Erkennung von Molekülen

Peter arbeitet für eine Firma, die Geräte zur Identifikation von Molekülen baut. Jedes Molekül hat eine positive ganzzahlige Masse. Das Gerät hat einen *Erkennungsbereich* $[l, u]$, wobei l und u positive ganze Zahlen sind. Das Gerät kann eine Menge von Molekülen genau dann identifizieren, wenn diese Menge eine Teilmenge von Molekülen enthält, deren Gesamtmasse innerhalb des Erkennungsbereichs des Geräts liegt. Formal: Betrachte n Moleküle mit positiven ganzen Massen $w_0, \ldots, w_{n-1}$. Die Erkennung ist erfolgreich, wenn es paarweise verschiedene Indizes $i_1, \ldots, i_m$ gibt, sodass $l \leq w_{i_1} + \ldots + w_{i_m} \leq u$.

Aufgrund der Spezifikationen des Geräts ist garantiert, dass die Lücke zwischen l und u grösser oder gleich der Massendifferenz zwischen dem schwersten und dem leichtesten Molekül ist. Formal: $u - l \geq w_{max} - w_{min}$, wobei $w_{max} = \max(w_0, \ldots, w_{n-1})$ und $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Schreibe ein Programm, welches entweder eine Teilmenge von Molekülen mit einer Gesamtmasse innerhalb des Erkennungsbereichs findet oder feststellt, dass es keine solche Teilmenge gibt.

Implementierungsdetails

Implementiere eine Funktion (Methode):

- `int[] solve(int l, int u, int[] w)`
  - `l` und `u`: die Endpunkte des Erkennungsbereichs,
  - `w`: Masse der Moleküle.
  - wenn die verlangte Teilmenge existiert, soll die Funktion ein Array von paarweise verschiedenen Indizes zurückgeben, welches irgendeine solche Teilmenge angibt. Wenn es mehrere richtige Antworten gibt soll die Funktion irgendeine davon ausgeben.
  - wenn die geforderte Teilmenge nicht existiert, soll die Funktion ein leeres Array zurückgeben.

Für die Sprache C ist die Funktionssignatur etwas anders:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: die Anzahl der Elemente in `w` (z.B., die Anzahl der Moleküle),
  - die anderen Parameter sind die gleichen wie oben.
  - anstatt ein Array von m Indizes zurückzugeben (wie oben), soll die Funktion die Indizes in die ersten m Zellen des Arrays `result` schreiben

und dann m zurückgeben.
- wenn die geforderte Teilmenge nicht existiert, soll die Funktion nichts in das Array `result` schreiben und 0 zurückgeben.

Dein Programm kann die Indizes in das zurückgegebene Array in beliebiger Reihenfolge schreiben (bzw. in das `result` Array in C).
Die zur Verfügung gestellten Vorlagen zeigen dir die Details in deiner Programmiersprache.

Beispiele

Beispiel 1

`solve(15, 17, [6, 8, 8, 7])`
In diesem Beispiel haben wir vier Moleküle mit den Massen 6, 8, 8 und 7. Das Gerät kann Teilmengen von Molekülen mit Gesamtmasse zwischen 15 und 17 (jeweils inklusive) erkennen. Beachte, dass $17 - 15 \geq 8 - 6$. Die Gesamtmasse der Moleküle 1 und 3 ist $w_1 + w_3 = 8 + 7 = 15$, sodass die Funktion `[1, 3]` zurückgeben kann. Andere mögliche richtige Antworten wären `[1, 2]` ($w_1 + w_2 = 8 + 8 = 16$) und `[2, 3]` ($w_2 + w_3 = 8 + 7 = 15$).

Beispiel 2

`solve(14, 15, [5, 5, 6, 6])`
in diesem Beispiel haben wir vier Moleküle mit Massen 5, 5, 6 und 6, und wir suchen eine Teilmenge mit einer Gesamtmasse zwischen 14 und 15 (inklusive). Beachte wieder, dass $15 - 14 \geq 6 - 5$. Es gibt keine Teilmenge von Molekülen mit Gesamtmasse zwischen 14 und 15 sodass die Funktion ein leeres Array zurückgeben muss.

Beispiel 3

`solve(10, 20, [15, 17, 16, 18])`
In diesem Beispiel haben wir 4 Moleküle mit Massen 15, 17, 16 und 18. Wir suchen eine Teilmenge von ihnen mit Gesamtmasse zwischen 10 und 20, inklusive. Beachte wieder, dass $20 - 10 \geq 18 - 15$. Jede Teilmenge, die aus genau einem Element besteht, hat eine Gesamtmasse zwischen 10 und 20, sodass die möglichen richtigen Antworten sind: `[0]`, `[1]`, `[2]` und `[3]`.

Subtasks

1. (9 Punkte): $1 \leq n \leq 100$, $1 \leq w_i \leq 100$, $1 \leq u, l \leq 1000$, alle w_i sind gleich.
2. (10 Punkte): $1 \leq n \leq 100$, $1 \leq w_i, u, l \leq 1000$, und $\max(w_0, \ldots, w_{n-1}) - \min(w_0, \ldots, w_{n-1}) \leq 1$.
3. (12 Punkte): $1 \leq n \leq 100$ und $1 \leq w_i, u, l \leq 1000$.
4. (15 Punkte): $1 \leq n \leq 10\,000$ und $1 \leq w_i, u, l \leq 10\,000$.

5. (23 Punkte): $1 \leq n \leq 10\,000$ und $1 \leq w_i, u, l \leq 500\,000$.
6. (31 Punkte): $1 \leq n \leq 200\,000$ und $1 \leq w_i, u, l < 2^{31}$.

Beispielgrader

Der Beispielgrader liest die Eingaben im folgenden Format:

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