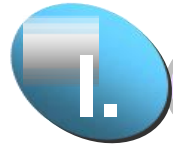


Chương 2: Biểu diễn đồ thị

Nội dung



Các cách biểu diễn đồ thị

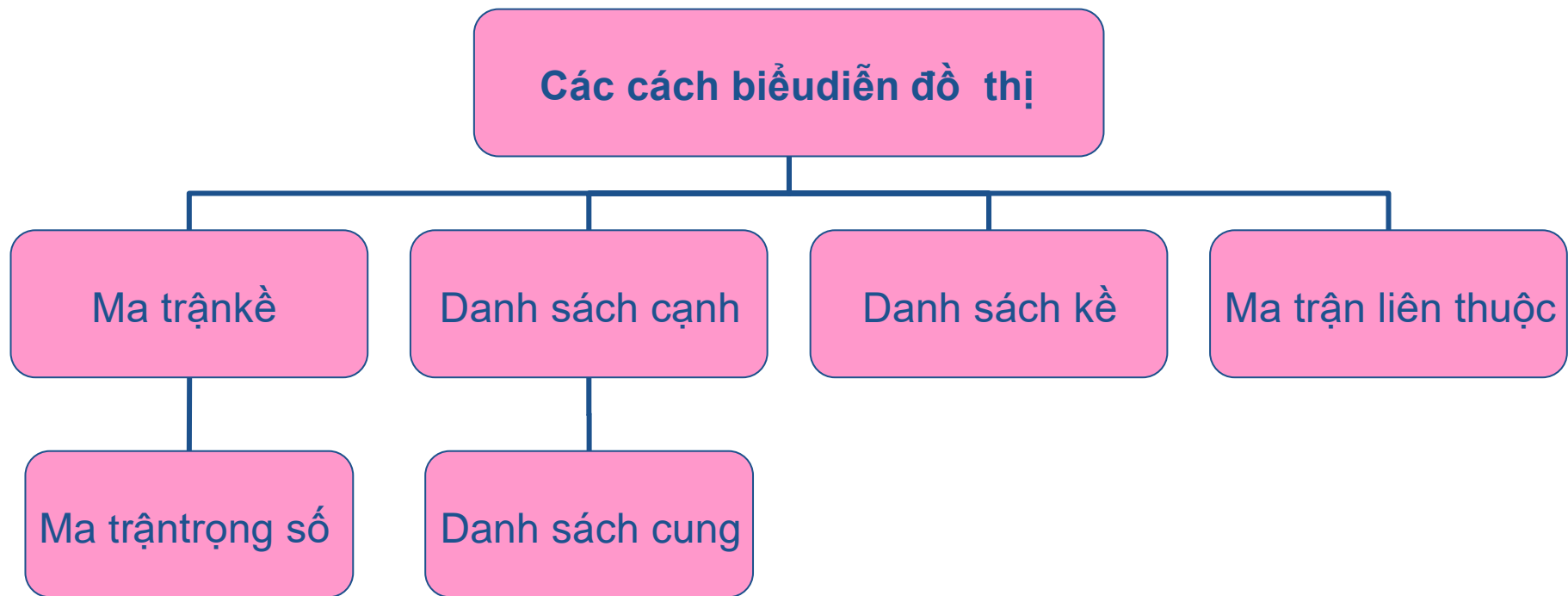


Sự đẳng cấu của các đồ thị



Hướng dẫn cài đặt

I. Các cách biểu diễn đồ thị

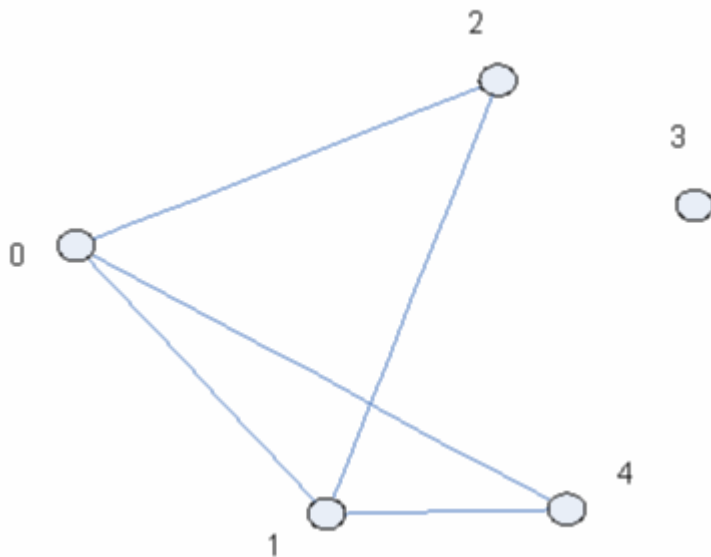


I.1. Ma trận kề (đơn đồ thị vô hướng)

Định nghĩa

- ③ Đơn đồ thị $G = (V, E)$ với tập đỉnh $V = \{0, \dots, n-1\}$, tập cạnh $E = \{e_0, e_1, \dots, e_{m-1}\}$. Ta gọi ma trận kề của G là
- ③ $A = \{a_{i,j}, i, j = 0, \dots, n-1\}$, với:

