1. Lí thuyết
   1. Định nghĩa dãy bậc của đồ thị

Dãy bậc của đồ thị được định nghĩa như sau: Một dãy số tự nhiên được gọi là dãy bậc nếu tồn tại một đồ thị đơn vô hướng n đỉnh để bậc của đỉnh i là

VD: 1 3 1 1 là dãy bậc của 1 đơn đồ thị vô hướng với bậc của các đỉnh lần lần lượt là 1, 3, 1, 1

1. Định lí Edros – Galai kiểm tra 1 dãy không tăng có phải là 1 dãy bậc hay không
   1. Kiểm tra 1 dãy tăng dần có phải là 1 dãy bậc của đồ thị hay không

Trước hết, ta phát biểu định lí sau:

Cho 1 dãy không tăng , khi đó dãy đã cho là 1 dãy bậc của 1 đồ thị nếu thỏa mãn các tính chất sau:

1. là 1 số chẵn
   1. Thuật toán

**Input**: 1 dãy không tăng

**Output**: true nếu dãylà dãy bậc của đồ thị và ngược lại

\* **Thuật toán**:

- Kiểm tra điều kiện i: biến *sum\_degree* tính tổng tất cả giá trị của dãy . Kiểm tra *sum\_degree* có phải số chẵn hay không. Nếu không, kết thúc thuật toán. Nếu có, kiểm tra tiếp điều kiện ii.

- Kiểm tra điều kiện ii:

+ Với mỗi giá trị k từ 1 đến n, tạo 1 biến *prefix\_sum* tính tổng dãy bên vế trái của bất đẳng thức, biến *right\_sum* tính giá trị

+ Với mỗi giá trị k có được: một vòng lặp từ duyệt từ i = k đến n được thực hiện, biến *right\_sum +*

+ So sánh *right\_sum* và *prefix\_sum*: nếu *prefix\_sum right\_sum* thì true, ngược lại false

**Mã giả**:

function Erdos\_Gallai(degree sequence){

n = size of degree sequence

sum\_degree = 0

for i from 0 to n - 1

sum\_degree = sum\_degree + degree\_sequence[i]

if sum\_degree is odd

return false

prefix\_sum = 0

for k from 0 to n - 1

prefix\_sum = prefix\_sum + degree\_sequence[i]

right\_sum = k\*(k-1)

for i from k to n - 1

right\_sum = right\_sum + min(degree\_sequence[i], k)

if prefix\_sum > right\_sum

return false

return true

}

Định lí Edros – Galai chỉ ra một tiêu chuẩn thuần túy toán về điều kiện cần và đủ để một dãy số là dãy bậc của đồ thị. Nó không cho ta biết cách xây dựng đồ thị tương ứng như thế nào mà chỉ kiểm tra được 1 dãy đã cho có phải dãy bậc hay không. Một cách cụ thể hơn để xây dựng được 1 đồ thị từ dãy bậc đã xác định được thể hiện thông qua định lí Havel – Hakimi.

1. Thuật toán Havel – Hakimi
   1. Định lí

Dãy số tự nhiên là dãy bậc khi và chỉ khi dãy số cũng là dãy bậc

Định lí 3 cho ta một thuật toán đệ qui xác định một dãy số tự nhiên cho trước có phải là một dãy bậc hay không, đồng thời nó cũng cho ta cách xây dựng một đồ thị tương ứng với dãy bậc này. Tuy nhiên đồ thị thu được là một đồ thị hết sức đặc biệt, trong đó đỉnh có bậc cao nhất kề với tất cả các đỉnh có bậc cao tiếp theo.

