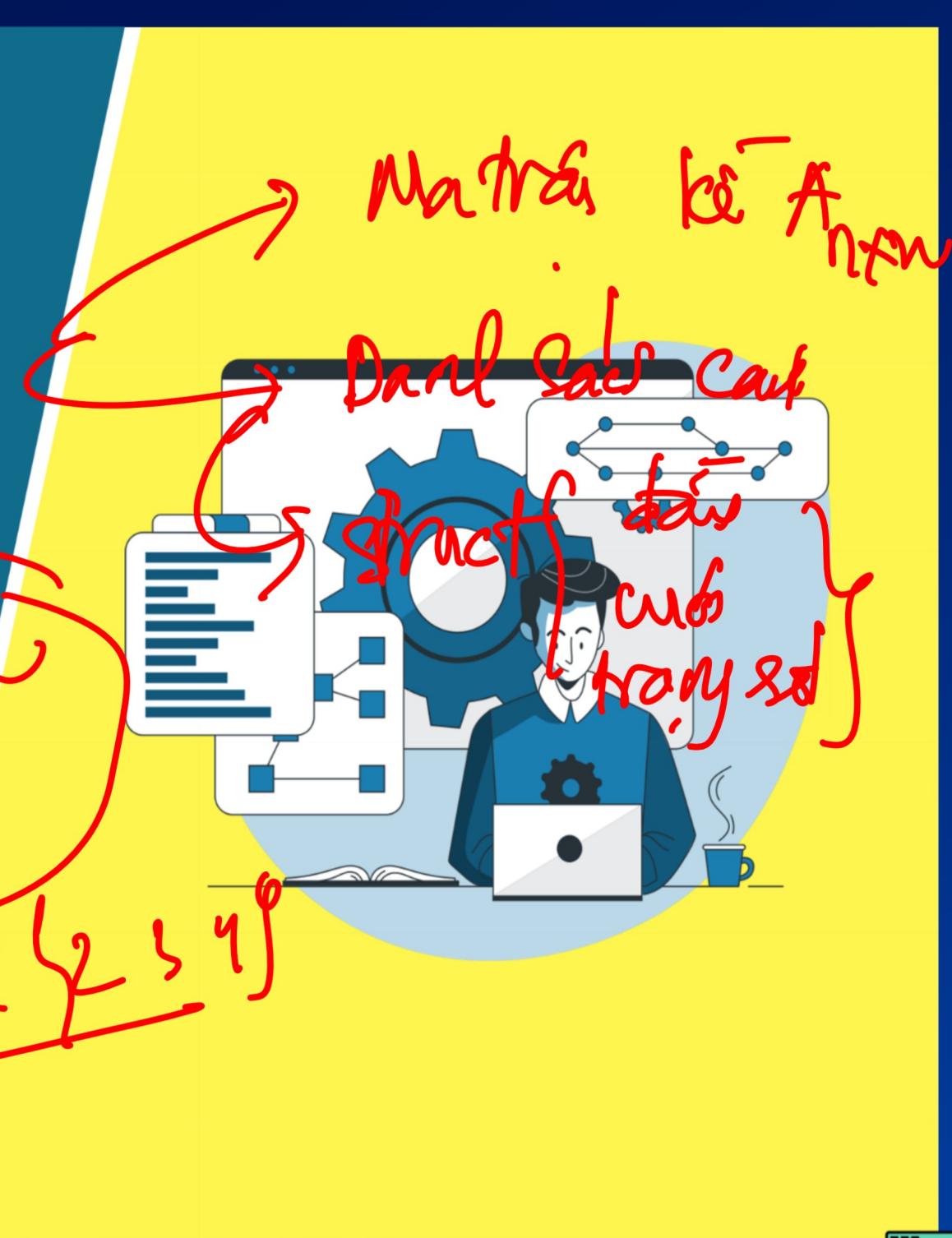




BIỂU DIỄN ĐỒ THỊ

Danh sách kí

3 cách để biểu diễn đồ thị





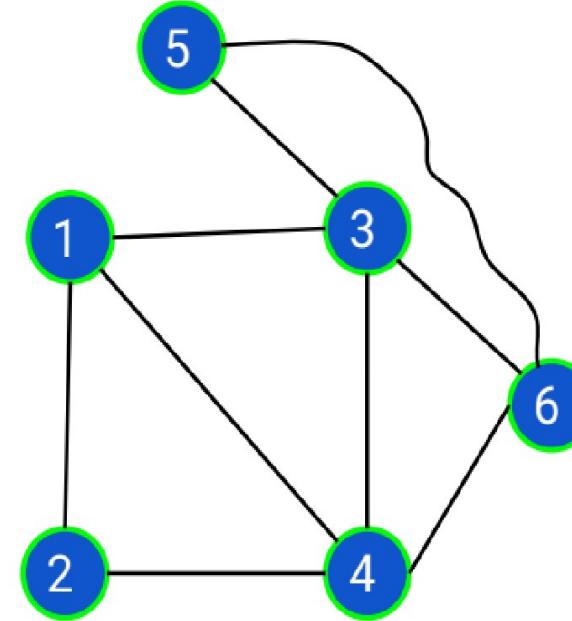
Ma trận kè:



Với đồ thị vô hướng ma trận kè của đồ thị có n đỉnh là ma trận vuông cỡ $n \times n$ có các phần tử là 0 hoặc 1. $A = \{a_{ij}, a_{ij} = 1$ nếu cạnh (i, j) là một cạnh của đồ thị, $a_{ij} = 0$ nếu cạnh (i, j) không là một cạnh của đồ thị}.

Tính chất

- Là ma trận đối xứng, tổng các phần tử trên ma trận bằng 2 lần số cạnh, tổng các phần tử trên hàng hoặc cột thứ u là bậc của đỉnh u .