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



	0	1	2	3	4
0	0	1	1	0	1
1	1	0	1	0	1
2	1	1	0	0	0
3	0	0	0	0	0
4	1	1	0	0	0

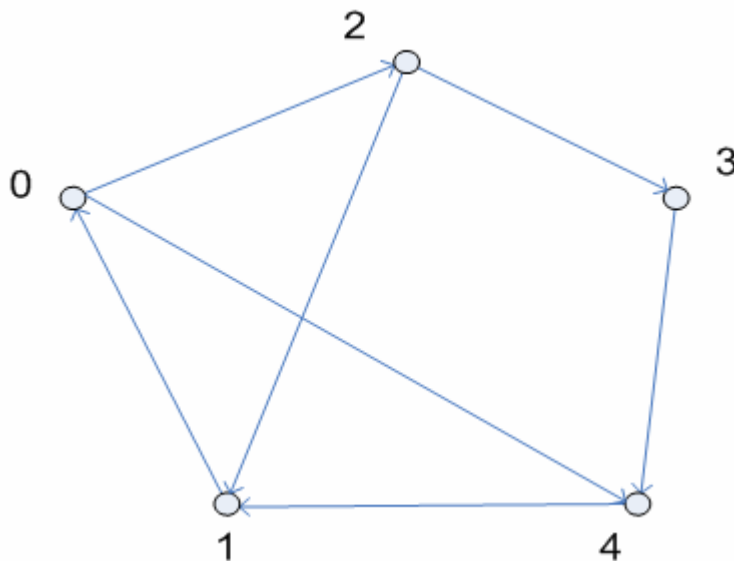
I.1. Ma trận kề (đơn đồ thị có hướng)

Định nghĩa

③ Giống đơn đồ thị có hướng

③ E là tập các **cung**

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

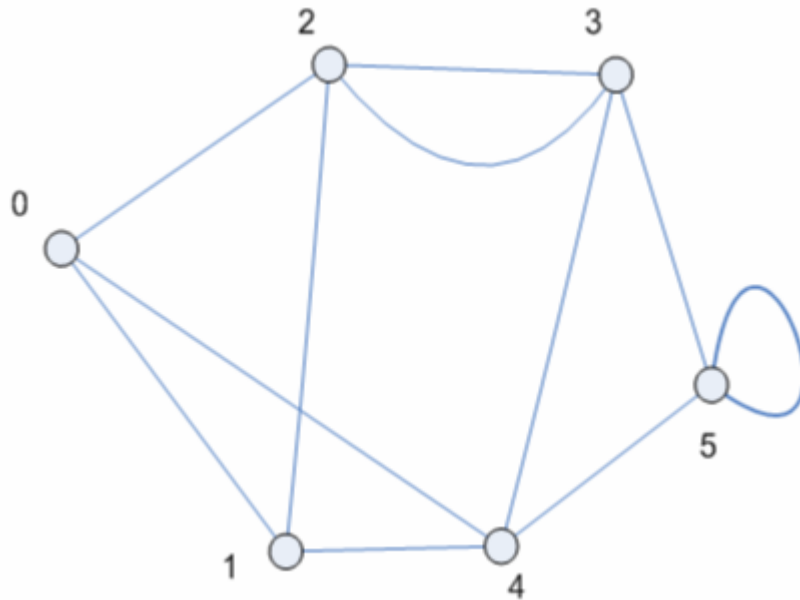


	0	1	2	3	4
0	0	0	1	0	1
1	1	0	0	0	0
2	0	1	0	1	0
3	0	0	0	0	1
4	0	1	0	0	0

I.1. Ma trận kề (Đồ thị)

Định nghĩa

- ③ E là tập các **cạnh/cung**
- ③ $A_{i,j}$ là **số cạnh** nối đỉnh i và đỉnh j



	0	1	2	3	4	5
0	0	1	1	0	1	0
1	1	0	1	0	1	0
2	1	1	0	2	0	0
3	0	0	2	0	1	1
4	1	1	0	1	0	1
5	0	0	0	1	1	1

I.1. Ma trận kề (Đồ thị)

Một số tính chất của ma trận kề

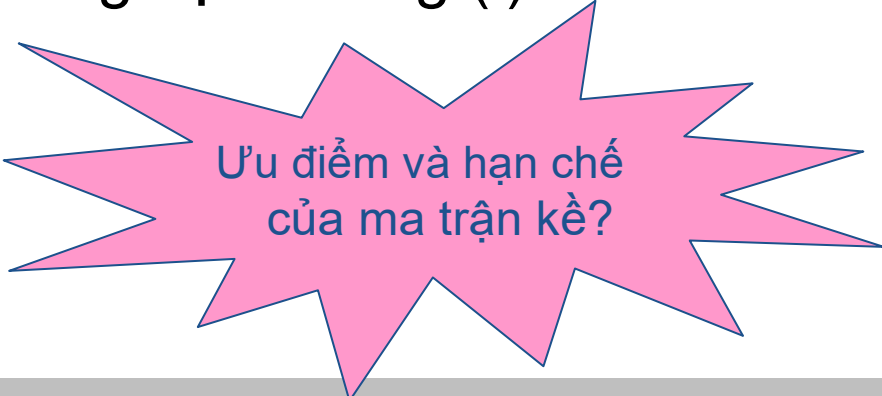
- ③ Ma trận kề của đồ thị vô hướng là đối xứng
 $a[i,j] = a[j,i]$. Ngược lại, ma trận đối xứng $(0,1)$, có đường chéo chính bằng 0, bậc n sẽ tương ứng với đồ thị vô hướng n đỉnh.

