Teams

Hay una clase con N estudiantes, enumerados desde 0 hasta $N-1$. Cada día el profesor de la clase tiene algunos proyectos para los estudiantes. Cada proyecto debe ser completado por un equipo de estudiantes dentro del mismo día. Los proyectos pueden tener diferentes dificultades. Por cada proyecto, el profesor sabe el tamaño exacto del equipo que debería trabajar en él.

Diferentes estudiantes pueden preferir diferentes tamaños de equipos. De manera más precisa, el estudiante i sólo puede ser asignado a un equipo con tamaño entre $A[i]$ y $B[i]$ inclusive. Por cada dia, un estudiante puede ser asignado a lo sumo a un equipo. Algunos estudiantes pueden no ser asignados a ningún equipo. Cada equipo trabajará con un solo proyecto.

El profesor ha seleccionado los proyectos para cada uno de los siguientes Q días. Para cada uno de esos días, determine si es posible asignar estudiantes a los equipos de tal manera que exista un equipo trabajando en cada proyecto.

Example

Suponga que existen $N = 4$ estudiantes y $Q = 2$ días. Las restricciones de los estudiantes para los tamaños de los equipos se aprecian en la tabla siguiente.

Estudiante	0	1	2	3
A	1	2	2	2
B	2	3	3	4

En el primer día hay $M = 2$ proyectos. Los tamaños de equipos requeridos para los proyectos son $K[0] = 1$ y $K[1] = 3$. Los dos equipos necesarios pueden ser formados asignando el estudiante 0 al equipo de tamaño 1 y los tres estudiantes restantes al equipo de tamaño 3.

En el segundo día existen $M = 2$ proyectos nuevamente, pero esta vez los tamaños de equipos requeridos para los proyectos son $K[0] = 1$ y $K[1] = 1$. En este caso no es posible formar los equipos, dado que sólo existe un estudiante dispuesto a formar un equipo de tamaño 1.

Task

A usted se le indicará la descripción de todos los estudiantes: N, A, y B, así como una secuencia de Q preguntas — una por cada día. Cada pregunta consiste en un número M de proyectos para ese día y una secuencia K de tamaño M incluyendo los tamaños de equipo requeridos por los proyectos. Para cada pregunta, su programa debe retornar si es posible formar todos los equipos necesarios.

Usted debe implementar las funciones `init` y `can`:

- `init(N, A, B)` — El grader invocará esta función primero y exactamente una vez.

- `N`: el número de estudiantes.
- `A`: un arreglo de tamaño `N`: `A[i]` es el mínimo tamaño de equipo aceptado por el estudiante i.
- `B`: un arreglo de tamaño `N`: `B[i]` el máximo tamaño de equipo aceptado por el estudiante i.
- Esta función no debe retornar nada.
- Usted puede asumir que $1 \leq$`A[i]`$\leq$`B[i]`$\leq$`N` para cada $i = 0, \ldots,$`N`-1.

- `can(M, K)` — Luego de invocar `init` una vez, el grader invocará esta función Q veces seguidas, una por cada día.
  - `M`: el número de proyectos para este día.
  - `K`: un arreglo de tamaño `M`, el cual, contiene el tamaño de equipo necesario para cada uno de los proyectos.
  - La función debe retornar 1 si es posible formar todos los equipos requeridos y 0 en caso contrario.
  - Usted puede asumir que $1 \leq$`M`$\leq N$, y para cada $i = 0, \ldots,$`M`-1 tenemos $1 \leq$`K[i]`$\leq N$. Note que la suma de todos los `K[i]` puede exceder N.

Subtasks

Sea S la suma de los valores de `M` en todas las llamadas a `can(M, K)`.

subtask	points	N	Q	Restricciones Adicionales
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	$1 \leq N \leq 500,000$	$1 \leq Q \leq 200,000$	$S \leq 200,000$

Implementation details

Usted debe enviar exactamente un archivo llamado `teams.c`, `teams.cpp`, `teams.pas`, o `teams.java`.

Dicho archivo debe implmentar las funciones descritas anteriormente como métodos o funciones utilizando las siguientes firmas.

C/C++ program (incluya `teams.h` en el inicio del archivo fuente)

```
void init(int N, int A[], int B[]);
int can(int M, int K[]);
```

Pascal programs (implement the described method in the unit `teams`)

```
procedure init(N : longint; var A, B : array of longint);
function can(M : longint; var K : array of longint) : longint;
```

Java programs (implement the described method in the public class `teams`)

```
void init(int N, int[] A, int[] B);
int can(int M, int[] K);
```

—>

Sample grader

El grader de ejemplo lee la entrada en el formato siguiente:

- línea 1: `N`
- líneas 2, …, `N` + 1: `A[i] B[i]`
- línea `N` + 2: `Q`
- líneas `N` + 3, …, `N` + `Q` + 2: `M K[0] K[1] … K[M - 1]`

Para cada pregunta, el grader ejemplo imprime el valor de retorno de la función `can`.