Detecting Molecules

Petr está trabajando para una compañía que ha construido una máquina para identificar moléculas. Cada molécula tiene un peso entero positivo. Esta máquina tiene un *rango de identificación* $[l, u]$, donde l y u son enteros positivos. La máquina puede identificar un conjunto de moléculas si este conjunto contiene un subconjunto de moléculas cuyo peso total pertenece al rango de identificación de la máquina.

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

Dadas las características de la máquina, se garantiza que la brecha entre l y u será mayor o igual que la brecha entre los pesos de la molécula más pesada y la molécula menos pesada. Formalmente, $u - l \geq w_{max} - w_{min}$, donde $w_{max} = \max(w_0, \ldots, w_{n-1})$ y $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Tu tarea es escribir un programa que encuentre cualquier subconjunto de moléculas con peso total en el rango de identificación, o que determine que tal subconjunto no existe.

Detalles de Implementación

Debes implementar una función (metódo):

- `int[] solve(int l, int u, int[] w)`
  - `l` y `u`: los extremos del rango de identificación,
  - `w`: pesos de las moléculas.
  - Si el subconjunto pedido existe, la función debe retornar un arreglo con los índices de las moléculas que lo forman. Si hubieran varias soluciones, retorna cualquiera de ellas.
  - Si el subconjunto pedido no existe, la función debe retornar un arreglo vacío.

Para el lenguaje C las definiciones de funciones difieren ligeramente:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: número de elementos en `w` (esto es, el número de moléculas),
  - los otros parámetros son iguales que la deficinión previa.
  - en vez de retornar un arreglo de m índices (como arriba), la función debe escribir los índices en las primeras m celdas del arreglo `result` y luego

retornar m.
- Si el subconjunto solicitado no existe, la función no debe escribir nada en `result` y retornar 0.

Tu programa debe escribir los índices en el arreglo retornado (o al arreglo `result` en C) en cualquier orden.

Por favor, utiliza los archivos plantillas provistos para los detalles de implementación en tu lenguaje de programación.

Ejemplos

Ejemplo 1

`solve(15, 17, [6, 8, 8, 7])`
En este ejemplo tenemos cuatro moléculas con pesos 6, 8, 8 y 7. La máquina puede identificar los subconjuntos de moléculas con peso total entre 15 y 17, inclusive. Nota que $17 - 15 \geq 8 - 6$. El peso total de las moléculas 1 y 3 es $w_1 + w_3 = 8 + 7 = 15$, entonces la función puede retornar `[1, 3]`. Otra solución correcta posible es `[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 estamos buscando un subconjunto de él con peso total entre 14 y 15, inclusive. Nuevamente, nota que $15 - 14 \geq 6 - 5$. No hay subconjunto de moléculas con peso total entre 14 y 15 sentonces 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 estamos buscando un subconjunto de él con peso total entre 14 y 15, inclusive. Nuevamente, nota que $20 - 10 \geq 18 - 15$. Cualquier subconjunto de exactamente un solo elemento tiene peso total entre 10 y 20, entonces las posibles 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 pesos 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 $1 \leq w_i, u, l < 2^{31}$.

Grader de ejemplo

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

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