Detecting molecules

Petru lucrează pentru o companie care a construit un dispozitiv pentru detectarea moleculelor. Fiecare moleculă are o greutate pozitivă întreagă. Dispozitivul are un *interval de detectare* $[l, u]$, unde l și u sunt numere întregi, pozitive. Dispozitivul poate detecta un set de molecule dacă și numai dacă acest set conține un subset de molecule cu greutate totală aparținând intervalului de detectare a dispozitivului.

Formal, considerăm n molecule cu greutăți întregi $w_0, \ldots, w_{n-1}$. Detectarea se consideră reușită dacă există un set de indici distincți $I = \{i_1, \ldots, i_m\}$ astfel încât $l \leq w_{i_1} + \ldots + w_{i_m} \leq u$.

Datorită specificului dispozitivului, decalajul între l și u este garantat mai mare sau egal cu decalajul de greutate dintre cea mai grea și cea mai ușoară moleculă. Formal, $u - l \geq w_{max} - w_{min}$, unde $w_{max} = \max(w_0, \ldots, w_{n-1})$ și $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Scrie un program care fie găsește un subset de molecule cu greutate totală în intervalul de detectare, fie determină că nu există un asemenea subset.

Detalii de implementare

Trebuie să implementezi funcția (metoda):

- `int[] solve(int l, int u, int[] w)`
  - `l` și `u`: punctele extreme ale intervalului de detectare,
  - `w`: greutățile moleculelor.
  - dacă subsetul căutat există, funcția trebuie să returneze un array de indici a moleculelor care formează acest subset. Dacă există mai multe răspunsuri corecte, se va returna oricare din ele.
  - dacă subsetul căutat nu există, funcția trebuie să returneze un array vid.

Pentru limbajul C signatura funcției este ușor diferită:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: numărul de elemente în `w` (altfel vorbind - numărul de molecule),
  - ceilalți parametri sunt identici celor descriși anterior.
  - în locul returnării unui array de m indici (ca în cazul descris anterior), funcția trebuie să înscrie indicii în primele m elemente din array-ul `result` iar apoi să returneze m.
  - dacă subsetul căutat nu există, funcția nu va scrie nimic în array-ul `result` și va returna 0.

Programul poate scrie indicii în array-ul returnat (sau în array-ul `result` pentru limbajul C) în oricare ordine.

Te rugăm să folosești fișierele șablon furnizate pentru detalii de implementare în limbajul de programare pe care îl folosești.

Exemple

Exemplul 1

`solve(15, 17, [6, 8, 8, 7])`
În acest exemplu avem patru molecule cu greutățile 6, 8, 8 și 7. Dispozitivul poate detecta subseturi de molecule cu greutate totală cuprinsă între 15 și 17, inclusiv. De remarcat, că $17 - 15 \geq 8 - 6$. Greutatea totală a moleculelor 1 și 3 este $w_1 + w_3 = 8 + 7 = 15$, astfel funcția poate returna `[1, 3]`. Alte răspunsuri corecte posibile sunt `[1, 2]` ($w_1 + w_2 = 8 + 8 = 16$) și `[2, 3]` ($w_2 + w_3 = 8 + 7 = 15$).

Exemplul 2

`solve(14, 15, [5, 5, 6, 6])`
În acest exemplu avem patru molecule cu greutățile 5, 5, 6 și 6, și căutăm un subset cu greutatea totală între 14 și 15, inclusiv. La fel, vom remarca $15 - 14 \geq 6 - 5$. Nu există un subset de molecule cu greutate totală între 14 și 15 astfel funcția va returna un array vid.

Exemplul 3

`solve(10, 20, [15, 17, 16, 18])`
În acest exemplu avem patru molecule cu greutățile 15, 17, 16 și 18, și căutăm un subset cu greutatea totală între 10 și 20, inclusiv. La fel, vom remarca $20 - 10 \geq 18 - 15$. Oricare subset format din exact un element satisface cerințele, astfel că răspunsuri corecte sunt: `[0]`, `[1]`, `[2]` și `[3]`.

Subtaskuri

1. (9 puncte): $1 \leq n \leq 100$, $1 \leq w_i \leq 100$, $1 \leq u, l \leq 1000$, toate w_i sunt egale.
2. (10 puncte): $1 \leq n \leq 100$, $1 \leq w_i, u, l \leq 1000$, și $\max(w_0, \ldots, w_{n-1}) - \min(w_0, \ldots, w_{n-1}) \leq 1$ $\max(w_0, \ldots, w_{n-1}) - \min(w_0, \ldots, w_{n-1}) \leq 1$.
3. (12 puncte): $n \leq 100$ și $w_i, u, l \leq 1000$.
4. (15 puncte): $n \leq 10\,000$ și $w_i, u, l \leq 10\,000$.
5. (23 puncte): $n \leq 10\,000$ și $w_i, u, l \leq 500\,000$
6. (31 puncte): $n \leq 200\,000$ și $w_i, u, l < 2^{31}$.

Sample grader

Sample grader-ul citește inputul în următorul format:

- linia 1: numerele întregi n, l, u.
- linia 2: n numere întregi: $w_0, \ldots, w_{n-1}$.