Detecting Molecules

Peter esta trabajando para una compañia que ha construido una maquina para detectar moléculas. Cada molécula tiene un peso expresado como un valor entero. La máquina tiene un *rango de detencion* $[l, u]$, donde l y u son enteros. La máquina puede detectar un subconjunto de moléculas solo si este conjunto contiene un subconjunto de moléculas con un peso total que corresponda al rango de detención de la máquina.

Oficialmente, considera n móleculas con pesos enteros positivos $w_0, \ldots, w_{n-1}$. La detección es satisfactoria si existe un conjunto de indices distintos $I = i_1, \ldots, i_m$ de tal manera que $l \leq w_{i_1} + \ldots w_{i_m} \leq u$.

Pese a las especificaciones de la máquina, el rango entre l y u esta garantizado para ser más grande o igual que el peso del rango 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 $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Tu trabajo consiste es escribir un programa que pueda hallar cualquier subconjunto de moleculas con un peso total comprendido entre el rango de detección, o determinar que no existe dicho subconjunto.

Detalles de Implementación

Debes implememntar una funcion (método):

- `int[] solve(int l, int u, int[] w)`
  - `l` y `u`: Los puntos finales del rango de detención,
  - `w`: los pesos de las moléculas
  - Si el subconjunto requerido existe, la función debe retornar un vector con los indices de las moléculas que forman el subconjunto deseado. Si existen muchas respuestas correctas, devuelva cualquiera de ellas.
  - Si el subconjunto requerido no existe, la función debe retornar un vector vacio.

Para el Lenguaje C el modelo de la función es un poco diferente:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: el número de elementos en W (es decir, el número de moléculas)
  - Los otros parametros son los mismo que los mencionados arriba.
  - instead of returning an array of m indices (as above), the function should write the indices to the first m cells of array `result` and then return m.

- En vez de retornar un vector de m indices(Como arriba), la función debe escribir los indices de las primeras m celdas del vector `result` y retornar m.
- Si el subconjunto requerido no existe, la función no debe escribir nada del vector resultantes y debe devolver 0

Tu programa debe escribir los indices en el vector de retorno (o el vector `result` en C) en cualqier orden.

Por favor usa las plantillas de archivos para la implementación de los detalles en tu lenguaje de programación.

Ejemplos

Ejemplo 1

`solve(15, 17, [6, 8, 8, 7])`
En este ejemplo tenemos cuatro moléculas con un peso de 6, 8, 8 y 7. La máquina puede detectar subconjuntos de moléculas con un total de pesos comprendidos entre 15 y 17 inclusive. Toma en cuenta, 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]`. Otras posibles respuestas correctas serian `[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 debemos encontrar el subconjunto con pesos entre 14 y 15 inclusive. Nuevamente, ten encuenta que $15 - 14 \geq 6 - 5$. No existe un subconjunto de moléculas con un peso total comprendido entre 14 and 15, es así que la función debe retornar un vector vacio.

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 ellas con un pesos total comprendido entre 10 y 20 inclusive. Nuevamente, ten en cuenta que $20 - 10 \geq 18 - 15$. Cualquier subconjunto de exactamente un elemento satisfce los requerimientos, de esa manera las respuestas serían: `[0]`, `[1]`, `[2]` y `[3]`.

SubTareas

1. (9 puntos): $n \leq 100$, $w_i \leq 100$, todos los w_i son iguales.
2. (10 puntos): $n \leq 100, w_i \leq 1000$, y $max(w_0, \ldots, w_{n-1}) - min(w_0, \ldots, w_{n-1}) \leq 1$.
3. (12 puntos): $n \leq 100$ y $w_i, u, l \leq 1000$.

4. (15 puntos): $n \leq 10\,000$ y $w_i, u, l \leq 10\,000$.
5. (23 puntos): $n \leq 10\,000$ y $w_i, u, l \leq 500\,000$
6. (31 puntos): $n \leq 200\,000$ y $w_i, u, l < 2^{31}$.

Sample grader

El Grader de muestra lee las entradas con el siguiente formato:

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