	1	2	3	4	5	6
1	0	1	1	1	0	0
2	1	0	0	1	0	0
3	1	0	0	1	1	1
4	1	1	1	0	0	1
5	0	0	1	0	0	1
6	0	0	1	1	1	0

Ma trận kè của đồ thị vô hướng



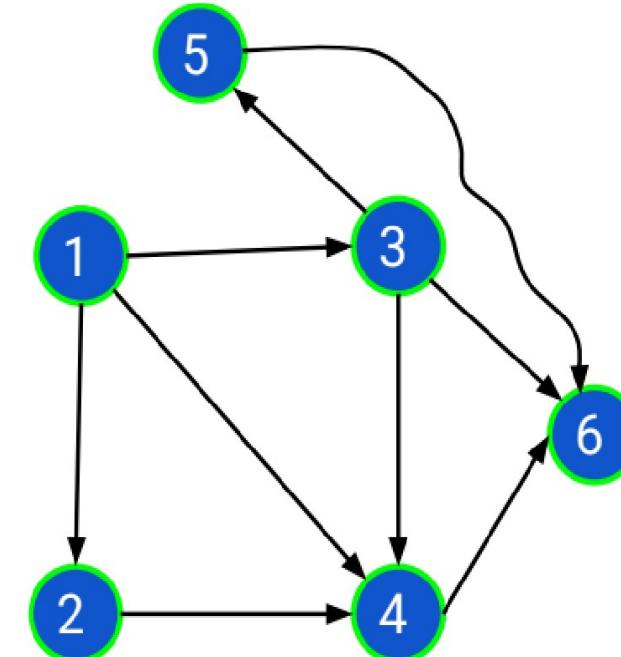
Ma trận kề:



Với đồ thị có hướng ma trận kề của đồ thị có n đỉnh là ma trận vuông cỡ $n \times n$ có các phần tử là 0 hoặc 1. $A = \{a_{ij}, a_{ij} = 1$ nếu cạnh (i, j) là một cạnh của đồ thị, $a_{ij} = 0$ nếu cạnh (i, j) không là một cạnh của đồ thị}.

