# Teams

There is a class of $N$ students, numbered $0$ through $N-1$. Every day the teacher of the class has some projects for the students. Each project has to be completed by a team of students within the same day. The projects may have various difficulty. For each project, the teacher knows the exact size of a team that should work on it.

Different students may prefer different team sizes. More precisely, student $i$ can only be assigned to a team of size between $A[i]$ and $B[i]$ inclusive. On each day, a student may be assigned to at most one team. Some students might not be assigned to any teams. Each team will work on a single project.

The teacher has already chosen the projects for each of the next $Q$ days. For each of these days, determine whether it is possible to assign students to teams so that there is one team working on each project.

## Example

Suppose there are $N = 4$ students and $Q = 2$ days. The students' constraints on team sizes are given in the table below.

| student | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $A$ | 1 | 2 | 2 | 2 |
| $B$ | 2 | 3 | 3 | 4 |

On the first day there are $M = 2$ projects. The required team sizes are $K[0] = 1$ and $K[1] = 3$. These two teams can be formed by assigning student 0 to a team of size 1 and the remaining three students to a team of size 3.

On the second day there are $M = 2$ projects again, but this time the required team sizes are $K[0] = 1$ and $K[1] = 1$. In this case it is not possible to form the teams, as there is only one student who can be in a team of size 1.

## Task

You are given the description of all students: $N$, $A$, and $B$, as well as a sequence of $Q$ questions — one about each day. Each question consists of the number $M$ of projects on that day and a sequence $K$ of length $M$ containing the required team sizes. For each question, your program must return whether it is possible to form all the teams.

You need to implement the functions `init` and `can`:

- `init(N, A, B)` — The grader will call this function first and exactly once.
  - `N`: the number of students.

- A: an array of length N: A[i] is the minimum team size for student $i$.
- B: an array of length N: B[i] is the maximum team size for student $i$.
- The function has no return value.
- You may assume that $1 \leq$A[i]$\leq$B[i]$\leq$N for each $i = 0, \ldots,$N$-1$.

- can(M, K) — After calling init once, the grader will call this function $Q$ times in a row, once for each day.
  - M: the number of projects for this day.
  - K: an array of length M containing the required team size for each of these projects.
  - The function should return 1 if it is possible to form all the required teams and 0 otherwise.
  - You may assume that $1 \leq$M$\leq N$, and that for each $i = 0, \ldots,$M$-1$ we have $1 \leq$K[i] $\leq N$. Note that the sum of all K[i] may exceed $N$.

# Subtasks

Let us denote by $S$ the sum of values of M in all calls to can(M, K).

| subtask | points | $N$ | $Q$ | Additional Constraints |
|---|---|---|---|---|
| 1 | 21 | $1 \leq N \leq 100$ | $1 \leq Q \leq 100$ | none |
| 2 | 13 | $1 \leq N \leq 100,000$ | $Q = 1$ | none |
| 3 | 43 | $1 \leq N \leq 100,000$ | $1 \leq Q \leq 100,000$ | $S \leq 100,000$ |
| 4 | 23 | $1 \leq N \leq 500,000$ | $1 \leq Q \leq 200,000$ | $S \leq 200,000$ |

## Sample grader

The sample grader reads the input in the following format:

- line 1: N
- lines 2, …, N + 1: A[i] B[i]
- line N + 2: Q
- lines N + 3, …, N + Q + 2: M K[0] K[1] … K[M - 1]

For each question, the sample grader prints the return value of can.