- ③ Nếu đồ thị vô hướng:

Tổng dòng thứ i = Tổng cột thứ i = $\deg(i)$

- ③ Nếu đồ thị có hướng:

Tổng dòng i = $\deg^+(i)$, Tổng cột i = $\deg^-(i)$



Ưu điểm và hạn chế
của ma trận kề?

I.2. Ma trận trọng số (đơn đồ thị)

Định nghĩa

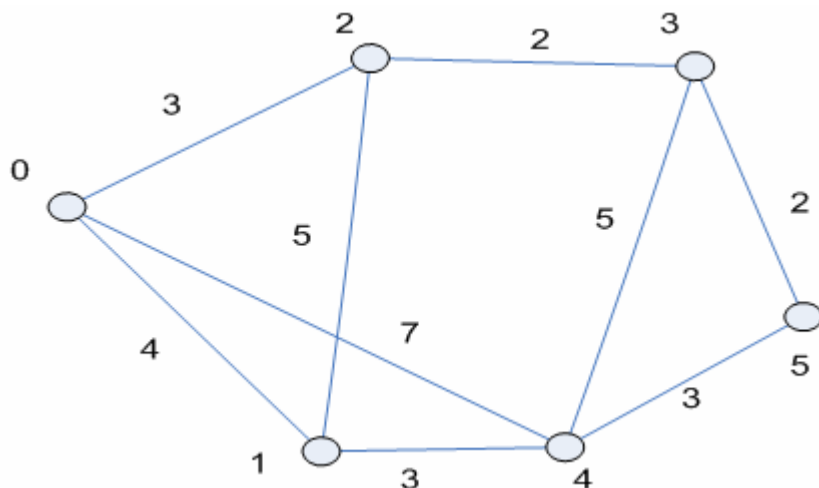
③ Đơn đồ thị $G = (V, E)$ với tập đỉnh $V = \{0, \dots, n-1\}$, tập cạnh $E = \{e_0, e_1, \dots, e_{m-1}\}$.

③ Ta gọi **ma trận kề trọng số** của G là

• $A = \{a_{i,j}, i, j = 0, \dots, n-1\}$, với:

$$a_{i,j} = \begin{cases} b, & \text{if } (i, j) \notin E \\ c_k, & \text{if } (i, j) \in E \end{cases}$$

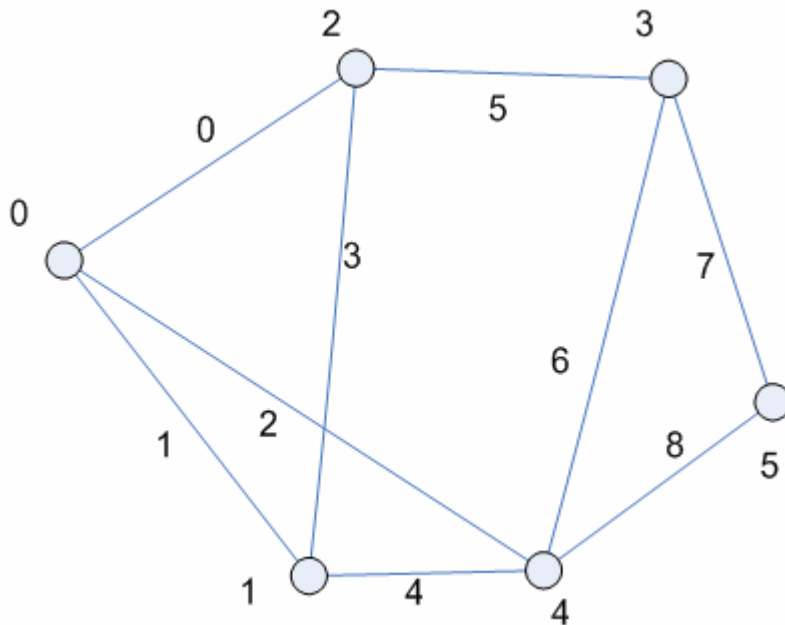
C_k là một giá trị nào đó được quy định trước (0, -1, ∞ , $-\infty$, ..)



	0	1	2	3	4	5
0	0	4	3	0	7	0
1	4	0	5	0	3	0
2	3	5	0	2	0	0
3	0	0	2	0	5	2
4	7	3	0	5	0	3
5	0	0	0	2	3	0

I.3. Danh sách cạnh

- ③ Đối với các đồ thị thưa n đỉnh, m cạnh ($m < 6n$) người ta thường dùng cách biểu diễn danh sách cạnh để tiết kiệm không gian lưu trữ
- ③ Lưu các cạnh $e=(u, v)$ của đồ thị trong một danh sách
- ③ Danh sách có thể được cài đặt bằng **mảng 1 chiều** hoặc **danh sách liên kết**.



