Phát hiện các phân tử (Detecting Molecules)

Petr làm việc cho một công ty nơi đã tạo được một chiếc máy để phát hiện các phân tử. Mỗi phân tử có trọng lượng là một số nguyên dương. Chiếc máy có *khoảng phát hiện* $[l, u]$, trong đó l và u là các số nguyên dương. Chiếc máy có thể phát hiện một tập các phân tử khi và chỉ khi tập này chứa một tập con các phân tử với tổng trọng lượng nằm trong khoảng phát hiện của chiếc máy.

Một cách hình thức, xét n phân tử với trọng lượng $w_0, \ldots, w_{n-1}$. Việc phát hiện là thành công nếu tồn tại một tập con các chỉ số phân biệt $I = \{i_1, \ldots, i_m\}$ sao cho $l \leq w_{i_1} + \ldots + w_{i_m} \leq u$.

Do các đặc thù của chiếc máy, khoảng cách biệt giữa l và u được đảm bảo lớn hơn hoặc bằng khoảng cách biệt trọng lượng giữa phân tử nặng nhất và phân tử nhẹ nhất. Một cách hình thức, $u - l \geq w_{max} - w_{min}$, trong đó $w_{max} = \max(w_0, \ldots, w_{n-1})$ và $w_{min} = \min(w_0, \ldots, w_{n-1})$.

Nhiệm vụ của bạn là viết một chương trình để tìm một tập con bất kỳ của các phân tử có tổng trọng lượng nằm trong khoảng phát hiện, hoặc xác định rằng không có tập con nào như vậy.

Chi tiết cài đặt

Bạn cần cài đặt một hàm (thủ tục):

- `int[] solve(int l, int u, int[] w)`
  - `l` và `u`: các điểm đầu mút của khoảng phát hiện,
  - `w`: trọng lượng của các phân tử.
  - Nếu tập con cần tìm tồn tại, hàm cần trả về một mảng chứa chỉ số của các phân tử mà tạo ra một tập con bất kỳ đó. Nếu có nhiều lời giải đúng, hãy trả về một lời giải bất kỳ.
  - Nếu tập con cần tìm không tồn tại, hàm cần trả về một mảng rỗng.

Với ngôn ngữ lập trình C, khuôn mẫu của hàm hơi khác một chút:

- `int solve(int l, int u, int[] w, int n, int[] result)`
  - `n`: số phần tử của `w` (là số các phân tử),
  - các tham số khác giống ở trên.
  - Thay vì trả về một mảng gồm m chỉ số (như trên), hàm cần ghi các chỉ số vào m ô đầu tiên của mảng `result` và trả về m.
  - Nếu tập con cần tìm không tồn tại, hàm không được ghi gì vào mảng `result` và cần trả về 0.

Chương trình của bạn có thể ghi các chỉ số vào mảng trả về (hoặc mảng result trong C) theo bất kỳ thứ tự nào.

Hãy sử dụng các file mẫu cho trước trong phần cài đặt theo ngôn ngữ bạn lựa chọn.

Các ví dụ

Ví dụ 1

`solve(15, 17, [6, 8, 8, 7])`
Trong ví dụ này, chúng ta có bốn phân tử với trọng lượng 6, 8, 8 và 7. Chiếc máy có thể phát hiện các tập con phân tử với tổng trọng lượng nằm giữa 15 và 17 kể cả hai đầu mút. Lưu ý rằng $17 - 15 \geq 8 - 6$. Tổng trọng lượng của phân tử 1 và 3 là $w_1 + w_3 = 8 + 7 = 15$, nên hàm có thể trả về `[1, 3]`. Các câu trả lời đúng khác là `[1, 2]` ($w_1 + w_2 = 8 + 8 = 16$) và `[2, 3]` ($w_2 + w_3 = 8 + 7 = 15$).

Ví dụ 2

`solve(14, 15, [5, 5, 6, 6])`
Trong ví dụ này, chúng ta có bốn phân tử với trọng lượng 5, 5, 6 và 6, và chúng ta cần tìm một tập con của chúng với tổng trọng lượng nằm giữa 14 và 15 kể cả hai đầu mút. Lưu ý lại là $15 - 14 \geq 6 - 5$.

Không có tập con phân tử nào với tổng trọng lượng nằm giữa 14 và 15 nên hàm cần trả về một mảng rỗng.

Ví dụ 3

`solve(10, 20, [15, 17, 16, 18])`
Trong ví dụ này, chúng ta có bốn phân tử với trọng lượng 15, 17, 16 và 18, và chúng ta cần tìm một tập con của chúng với tổng trọng lượng nằm giữa 10 và 20 kể cả hai đầu mút. Lưu ý lại là $20 - 10 \geq 18 - 15$. Bất kỳ tập con nào chứa đúng một phần tử đều có tổng trọng lượng nằm giữa 10 và 20, nên các câu trả lời đúng là: `[0]`, `[1]`, `[2]` và `[3]`.

Subtasks

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

Chương trình chấm mẫu (Sample grader)

Chương trình chấm mẫu đọc dữ liệu đầu vào theo định dạng sau:

- dòng 1: các số tự nhiên n , l , u .
- dòng 2: n số tự nhiên: $w_0, \ldots, w_{n-1}$.