Tính chất

- Có thể không đối xứng
- Tổng các phần tử của ma trận bằng số cạnh
- Tổng các phần tử trên hàng thứ u là bán bậc ra của đỉnh u
- Tổng các phần tử trên cột thứ u là bán bậc vào của đỉnh u



	1	2	3	4	5	6
1	0	1	1	1	0	0
2	0	0	0	1	0	0
3	0	0	0	1	1	1
4	0	0	0	0	0	1
5	0	0	0	0	0	1
6	0	0	0	0	0	0

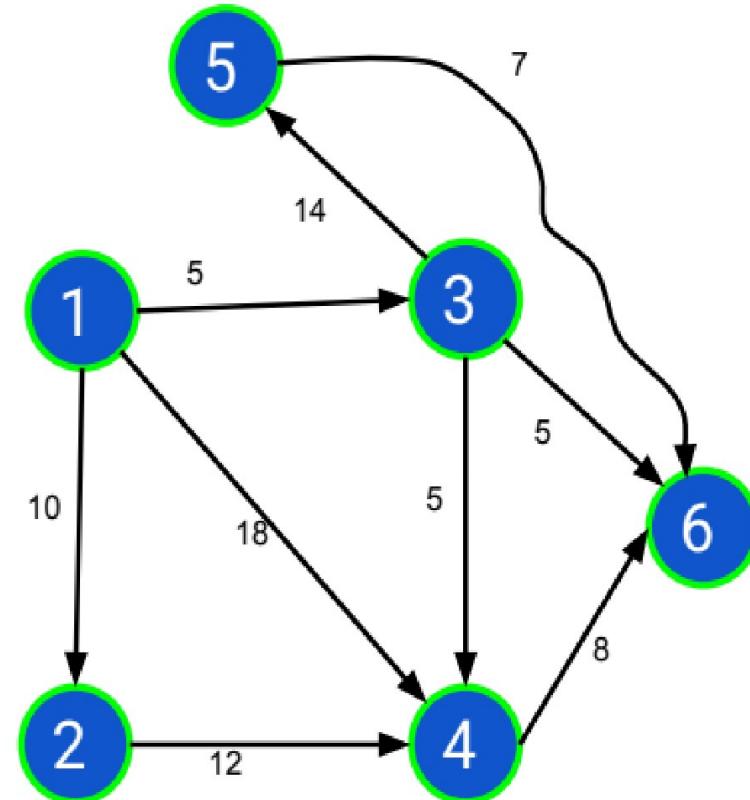
Ma trận kề của đồ thị có hướng



Ma trận trọng số:



Mỗi cạnh của đồ thị được gán một trọng số, giả sử mỗi cạnh $e = (u, v)$ có trọng số là $c(u, v)$. Ma trận trọng số $c = c[i, j]$, trong đó $c[i, j] = c(i, j)$ nếu (i, j) là một cạnh của đồ thị, $c[i, j] = 0$ (hoặc vô cùng, âm vô cùng tùy thuộc vào bài toán) nếu (i, j) không là một cạnh của đồ thị.



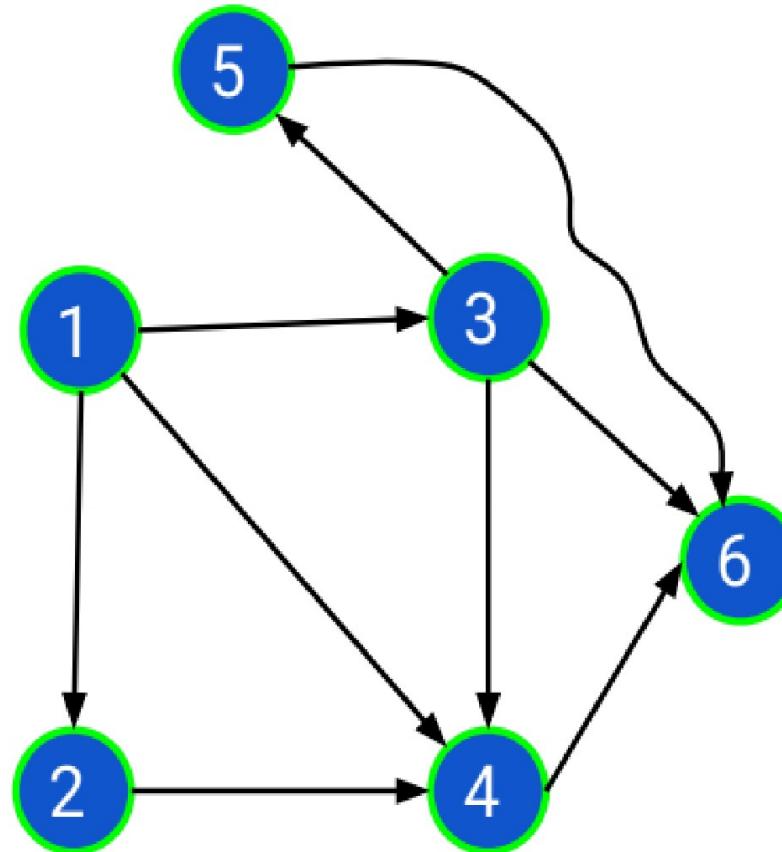
	1	2	3	4	5	6
1	0	10	5	18	0	0
2	0	0	0	12	0	0
3	0	0	0	5	14	5
4	0	0	0	0	0	8
5	0	0	0	0	0	7
6	0	0	0	0	0	0