Cạnh	Đầu 1	Đầu 2
0	0	2
1	0	1
2	0	4
3	1	2
4	1	4
5	2	3
6	3	4
7	3	5
8	4	5

I.3. Danh sách cạnh

Cài đặt bằng mảng 1 chiều

0	1	2	3	4	5	6	7	8
(0,2)	(0,1)	(0,4)	(1,2)	(1,4)	(2,3)	(3,4)	(3,5)	(4,5)

Cài đặt bằng danh sách liên kết



```
typde struct tagNode
```

```
{
```

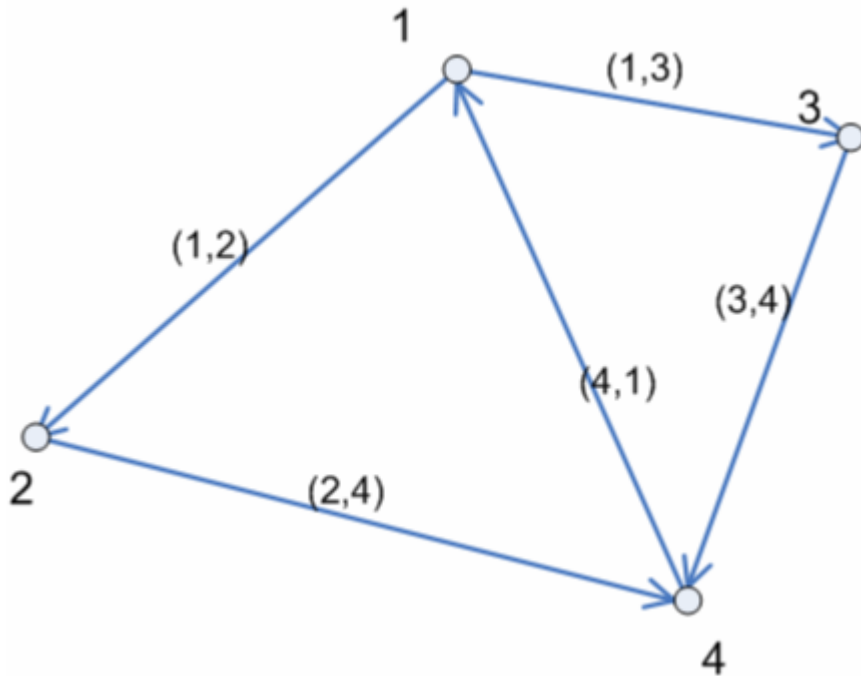
```
    int diem dau1, diem dau2;
```

```
} Canh;
```

Cạnh \ Đầu	1	Đầu 2
0	0	2
1	0	1
2	0	4
3	1	2
4	1	4
5	2	3
6	3	4
7	3	5
8	4	5

I.4. Danh sách cung

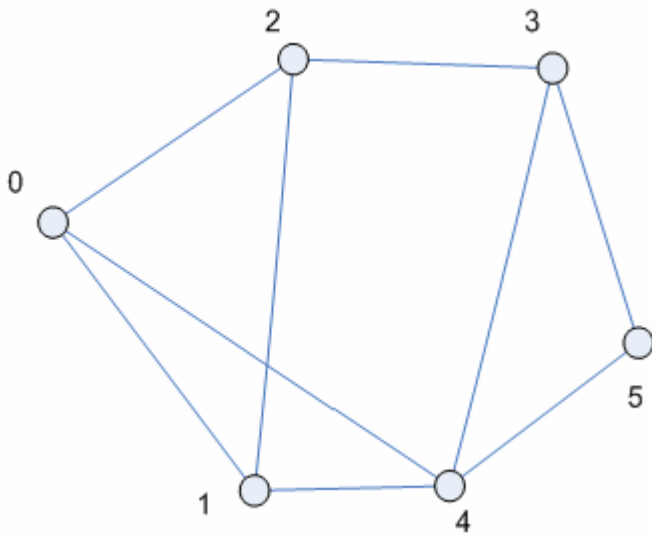
- ③ Trong trường hợp đồ thị có hướng thì mỗi phần tử của danh sách (gọi là **danh sách cung**) là một cung $e=(u, v)$. Trong đó u là **đỉnh đầu**, v là **đỉnh cuối** của cung.



Cạnh	Đầu 1	Đầu 2
(1,2)	1	2
(4,1)	4	1
(1,3)	1	3
(2,4)	2	4
(3,4)	3	4

I.4. Danh sách kề

- ③ Tương ứng với mỗi đỉnh v của đồ thị, ta có tương ứng một danh sách để lưu các đỉnh kề với nó.
- ③ Danh sách: mảng 1 chiều, hoặc danh sách liên kết