* 1. Thuật toán

Input: Số lượng đỉnh là n, dãy bậc tương ứng

Ý tưởng:

+ Sắp xếp dãy trên thành dãy giảm dần, sau đó kiểm tra dãy đó cỏ thể xây dựng được thành 1 đồ thị hay không bằng thuật toán Edros – Galai. Nếu có thể thì ta tiến hành xây dựng

+ Tạo một ma trận *adj* với các giá trị đầu bằng 0 và mảng *resorted* là dãy bậc đã được sắp xếp theo thứ tự giảm dần

+ Với mỗi phần tử *resorted[i],* duyệt qua các phần tử có giá trị từ 1 đến *resorted[i]*, với mỗi bước duyệt, tăng giá trị của ma trận *adj*  tại vị trí (i, i + j) và (i + j, i) lên 1

+ Lặp lại cho đến khi tất cả giá trị của mảng *resorted* bằng 0, khi đó, ma trận *adj* là ma trận kề của đồ thị vừa được xây dựng

Mã giả:

resorted = sort(degree\_sequence)

n = size of resorted

if Erdos\_Gallai (degree\_sequence) = false

return false

adj[n][n]

for i from 0 to n - 1

for j from 1 to resorted[i]

if i + j < n

adj[i][i + j] = adj[i][i + j] + 1

adj[i + j][i] = adj[i + j][i] + 1

VD1: 1 dãy số 2 3 2 1 có thể xây dựng được thành 1 đồ thị hay không.

* Trước hết, ta kiểm tra dãy trên có phải là 1 dãy bậc của đồ thị hay không bằng định lí Edros – Galai.
* Sắp xếp dãy đã cho là thành dãy giảm dần: 3 2 2 1
* Tổng các giá trị trong dãy là: 8 là 1 số chẵn, thỏa được điều kiện đầu tiên của định lý
* Kiểm tra điều kiện ii:

+ Biến ­*prefix\_sum* = 8

+ Với i = 0: d[0] = 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | 0 | 1 | 2 | 3 |
| *Right\_sum* | 0 | 0 | 0 | 2 |
|  | 0 | 1 | 2 | 3 |
| Right\_sum + min( | 0 | 0 | 2 | 5 |

* *Prefix\_sum > Right\_sum: True*

+ Với i = 1: d[1] = 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | 0 | 1 | 2 | 3 |
| *Right\_sum* | 0 | 0 | 1 | 3 |
|  | 0 | 1 | 2 | 2 |
| Right\_sum + min( | 0 | 1 | 3 | 5 |

* *Prefix\_sum > Right\_sum: True*

+ Với i = 2: d[i] = 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | 0 | 1 | 2 | 3 |
| *Right\_sum* | 0 | 0 | 1 | 3 |
|  | 0 | 1 | 2 | 2 |
| Right\_sum + min( | 0 | 1 | 3 | 5 |