Ma trận trọng số của đồ thị có hướng



Ma trận kề (adjacency matrix):

- ✓ **Ưu điểm:** Đơn giản, dễ cài đặt, dễ dàng kiểm tra 2 đỉnh có kề nhau hay không trong $O(1)$ bằng cách kiểm tra giá trị của $A[i, j]$.
- ✗ **Nhược điểm:** Tốn bộ nhớ nên không thể biểu diễn được đồ thị với số đỉnh lớn



	1	2	3	4	5	6
1	0	1	1	1	0	0
2	0	0	0	1	0	0
3	0	0	0	1	1	1
4	0	0	0	0	0	1
5	0	0	0	0	0	1
6	0	0	0	0	0	0

Ma trận kề của đồ thị có hướng



Danh sách cạnh:



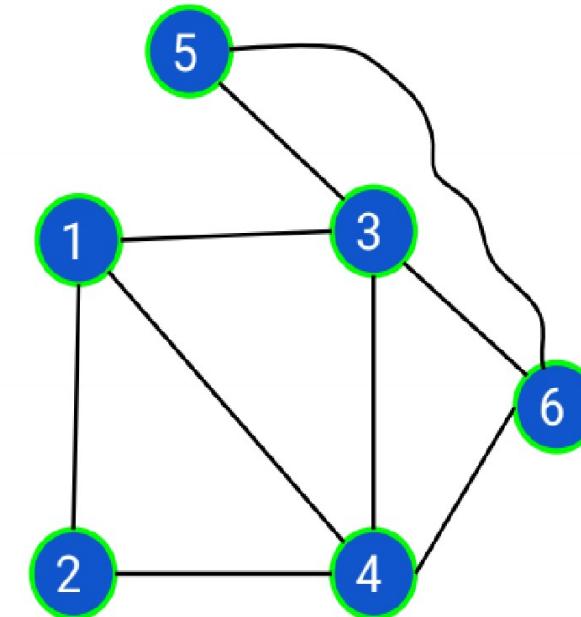
Danh sách cạnh: Thường được biểu diễn khi đồ thị thưa (số lượng cạnh \leq 6 lần số đỉnh).



Đối với đồ thị vô hướng, nếu tồn tại cạnh giữa 2 đỉnh u, v . Chỉ cần liệt kê cạnh (u, v) không cần liệt kê cạnh (v, u) , thường chọn $u < v$. Thường liệt kê các cạnh theo thứ tự tăng dần đỉnh đầu của các cạnh.



Đối với đồ thị có hướng, mỗi cạnh là bộ có tính đến thứ tự của các đỉnh.



Đỉnh đầu	Đỉnh cuối
1	2
1	3
1	4
2	4
3	4
3	5
3	6
4	6
5	6

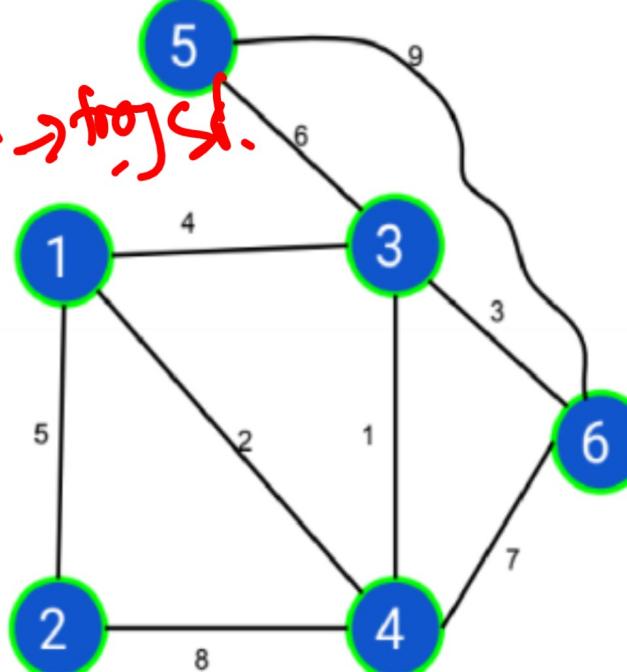


Danh sách cạnh:



Trong trường hợp đồ thị có trọng số, mỗi cạnh sẽ có thêm trọng số đi kèm đỉnh đầu và đỉnh cuối. Trong trường hợp danh sách cạnh không có trọng số có thể dùng pair<int, int> để biểu diễn thông tin một cạnh, với cạnh có trọng số có thể dùng tuple hoặc định nghĩa 1 struct lưu thông tin cạnh như sau:

struct edge{
 int dau, cuoi, w;
};



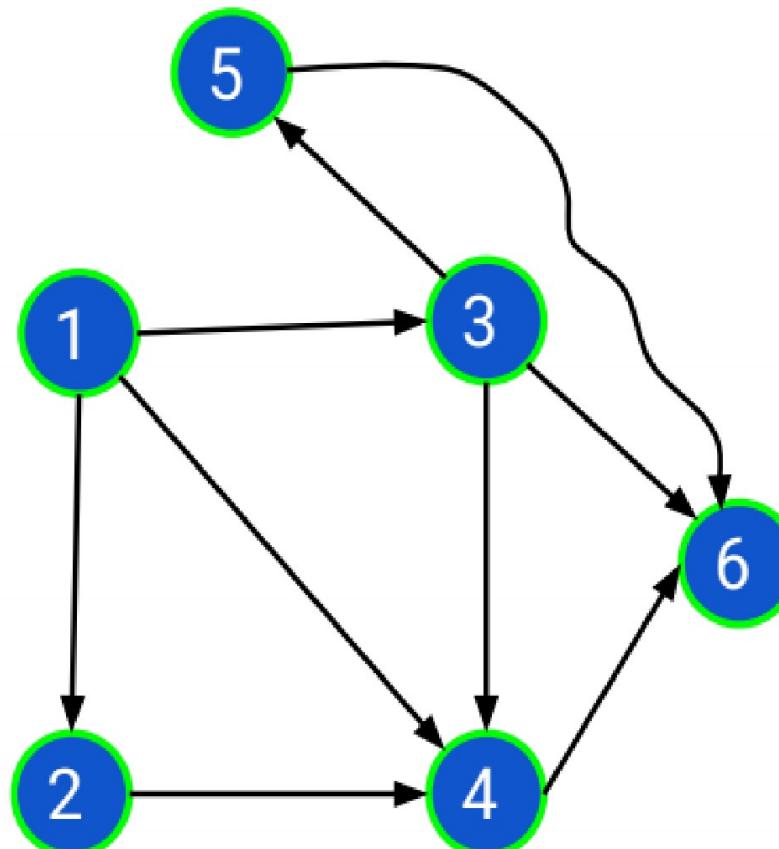
Đỉnh đầu	Đỉnh cuối	Trọng số
1	2	5
1	3	4
1	4	2
2	4	8
3	4	1
3	5	6
3	6	3
4	6	7
5	6	9



Danh sách cạnh:



Trong trường hợp đồ thị có hướng, chú ý tới hướng của cạnh. Với đồ thị có hướng có trọng số ta làm tương tự như với đồ thị vô hướng có trọng số.