Đỉnh V	Các cạnh kề
0	1, 2, 4
1	0, 2, 4
2	0, 1, 3
3	2, 4, 5
4	0, 1, 3, 5
5	3, 4

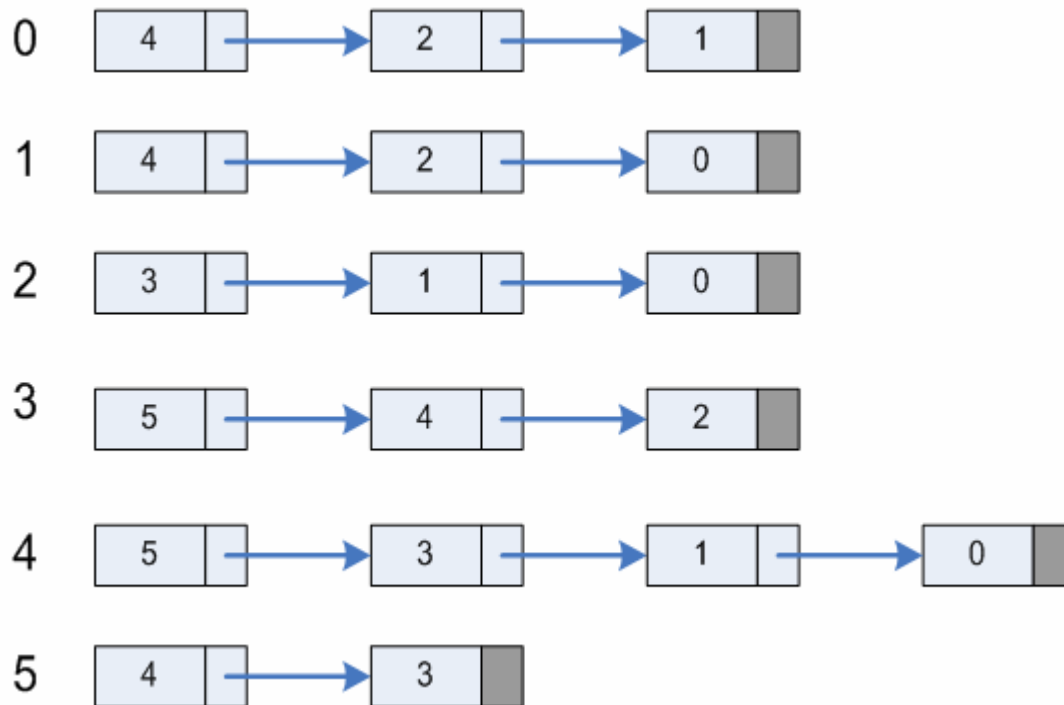
Cài đặt bằng mảng:

$Ke[] = \{1, 2, 4, 0, 2, 4, 0, 1, 3, 2, 4, 5, 0, 1, 3, 5, 3, 4\}$

$ViTri[] = \{0, 3, 6, 9, 12, 16\}$

I.4. Danh sách kề

③ Cài đặt bằng danh sách kề liên kết



Đỉnh V	Các cạnh kề
0	1, 2, 4
1	0, 2, 4
2	0, 1, 3
3	2, 4, 5
4	0, 1, 3, 5
5	3, 4

I.4. Danh sách kê

③ Thuật toán xây dựng danh sách kê liên kết

```
# include <iostream.h>
# include <stdlib.h>
const maxV = 99;
typedef struct Node {
    int v;
    struct Node*next;
}node;
int j, x, y, m, n, v ;
node *p, *ke[maxV];
```

I.4. Danh sách kề

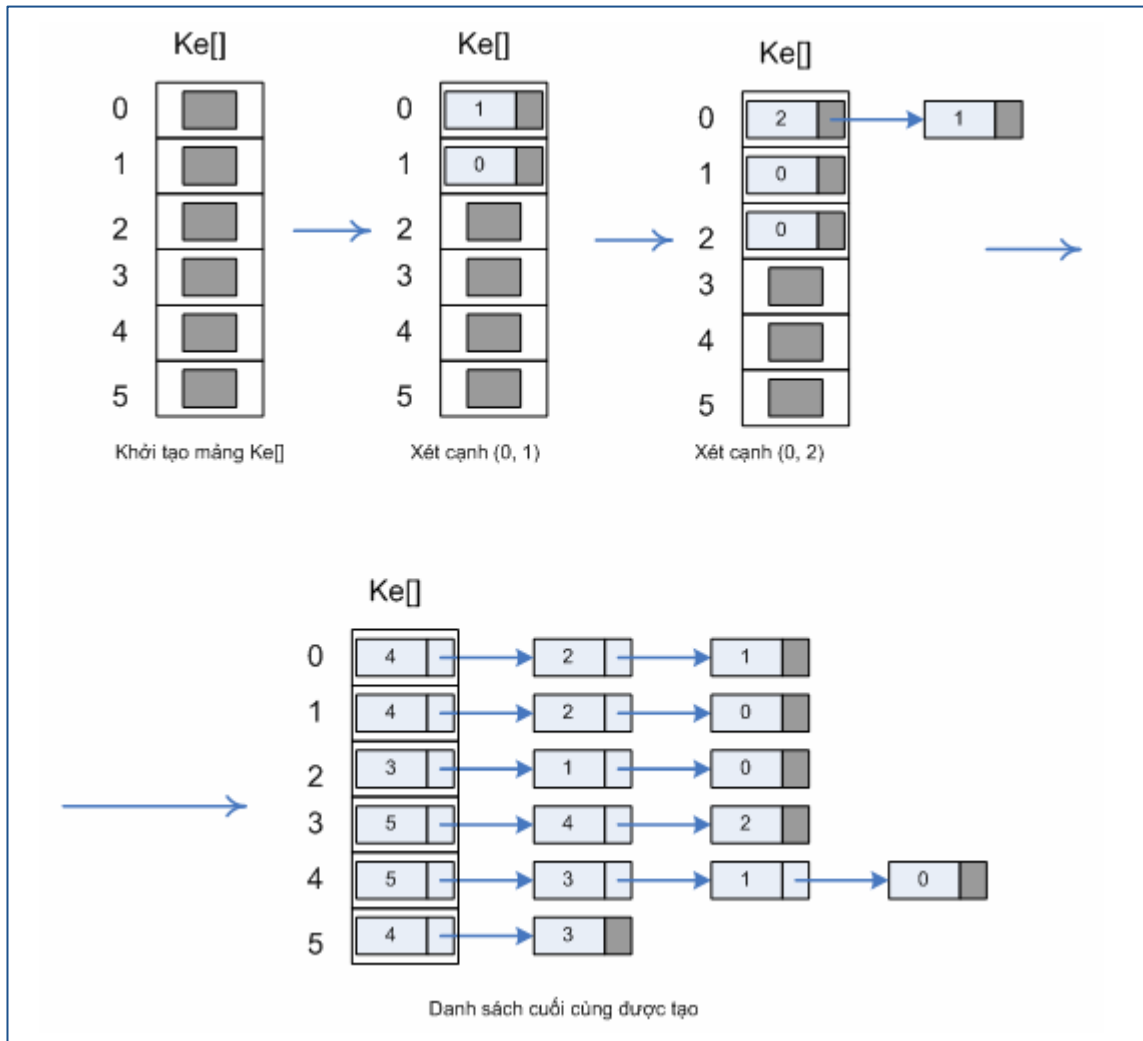
③ Thuật toán xây dựng danh sách kề liên kết



```
int main(int argc, char* argv[])
{
    cout<<"Cho so canh va so dinh cua do thi: ";
    cin>>m>>n;
    for(j=0;j<n;j++)
        ke[j]=NULL;
    for(j=1;j<=m;j++)
    {
        cout<<"Cho dinh dau, dinh cuoi cua canh "<<j<<":";
        cin>>x>>y;
        p = (node*)malloc(sizeof(node));
        p->v = x;
        p->next = ke[y];
        ke[y]=p;
        p = (node*)malloc(sizeof(node));
        p->v = y;
        p->next = ke[x];
        ke[x]=p;
    }
}
```