+ Với i = 3: d[i] = 1

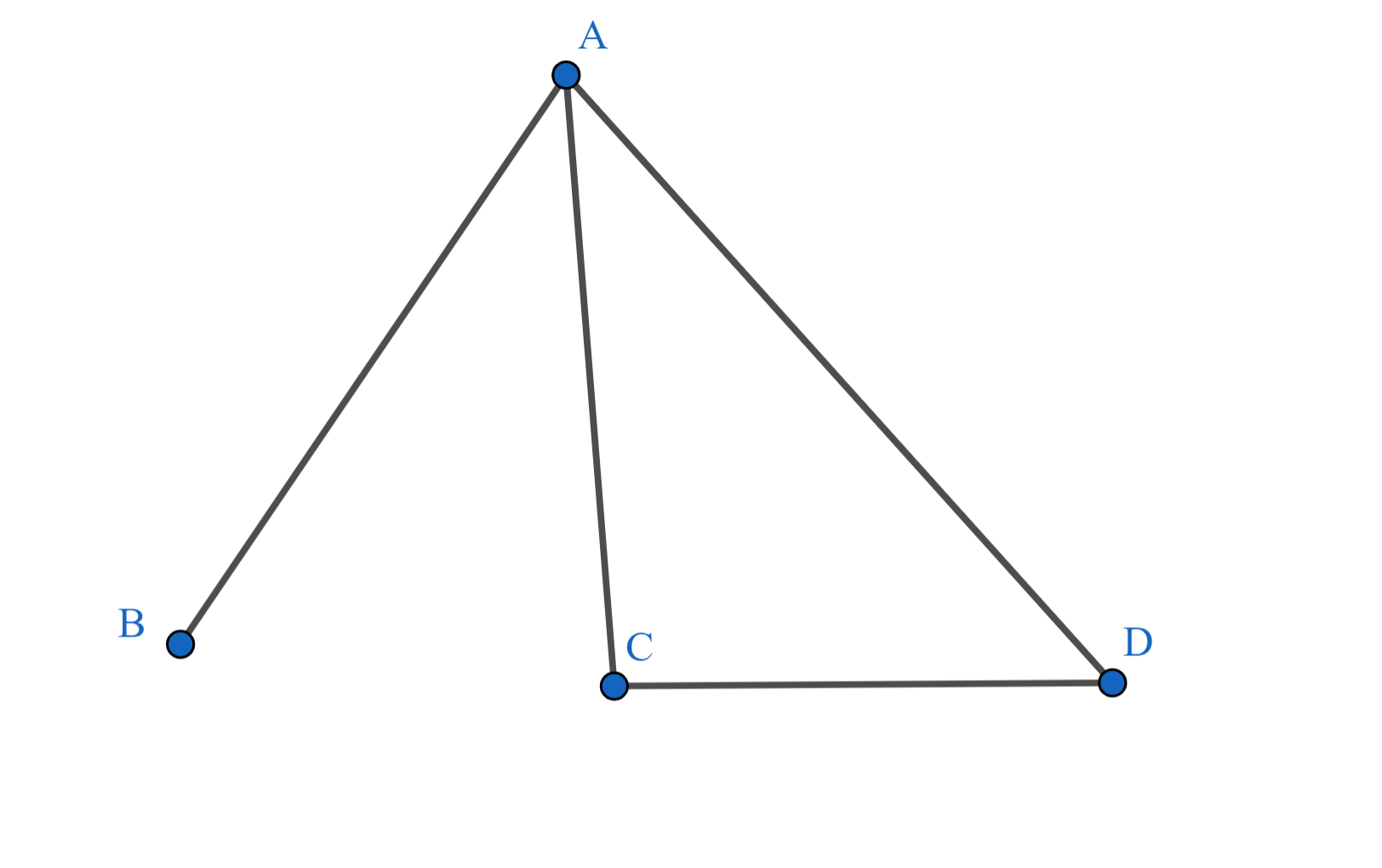
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | 0 | 1 | 2 | 3 |
| *Right\_sum* | 0 | 0 | 1 | 2 |
|  | 0 | 1 | 1 | 1 |
| Right\_sum + min( | 0 | 1 | 2 | 3 |

* *Prefix\_sum > Right\_sum: True*
* Như vậy: dãy đã cho là 1 dãy bậc của đồ thị
* Ta tiến hành xây dựng đồ thị:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đỉnh |  |  |  |  |
| Bậc | 3 | 2 | 2 | 1 |
|  | x | 1 | 1 | 0 |
|  |  | 0 | 0 | x |

* Lúc này ta có ma trận kề của đồ thị trên như sau:

Và phác họa đồ thị:



VD2: Áp dụng với dãy 0 1

Tương tự ta có 1 dãy giảm dần là: 1 0 và tổng các giá trị trong dãy bằng 1 là 1 số lẻ. Do đó không thể xây dựng được đồ thị với dãy 0 1

1. Cách chạy commandline trên code đã có

* Trước hết, ta tạo file a.exe bằng terminal bằng cách gõ dòng lệnh: g++ [your\_programm].cpp –o [name\_of\_application]

VD: g++ main.cpp –o a

* Mở commandline và dùng dòng lệnh [name\_of\_application].exe Input.dat Output.res

VD: a.exe Input.dat Output.res