Detectando Moléculas

La compañía para la que Petr trabaja ha construído una máquina detectora de moléculas. Cada molécula tiene un peso entero positivo. La máquina tiene un *rango de detección* $[l, u]$, siendo l y u enteros positivos. La máquina puede detectar correctamente un cierto conjunto de moléculas si y solo si dicho conjunto contiene un subconjunto de moléculas tal que su peso total pertenezca al rango de detección de la máquina.

Formalmente, sean n moléculas con pesos enteros positivos $w_0, \ldots, w_{n-1}$. La detección es exitosa si existe un conjunto de índices distintos $I = \{i_1, \ldots, i_m\}$ tal que $l \leq w_{i_1} + \ldots + w_{i_m} \leq u$.

Debido a las particularidades de la máquina, se sabe que la diferencia entre l y u siempre es mayor o igual que la diferencia de peso entre la molécula más pesada y la más liviana. Formalmente, $u - l \geq w_{max} - w_{min}$, donde $w_{max} = \max(w_0, \ldots, w_{n-1})$ y $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Su tarea es escribir un programa que, o bien encuentre algún subconjunto de moléculas con peso total en el rango de detección, o bien determine que no existe tal subconjunto.

Detalles de implementación

Deberá implementar una función:

- `int[] solve(int l, int u, int[] w)`
  - `l` y `u`: los extremos del rango de detección,
  - `w`: los pesos de las moléculas.
  - si existe el subconjunto pedido, la función debe retornar un arreglo de índices de moléculas que formen un subconjunto posible. Si hay múltiples respuestas correctas, retornar cualquiera de ellas.
  - si no existe el subconjunto pedido, la función deberá retornar un arreglo vacío.

Para el lenguaje C la signatura es levemente diferente:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: la cantidad de elementos de `w` (o sea, la cantidad de moléculas),
  - los demás parámetros son como se explicó antes.
  - en vez de retornar un arreglo de m índices, la función deberá escribir los índices en las primeras m posiciones del arreglo `result` y luego retornar m.
  - si no existe el subconjunto pedido, la función no debe escribir nada en el arreglo `result`, y deberá retornar 0.

Su programa puede escribir los índices en el arreglo retornado (o en el arreglo result en C) en cualquier orden.

Utilice los archivos de ejemplo provistos para ver los detalles de implementación en su lenguaje de programación.

Ejemplos

Ejemplo 1

`solve(15, 17, [6, 8, 8, 7])`
En este ejemplo se tienen cuatro moléculas, con pesos 6, 8, 8 y 7. La máquina puede detectar subconjuntos de moléculas con peso total entre 15 y 17, inclusive. Notar que $17 - 15 \geq 8 - 6$. El peso total de las moléculas 1 y 3 es $w_1 + w_3 = 8 + 7 = 15$, por lo que la función puede retornar `[1, 3]`. Otras respuestas posibles son `[1, 2]` ($w_1 + w_2 = 8 + 8 = 16$) y `[2, 3]` ($w_2 + w_3 = 8 + 7 = 15$).

Ejemplo 2

`solve(14, 15, [5, 5, 6, 6])`
En este ejemplo tenemos cuatro moléculas con pesos 5, 5, 6 y 6, y buscamos un subconjunto de ellas con un peso total entre 14 y 15, inclusive. Notar que $15 - 14 \geq 6 - 5$. No existe ningún subconjunto de moléculas con peso total entre 14 y 15 por lo que la función debe retornar un arreglo vacío.

Ejemplo 3

`solve(10, 20, [15, 17, 16, 18])`
En este ejemplo tenemos cuatro moléculas con pesos 15, 17, 16 y 18, y buscamos un subconjunto de ellas con peso total entre 10 y 20, inclusive. Nuevamente, notar que $20 - 10 \geq 18 - 15$. Cualquier subconjunto que contenga exactamente una molécula tiene peso total entre 10 y 20, y las respuestas correctas son: `[0]`, `[1]`, `[2]` y `[3]`.

Subtareas

1. (9 puntos): $1 \leq n \leq 100$, $1 \leq w_i \leq 100$, $1 \leq u, l \leq 1000$, todos los w_i son iguales.
2. (10 puntos): $1 \leq n \leq 100$, $1 \leq w_i, u, l \leq 1000$, y $\max(w_0, \ldots, w_{n-1}) - \min(w_0, \ldots, w_{n-1}) \leq 1$.
3. (12 puntos): $1 \leq n \leq 100$ y $1 \leq w_i, u, l \leq 1000$.
4. (15 puntos): $1 \leq n \leq 10\,000$ y $1 \leq w_i, u, l \leq 10\,000$.
5. (23 puntos): $1 \leq n \leq 10\,000$ y $1 \leq w_i, u, l \leq 500\,000$
6. (31 puntos): $1 \leq n \leq 200\,000$ y $1 \leq w_i, u, l < 2^{31}$.

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada con el siguiente formato:

- línea 1: enteros n, l, u.
- línea 2: n enteros: $w_0, \ldots, w_{n-1}$.