I.4. Danh sách kề

③ Ví dụ



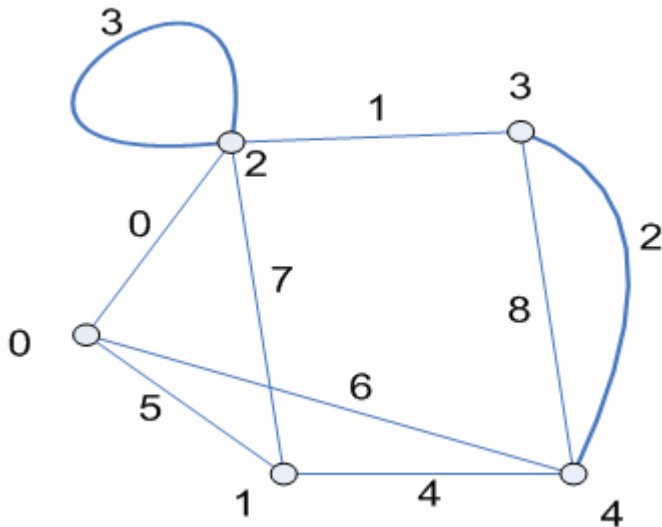
Đỉnh V	Các cạnh kề
0	1, 2, 4
1	0, 2, 4
2	0, 1, 3
3	2, 4, 5
4	0, 1, 3, 5
5	3, 4

I.5. Ma trận liên thuộc (đồ thị vô hướng)

Định nghĩa

③ Đồ thị vô hướng $G=(V, E)$. Tập đỉnh $V=\{0, 1, 2, \dots, n-1\}$. Tập cạnh $E=\{e_1, e_2, \dots, e_{m-1}\}$. Ta gọi ma trận liên thuộc của G là $B = \{b_{i,j}, i = 0, \dots, n-1, j = 0, \dots, m-1\}$. Trong đó

- $b_{i,j} = 1$ nếu đỉnh i kề cạnh j
- $b_{i,j} = 0$ nếu đỉnh i không kề cạnh j

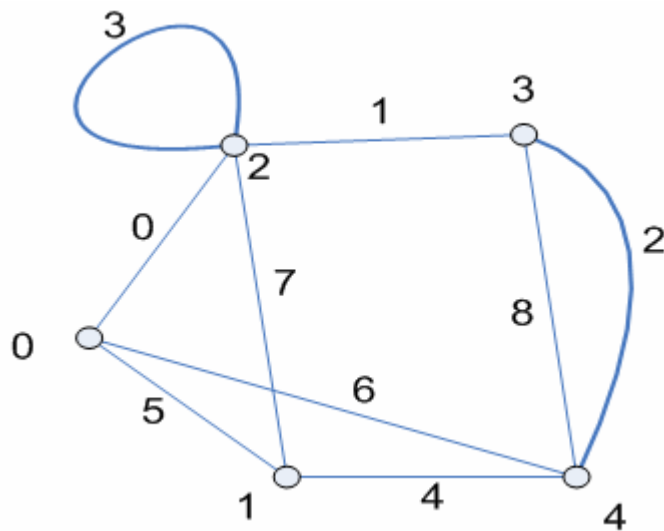


	0	1	2	3	4	5	6	7	8		
0	1	0	0	0	0	1	1	0	0		
1	0	0	0	0	1	1	0	1	0		
2	1	1	0	1	0	0	0	1	0		
3	0	1	1	0	0	0	0	0	1		
4	0	0	1	0	1	0	1	0	1		

I.5. Ma trận liên thuộc (đồ thị vô hướng)

Tính chất

- ③ Mỗi cột chứa đúng hai số 1 chỉ hai đầu của cạnh tương ứng với đỉnh ứng với cột đó. Cột ứng với khuyên chứa đúng một số 1.
- ③ Các cột ứng với các cạnh lặp thì giống nhau.
- ③ Nếu đồ thị không có khuyên thì tổng hàng i là bậc của đỉnh.



