Gap

Cho N số nguyên không âm $a_1, a_2, \ldots, a_N$ thỏa mãn bất đẳng thức sau $0 \leq a_1 < a_2 < \cdots < a_N \leq 10^{18}$. Jeehak muốn biết giá trị *lớn nhất có thể* của $a_{i+1} - a_i$ với i từ 1 đến $N-1$. Các số nguyên đầu vào sẽ không được cung cấp trực tiếp cho chương trình của Jeehak nhưng có thể truy cập thông qua một hàm đặc biệt. Xem chi tiết trong phần Implementation theo ngôn ngữ lập trình mà bạn lựa chọn.

Task

Hãy giúp Jeehak xây dựng một hàm trả về giá trị lớn nhất có thể của $a_{i+1} - a_i$ với i chạy từ 1 đến $N-1$.

Implementation cho C và C++

Bạn cần xây dựng một hàm `findGap(T, N)` nhận tham số dưới đây và trả về một số nguyên kiểu `long long`:

- `T` — Số hiệu subtask (1 hoặc 2)
- `N` — Số lượng các số nguyên đã cho

Hàm `findGap` có thể gọi hàm `MinMax(s, t, &mn, &mx)` trong đó hai tham số đầu tiên `s` và `t` là các số nguyên kiểu `long long`, còn hai tham số `&mn` và `&mx` là con trỏ biến nguyên kiểu `long long`, nghĩa là, `mn` và `mx` là biến nguyên kiểu `long long`. Khi hàm `MinMax(s, t, &mn, &mx)` trả về, biến `mn` sẽ chứa giá trị nhỏ nhất trong các số a_i lớn hơn hoặc bằng giá trị `s` và biến `mx` sẽ chứa giá trị lớn nhất trong các số a_j nhỏ hơn hoặc bằng giá trị `t`. Trong trường hợp không có số nguyên đầu vào nằm giữa `s` và `t` (kể cả 2 đầu mút), thì cả `mn` và `mx` sẽ chứa giá trị `-1`. Giá trị `s` không nên lớn hơn giá trị `t` khi gọi hàm `MinMax`. Nếu điều kiện này không được thỏa mãn, chương trình sẽ bị kết thúc với một mã thoát khác 0.

Implementation cho Pascal

Bạn cần xây dựng hàm `findGap(T, N)` nhận tham số sau đây và trả về là một số nguyên kiểu `Int64`:

- `T` — Số hiệu subtask (1 hoặc 2) (kiểu `Integer`)
- `N` — Số lượng các số nguyên đã cho (kiểu `LongInt`)

Hàm `findGap` có thể gọi thủ tục `MinMax(s, t, mn, mx)` trong đó hai tham số đầu tiên `s` và `t` là hai số nguyên kiểu `Int64`, còn hai tham số `mn` và `mx` là các biến **gọi theo tham biến** kiểu `Int64`, nghĩa là, `mn` và `mx` là các biến nguyên kiểu `Int64`. Khi thủ tục `MinMax(s, t, mn, mx)` thoát, biến `mn` sẽ chứa giá trị nhỏ nhất trong các số a_i lớn hơn hoặc bằng giá trị `s` và biến `mx` sẽ chứa giá trị lớn nhất trong các số a_j nhỏ hơn hoặc bằng giá trị `t`. Trong trường hợp không có số nguyên đầu vào nằm giữa `s` và `t` (kể cả 2 đầu mút), thì cả `mn` và `mx` sẽ chứa giá trị `-1`. Giá trị `s` không nên lớn hơn giá trị `t` khi gọi thủ tục `MinMax`. Nếu điều kiện này không được thỏa mãn,

chương trình sẽ kết thúc.

Implementation chung

Ngoài các yêu cầu tiêu chuẩn (giới hạn về thời gian và bộ nhớ, chạy không lỗi,...), bài nộp của bạn phải thực hiện được các yêu cầu sau đây để giải quyết một trường hợp thử nghiệm:

- hàm `findGap` phải trả về kết quả đúng,
- chi phí M của các lần gọi hàm `MinMax` không được vượt quá giới hạn cho phép (xem phần Scoring).

Ví dụ cho C, C++

Xét trường hợp $N = 4$ và $a_1 = 2, a_2 = 3, a_3 = 6$, và $a_4 = 8$.

Kết quả, ở đây là 3, có thể được tính toán và trả về bởi `findGap` nếu các lần gọi `MinMax` sau đây được thực hiện:

- `MinMax(1, 2, &mn, &mx)` được gọi, cả `mn` và `mx` đều chứa giá trị 2.
- `MinMax(3, 7, &mn, &mx)` được gọi, `mn` chứa giá trị 3 và `mx` chứa giá trị 6.
- `MinMax(8, 9, &mn, &mx)` được gọi, cả `mn` và `mx` đều chứa giá trị 8.

Ví dụ cho Pascal

Xét trường hợp $N = 4$ và $a_1 = 2, a_2 = 3, a_3 = 6$, và $a_4 = 8$.

Kết quả, ở đây là 3, có thể được tính toán và trả về bởi `findGap` nếu các lần gọi `MinMax` sau đây được thực hiện:

- `MinMax(1, 2, mn, mx)` được gọi, cả `mn` và `mx` đều chứa giá trị 2.
- `MinMax(3, 7, mn, mx)` được gọi, `mn` chứa giá trị 3 và `mx` chứa giá trị 6.
- `MinMax(8, 9, mn, mx)` được gọi, cả `mn` và `mx` đều chứa giá trị 8.

Scoring

Tất cả các subtask thỏa mãn $2 \leq N \leq 100,000$.

Subtask 1 (30 điểm): Mỗi lần gọi `MinMax` sẽ cộng 1 vào M. Bạn sẽ nhận được toàn bộ điểm cho subtask này nếu $M \leq \frac{N+1}{2}$ với tất cả các test.

Subtask 2 (70 điểm): Gọi k là số lượng các số nguyên đầu vào lớn hơn hoặc bằng `s` và nhỏ hơn hoặc bằng `t` trong một lần gọi `MinMax`. Mỗi lần gọi `MinMax` sẽ cộng $k + 1$ vào M. Điểm cuối cùng sẽ được tính theo quy tắc sau: Điểm cuối cùng cho subtask là điểm nhỏ nhất bạn nhận được trong tất cả các test. Với mỗi test, điểm là 70 nếu $M \leq 3N$ và điểm là $\frac{60}{\sqrt{\frac{M}{N}+1}-1}$ trong trường hợp trái lại.

Experimentation

Chương trình grader mẫu mà bạn có thể tải về từ hệ thống chấm thi, sẽ đọc dữ liệu từ thiết bị vào chuẩn. Dòng đầu tiên của dữ liệu vào chứa hai số nguyên, số hiệu subtask T, và N. Dòng tiếp theo chứa N số nguyên theo thứ tự tăng dần. Chương trình grader mẫu sẽ ghi ra thiết bị ra chuẩn giá trị trả về bởi `findGap` trong dòng đầu tiên và giá trị của M tương ứng với subtask và dữ liệu vào.

Đầu vào sau đây mô tả ví dụ trên:

```
2 4
2 3 6 8
```