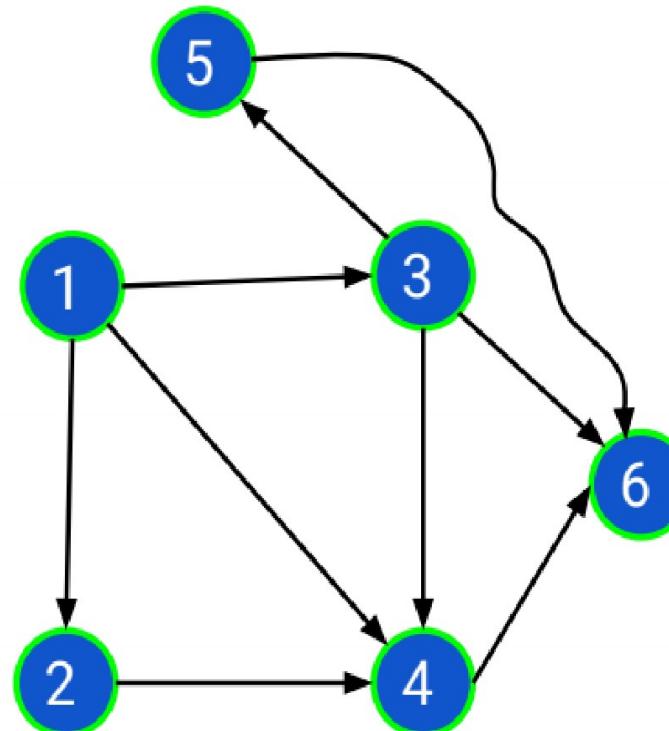


Đỉnh đầu	Đỉnh cuối
1	2
1	3
3	4
3	5
3	6
4	1
4	2
4	6
6	5



Danh sách cạnh:

- Ưu điểm:** Tiết kiệm được bộ nhớ nếu đồ thị thưa, thuận lợi cho các bài toán chỉ liên quan tới cạnh của đồ thị
- Nhược điểm:** Khi cần duyệt các đỉnh kề với đỉnh nào đó, bắt buộc phải duyệt tất cả các cạnh dẫn tới chi phí tính toán lớn.



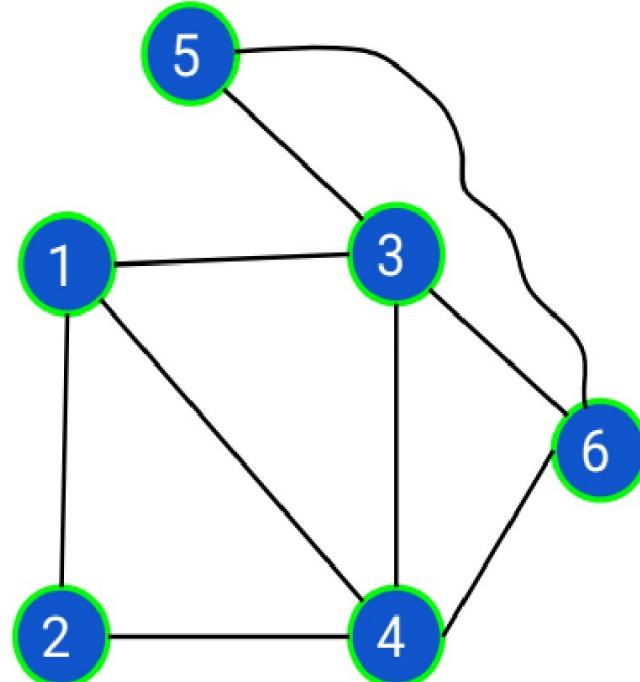
Đỉnh đầu	Đỉnh cuối
1	2
1	3
3	4
3	5
3	6
4	1
4	2
4	6
6	5



Danh sách kề (Adjacency list):

 Đối với mỗi đỉnh u của đồ thị, ta lưu trữ danh sách các đỉnh kề với đỉnh u . Trong C++ để lưu trữ danh sách kề của 1 đỉnh, ta dùng 1 vector. Khi đó để lưu trữ toàn bộ danh sách kề của các đỉnh ta dùng một mảng các vector.