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

I.5. Ma trận liên thuộc (đồ thị có hướng)

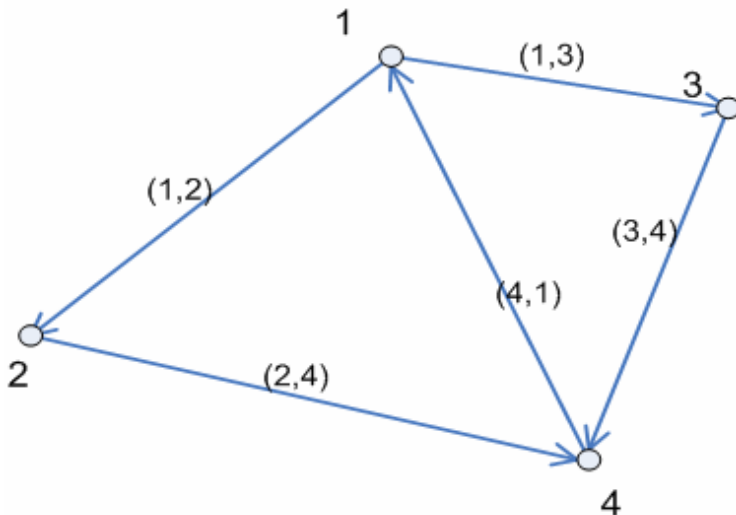
Định nghĩa

③ Đơn đồ thị có hướng $G=(V, E)$. Tập đỉnh $V=\{0, 1, 2, \dots, n-1\}$. Tập cung $E=\{e_1, e_2, \dots, e_{m-1}\}$.

Ta gọi ma trận liên thuộc của G là $B = \{b_{i,j}, i = 0, \dots, n-1, j = 0, \dots, m-1\}$.

Trong đó

- $b_{i,j} = 1$ nếu đỉnh i là đỉnh đầu của cung j
- $b_{i,j} = -1$ nếu đỉnh i là đỉnh cuối của cung j
- $b_{i,j} = 0$ nếu đỉnh i không là đầu mút của cung j



	(1,2)	(4,1)	(1,3)	(3,4)	(2,4)
1	1	-1	1	0	0
2	-1	0	0	0	1
3	0	0	-1	1	0
4	0	1	0	-1	-1

I. Các cách biểu diễn đồ thị

Các cách biểu diễn đồ thị

Ma trận kề

- ① n Đơn vị bộ nhớ
- ① Dễ kiểm tra đ/k kề nhau

Danh sách cạnh

- ① $2m$ Đơn vị bộ nhớ
- ① Đồ thị thưa
- ① Khó kiểm tra đ/k kề nhau

Danh sách kề

- ① $2m+n$ Đơn vị bộ nhớ
- ① Dễ dàng việc thêm bớt các cạnh, đỉnh

Ma trận liên thuộc

- ① $m*n$ Đơn vị bộ nhớ
- ① Dễ dàng việc thêm bớt các cạnh, đỉnh

Nội dung



Các cách biểu diễn đồ thị



Sự đẳng cấu của các đồ thị

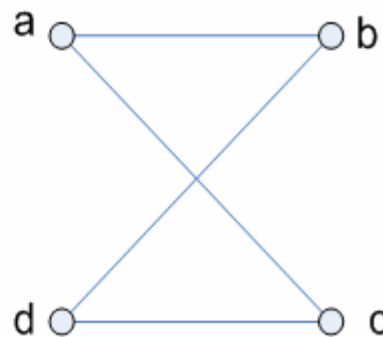
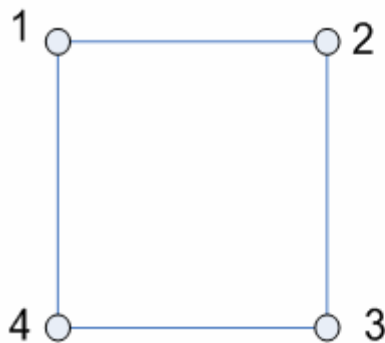


Hướng dẫn cài đặt

II. Sự đẳng cấu của các đồ thị

Định nghĩa

- ③ Các đồ thị đơn $G = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ là đẳng cấu nếu có hàm song ánh :
 $f : V_1 \rightarrow V_2$ sao cho \forall đỉnh a & b kề trong $G \Rightarrow f(a)$ & $f(b)$ kề trong G_2 .
- ③ \Leftrightarrow Tồn tại một phép tương ứng một – một giữa các đỉnh của hai đồ thị đồng thời đảm bảo quan hệ liên kề.

