**Tên đề: Anagram trong biểu diễn nhị phân**

**Mô tả**

Cho hai số nguyên không âm a và b, kiểm tra xem biểu diễn nhị phân của hai số đó có phải là anagram của nhau hay không.

Hai chuỗi được gọi là anagram nếu chúng có cùng số lượng ký tự '0' và '1', dù thứ tự các bit có thể khác nhau.

Ví dụ:

* a = 8 (1000₂), b = 4 (0100₂) → **Yes** (cả hai đều có 1 chữ số '1' và 3 chữ số '0').
* a = 4 (0100₂), b = 5 (0101₂) → **No** (số lượng chữ số '1' và '0' không giống nhau).

**Input**

* Dòng đầu tiên là số nguyên T — số lượng bộ test (1 ≤ T ≤ 10^5).
* T dòng tiếp theo, mỗi dòng chứa hai số nguyên không âm a và b (0 ≤ a, b ≤ 10^9).

**Output**

Với mỗi bộ test, in ra "Yes" nếu biểu diễn nhị phân của a và b là anagram, ngược lại in "No".

**Ràng buộc**

* Thời gian chạy tối đa: 1 giây
* Bộ nhớ tối đa: 256 MB

**Gợi ý**

* Chuyển hai số a và b sang dạng chuỗi nhị phân bằng hàm bin().
* Để so sánh dễ dàng, hãy sử dụng zfill() để làm cho hai chuỗi nhị phân có cùng độ dài (ví dụ 32 bit).
* Dùng collections.Counter để đếm số lượng '0' và '1' trong mỗi chuỗi, lưu dưới dạng dictionary.
* So sánh hai dictionary này: nếu bằng nhau thì in "Yes", ngược lại in "No".

Phương pháp này giúp tận dụng khả năng đếm và so sánh trực tiếp của dictionary trong python, tránh phải đếm thủ công từng bit, đồng thời đảm bảo độ dài chuỗi nhị phân bằng nhau để tránh sai lệch.

**Ví dụ**

**Input:**

3

8 4

4 5

0 0

**Output:**

Yes

No

Yes