



# Monk and his Friends

**Link submit:** <https://www.hackerearth.com/practice/data-structures/trees/binary-search-tree/practice-problems/algorithm/monk-and-his-friends/description/>

**Solution:**

	Không dùng thư viện	Dùng thư viện
C++	<a href="https://ideone.com/l6TV03">https://ideone.com/l6TV03</a>	<a href="https://ideone.com/klRC5e">https://ideone.com/klRC5e</a>
Java	<a href="https://ideone.com/qyUEwt">https://ideone.com/qyUEwt</a>	<a href="https://ideone.com/jvWMgc">https://ideone.com/jvWMgc</a>
Python	<a href="https://ideone.com/DM8sDV">https://ideone.com/DM8sDV</a> (TLE)	<a href="https://ideone.com/Coazk9">https://ideone.com/Coazk9</a>

**Tóm tắt đề:**

Có N sinh viên trong lớp học. Có M sinh viên khác đang đến, khi mỗi học sinh này đến, họ sẽ tìm đến ngồi cạnh với các học sinh khác đã có trong lớp mà số kẹo của họ đúng bằng nhau. Với mỗi học sinh trong M học sinh đó, in ra YES nếu như người đó có thể tìm được chỗ ngồi cạnh người có cùng số kẹo với mình, ngược lại thì in ra NO.

**Input:**

Dòng đầu tiên chứa T ( $1 \leq T \leq 10$ ) số lượng bộ test, mỗi bộ test gồm các thông tin sau:

- Dòng thứ nhất chứa hai số cách nhau bởi dấu khoảng trắng là N và M ( $1 \leq N, M \leq 10^5$ )
- Dòng thứ hai chứa N + M số lần lượt là N sinh viên và M sinh viên khác ( $0 \leq A_i \leq 10^{12}$ )

**Output:**

Mỗi bộ test in ra M dòng trả lời cho M sinh viên. In ra YES nếu tìm thấy, NO nếu không tìm thấy.

**Ví dụ:**

1	NO
2 3	NO
3 2 9 11 2	YES

**Giải thích ví dụ:**

Ví dụ trên gồm một bộ test. Trong bộ test đó, có hai sinh viên đã có sẵn trong lớp có số kẹo lần lượt là 3, 2.

Sinh viên đầu tiên đến có số kẹo là 9, trong lớp chưa có sinh viên có cùng số kẹo  $\Rightarrow$  NO.

Sinh viên thứ hai đến có số kẹo là 11, trong lớp chưa có sinh viên có cùng số kẹo  $\Rightarrow$  NO.

Sinh viên thứ ba đến có số kẹo là 2, trong lớp có sinh viên có cùng số kẹo  $\Rightarrow$  YES.

### Hướng dẫn giải:

Đầu tiên, bạn sẽ để N giá trị sinh viên vào trong cây nhị phân tìm kiếm.

Sau đó, bạn lần lượt thêm các sinh viên trong M sinh viên vào trong set. Nếu đã có rồi thì bạn xuất ra YES. Nếu chưa thì xuất ra NO.

**Độ phức tạp:**  $O(T * M * \log(N + M))$  với N và M lần lượt là số lượng sinh viên đã có sẵn và số lượng sinh viên vào lớp học và T là số lượng bộ test.

**\* Lưu ý:** Đối với C++ thì bài này yêu cầu xuất dữ liệu rất nhiều, nên để có thể Accepted được thì bạn cần giảm chi phí thời gian khi xuất dữ liệu, có 2 cách là sử dụng “\n” thay vì dùng endl hoặc thêm dòng lệnh: `ios_base::sync_with_stdio(false);` ở đầu chương trình.

“\n” khác endl ở chỗ endl ngoài việc xuống dòng, nó còn đưa toàn bộ dữ liệu trong buffer output stream ra output stream (tức xuất kết quả) mặc dù buffer (bộ nhớ đệm) vẫn còn trống, việc cứ phải đưa vào bộ nhớ đệm và lấy ra, xóa nhiều lần như vậy ảnh hưởng đến tốc độ xử lý. Còn “\n” chỉ xuống dòng, khi nào buffer đầy thì mới đưa ra output stream, giảm thiểu được số lần giải phóng bộ nhớ đệm.

Việc thêm dòng lệnh trên dùng để tăng tốc cho cin, cout của C++ nhanh như C bình thường vì bản chất cin, cout chậm hơn scanf và printf rất nhiều. Bạn có thể đọc thêm tài liệu ở đây.

[https://www.quora.com/What-is-use-of-the-statement-ios\\_base-sync\\_with\\_stdio-false-cin-tie-NULL-cout-tie-NULL-What-does-it-do](https://www.quora.com/What-is-use-of-the-statement-ios_base-sync_with_stdio-false-cin-tie-NULL-cout-tie-NULL-What-does-it-do)