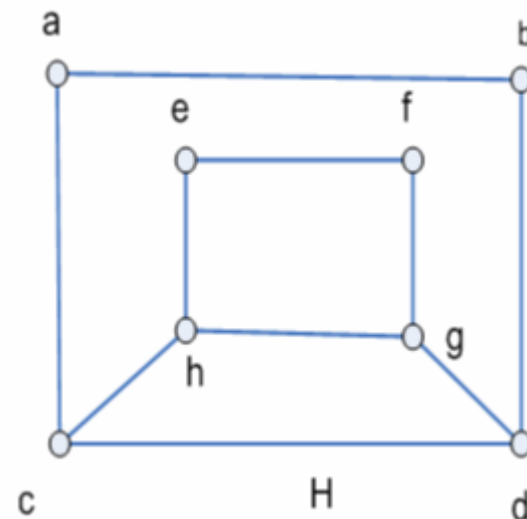
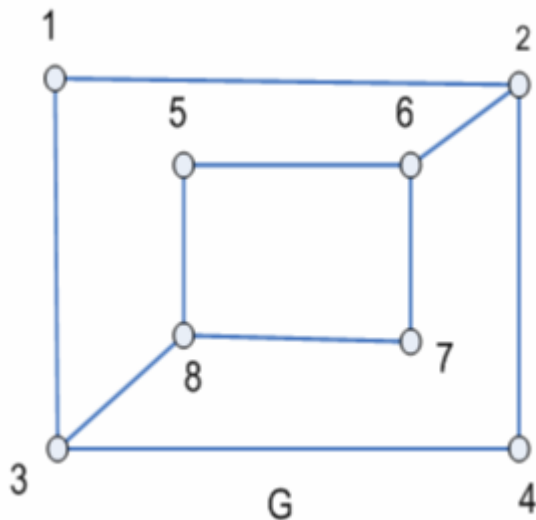


$$\begin{aligned} f(1) &= a, f(2) = b \\ f(3) &= d, f(4) = c \end{aligned}$$

II. Sự đẳng cấu của các đồ thị

Tính bất biến

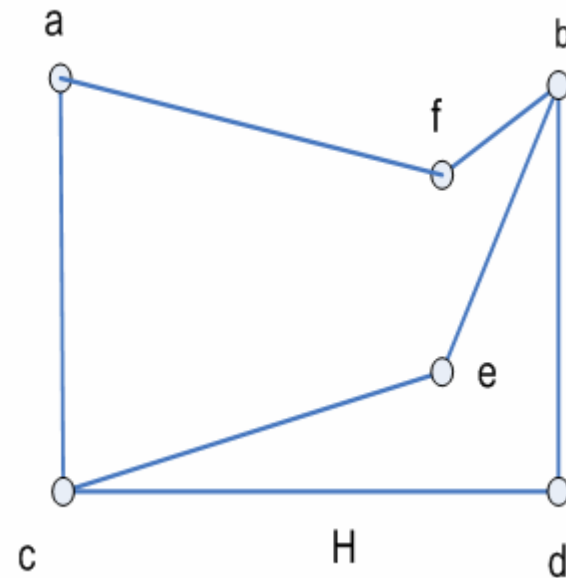
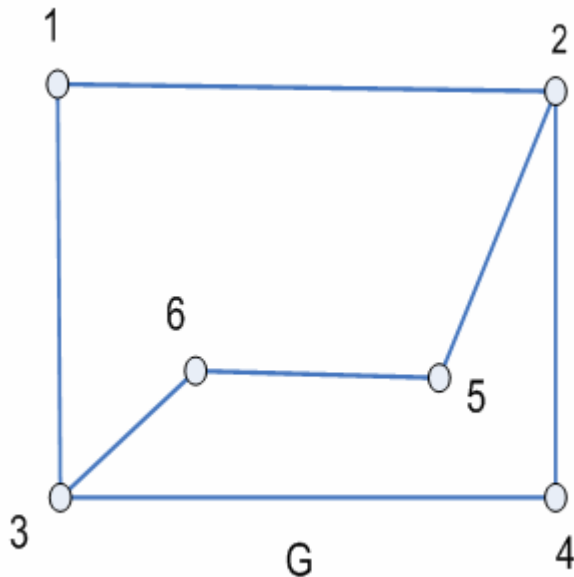
- ③ Hai đồ thị đẳng cấu bất kỳ có **tính chất giống nhau** (số đỉnh, số cạnh, bậc của một đỉnh,...). Người ta gọi đó là tính bất biến trong các đồ thị đẳng cấu.



II. Sự đẳng cấu của các đồ thị

Chứng minh 2 đồ thị là đẳng cấu

- ③ Tìm một ánh xạ f tương ứng một – một giữa các đỉnh
- ③ So sánh 2 ma trận liên kề tạo ra dựa trên ánh xạ f



Nội dung



I. Các cách biểu diễn đồ thị



II. Sự đẳng cấu của các đồ thị



III. Hướng dẫn cài đặt

III. Hướng dẫn cài đặt

- ③ Khai báo file
- ③ Kết nối biến file với tên thực của file ở trên đĩa (floppy or hard disk)
- ③ Mở file, đóng file
- ③ Đọc thông tin từ file và ghi thông tin vào file
- ③ Để hiểu tốt danh sách kê liên kết cần tham khảo phần biến con trỏ trong các tài liệu về lập trình.