Roller Coaster Railroad

Anna trabaja en un parque de diversiones y está a cargo de determinar el diseño para una nueva montaña rusa. Ella ha diseñado n secciones especiales (convenientemente numeradas de 0 a $n-1$), las cuales afectan la velocidad de un tren de la montaña rusa. Ahora tiene que unirlas y proponer un diseño final para la montaña rusa. Para el propósito de este problema puedes asumir que el largo del tren es cero.

Para cada i entre 0 y $n-1$, inclusive, la sección especial i tiene dos propiedades:

- al ingresar a la sección, hay un límite de velocidad: la velocidad del tren debe ser **menor o igual a** s_i km/h (kilómetros por hora),
- al salir de la sección, la velocidad del tren es **exactamente** t_i km/h, sin importar la velocidad a la cual el tren ingresó a la sección.

Después de construida, la montaña rusa debe consistir en un único riel que contiene las n secciones especiales en un cierto orden. Cada una de las n secciones debe ser usada exactamente una vez. Dos secciones consecutivas están conectadas por vías. Anna debe escoger el orden de las n secciones y luego decidir los largos de las vías. El largo de una vía está medido en metros y puede ser igual a cualquier entero no negativo (posiblemente cero).

Cada metro de la vía entre dos secciones especiales frena la velocidad del tren en 1 km/h. Al comienzo del viaje, el tren ingresa a la primera sección especial en el orden seleccionado por Ana, viajando a 1 km/h.

El diseño final debe satisfacer las siguientes restricciones:

- el tren no viola ningún límite de velocidad al ingresar a las secciones especiales;
- la velocidad del tren es positiva en todo momento.

En todas las subtareas excepto la subtarea 3, tu labor es encontrar el mínimo largo total posible de las vías entre las secciones. En la subtarea 3 solo necesitas revisar si existe un diseño de montaña rusa válido, de modo que cada vía tenga largo cero.

Detalles de implementación

Tienes que implementar la siguiente función (método):

- `int64 plan_roller_coaster(int[] s, int[] t)`
  - `s`: arreglo de largo n, velocidades máximas de entrada permitidas.
  - `t`: arreglo de largo n, velocidades de salida.
  - En todas las subtareas excepto en la subtarea 3, la función debe retornar el mínimo largo total posible de todas las vías. En la subtarea 3 la función debe retornar 0 si existe un diseño válido en el que cada vía tiene largo cero, o cualquier entero positivo si es que tal diseño no existe.

Para el lenguaje C la firma de la función es ligeramente diferente:

- `int64 plan_roller_coaster(int n, int[] s, int[] t)`
  - `n`: el número de elementos en `s` y `t` (es decir, el número de secciones especiales),
  - los otros parámetros son los mismos que arriba.

Ejemplo

`int64 plan_roller_coaster([1, 4, 5, 6], [7, 3, 8, 6])`

En este ejemplo hay cuatro secciones especiales. La mejor solución es construirlas en el orden 0, 3, 1, 2, y conectarlas con vías de largos 1, 2, 0 respectivamente. Así es como un tren viaja a través de este riel:

- Inicialmente la velocidad del tren es 1 km/h.
- El tren comienza el viaje entrando a la sección especial 0.
- El tren abandona la sección 0 a una velocidad de 7 km/h.
- Luego hay una vía de largo 1 m. Cuando el tren llega al final de la vía, su velocidad es 6 km/h.
- El tren ingresa a la sección especial 3 a una velocidad de 6 km/h y la abandona a la misma velocidad.
- Después de salir de la sección 3, el tren viaja por una vía de 2m de largo. Su velocidad decrece a 4 km/h.
- El tren ingresa a la sección especial 1 a una velocidad de 4 km/h y la abandona a una velocidad de 3 km/h.
- Inmediatamente después de la sección especial 1 el tren ingresa a la sección especial 2.
- El tren abandona la sección especial 2. Su velocidad final es de 8 km/h.

La función debe retornar el largo total de las vías entre las secciones especiales: $1 + 2 + 0 = 3$.

Subtareas

En todas las subtareas $1 \le s_i \le 10^9$ y $1 \le t_i \le 10^9$.

1. (11 puntos): $2 \le n \le 8$,
2. (23 puntos): $2 \le n \le 16$,
3. (30 puntos): $2 \le n \le 200\,000$. En esta subtarea tu programa solo necesita revisar si la respuesta es cero o no. Si la respuesta es distinta de cero, cualquier entero positivo como respuesta es considerado correcto.
4. (36 puntos): $2 \le n \le 200\,000$.

Grader de ejemplo

El grader de ejemplo lee el input en el siguiente formato:

- Línea 1: entero n.
- Línea 2 + i, para i entre 0 y $n - 1$: enteros s_i y t_i.