cạnh
    - In ma trận kề/ma trận liện thuộc
    - Duyệt đồ thị (theo DFS và BFS)
    - Tính bậc của đỉnh
    - Tìm một đường đi từ đỉnh x đến đỉnh y (áp dụng DFS, BFS)
    - Kiểm tra tính liên thông của đồ thị
    - Tìm các thành phần liên thông
    - Kiểm tra đồ thị có phải là đồ thị con của một đồ thị khác