Molekulların aşkarlanması

Pyotr molekulları aşkarlayan maşın düzəltmiş şirkətdə çalışır. Hər bir molekul tam ədədlə ölçülən çəkiyə malikdir. Maşın $[l, u]$ *aşkarlama aralığına* malikdir, burada l və u tam ədədlərdir. Maşın molekullar çoxluğunu yalnız o zaman aşkarlaya bilər ki, həmin çoxluğun elə altçoxluğu var ki, ora daxil olan molekulların çəkiləri cəmi maşının aşkarlama aralığında yerləşsin.

Formal olaraq, müsbət tam $w_0, \ldots, w_{n-1}$ çəkiləri olan n molekula baxaq. Aşkarlama o zaman uğurlu olur ki, elə fərqli $I = i_1, \ldots, i_m$ indeksləri var ki, $l \leq w_{i_1} + \ldots w_{i_m} \leq u$.

Maşının özəlliyi sayəsində l və u arasındakı intervalın ən ağır və an yüngül molukulun çəkiləri arasındakı intervaldan böyük və ya bərabər olmasına zəmanət verilir. Formal olaraq, $u - l \geq w_{max} - w_{min}$, burada $w_{max} = \max(w_0, \ldots, w_{n-1})$ və $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Siz elə proqram yazmalısınız ki, toplam çəkiləri aşkarlama aralığında olan molekullar altçoxluğunu tapsın və ya belə altçoxluğun olmadığını müəyyənləşdirsin.

Implementation details

You should implement one function (method):

- `int[] solve(int l, int u, int[] w)`
  - `l` and `u`: the endpoints of the detection range,
  - `w`: weights of the molecules.
  - if the required subset exists, the function should return an array of indices of molecules that form any one such subset. If there are several correct answers, return any of them.
  - if the required subset does not exist, the function should return an empty array.

For the C language the function signature is slightly different:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: the number of elements in `w` (i.e., the number of molecules),
  - the other parameters are the same as above.
  - 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.
  - if the required subset does not exist, the function should not write anything to the `result` array and it should return 0.

Please use the provided template files for details of implementation in your programming language.

Examples

Example 1

`solve(15, 17, [6, 8, 8, 7])`
In this example we have four molecules with weights 6, 8, 8 and 7. The machine can detect subsets of molecules with total weight between 15 and 17, inclusive. Note, that $17 - 15 \geq 8 - 6$. The total weight of molecules 1 and 3 is $w_1 + w_3 = 8 + 7 = 15$, so the function can return `[1, 3]`. Other possible correct answers are `[1, 2]` ($w_1 + w_2 = 8 + 8 = 16$) and `[2, 3]` ($w_2 + w_3 = 8 + 7 = 15$).

Example 2

`solve(14, 15, [5, 5, 6, 6])`
In this example we have four molecules with weights 5, 5, 6 and 6, and we are looking for a subset of them with total weight between 14 and 15, inclusive. Again, note that $15 - 14 \geq 6 - 5$. There is no subset of molecules with total weight between 14 and 15 so the function should return an empty array.

Example 3

`solve(10, 20, [15, 17, 16, 18])`
In this example we have four molecules with weights 15, 17, 16 and 18, and we are looking for a subset of them with total weight between 10 and 20, inclusive. Again, note that $20 - 10 \geq 18 - 15$. Any subset consisting of exactly one element satisfies the requirement, so correct answers are: `[0]`, `[1]`, `[2]` and `[3]`.

Subtasks

1. (9 points): $n \leq 100$, $w_i \leq 100$, all w_i are equal.
2. (10 points): $n \leq 100, w_i \leq 1000$, and $max(w_0, \ldots, w_{n-1}) - min(w_0, \ldots, w_{n-1}) \leq 1$.
3. (12 points): $n \leq 100$ and $w_i, u, l \leq 1000$.
4. (15 points): $n \leq 10\,000$ and $w_i, u, l \leq 10\,000$.
5. (23 points): $n \leq 10\,000$ and $w_i, u, l \leq 500\,000$
6. (31 points): $n \leq 200\,000$ and $w_i, u, l < 2^{31}$.

Sample grader

The sample grader reads the input in the following format:

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