 **Ví dụ** `vector<int> adj[1001];` Hoặc ví dụ dùng một vector các vector: `vector<vector<int,>> adj;`



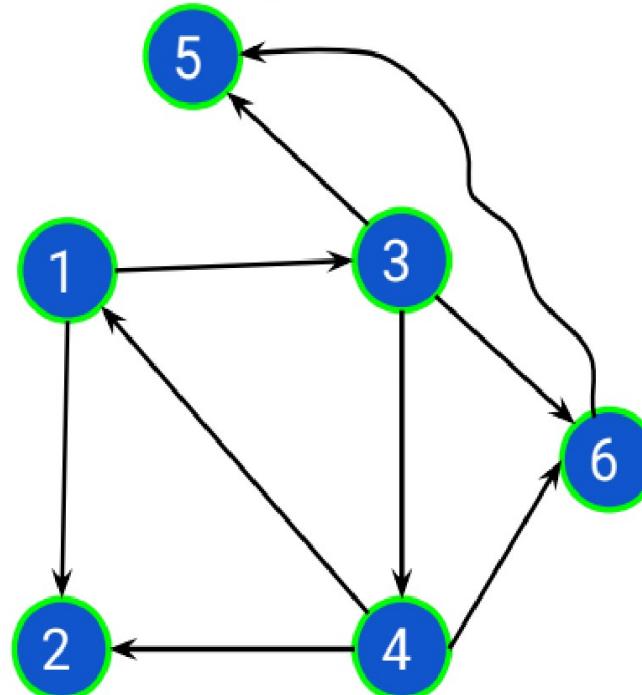
Đỉnh	Danh sách kề
1	{2, 3, 4}
2	{1, 4}
3	{1, 4, 5, 6}
4	{1, 2, 3, 6}
5	{3, 6}
6	{3, 4, 5}



Danh sách kề (Adjacency list):

 Đối với mỗi đỉnh u của đồ thị, ta lưu trữ danh sách các đỉnh kề với đỉnh u . Trong C++ để lưu trữ danh sách kề của 1 đỉnh, ta dùng 1 vector. Khi đó để lưu trữ toàn bộ danh sách kề của các đỉnh ta dùng một mảng các vector.

 **Ví dụ** `vector<int> adj[1001];` Hoặc ví dụ dùng một vector các vector: `vector<vector<int,>> adj;`

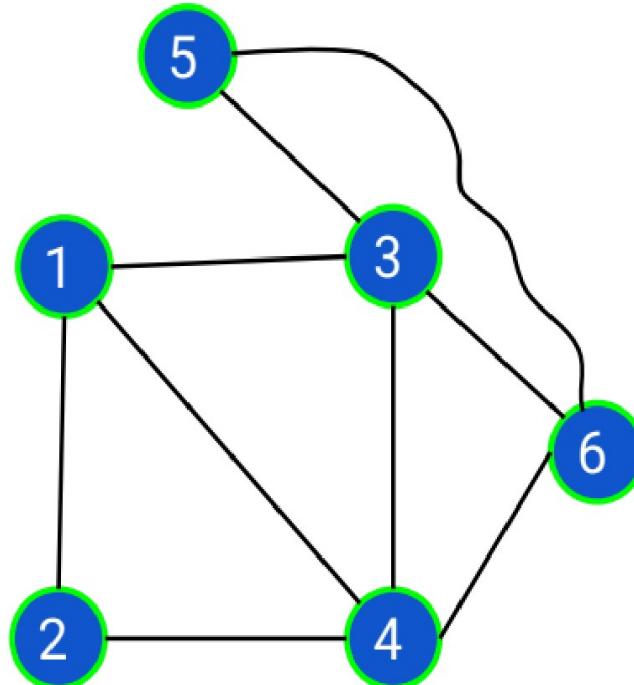


Đỉnh	Danh sách kề
1	{2, 3}
2	{}
3	{4, 5, 6}
4	{1, 2, 6}
5	{}
6	{5}



Danh sách kề (Adjacency list):

- ✓ **Ưu điểm:** Dễ dàng duyệt các đỉnh kề của một đỉnh; Dễ dàng duyệt các cạnh của đồ thị trong mỗi danh sách kề; Tối ưu về phương pháp biểu diễn.
- ✗ **Nhược điểm:** Khó khăn với người có kỹ năng lập trình yếu.



Đỉnh	Danh sách kề
1	{2, 3, 4}
2	{1, 4}
3	{1, 4, 5, 6}
4	{1, 2, 3, 6}
5	{3, 6}
6	{3, 4, 5}



+ Chuyển từ Dls sang trâu tò

⇒ Nhập file input w v $\Rightarrow \begin{cases} a[v][v] \\ a[w][w] \end{cases} = 1.$

Mảng vector $\text{vector}\langle \text{int} \rangle a[100]$ \rightarrow mảng vector

$a[i]$ vector
 $a[2]$