

LIBRARY CHECKER

Tác giả: Phan Thành Hưng

January 28, 2026

Tài liệu gồm 7 bài, có 8 trang.

1 SEGTree - Segment Tree (Basic)

Cho dãy số nguyên a_1, a_2, \dots, a_n . Xử lý q truy vấn thuộc một trong các dạng sau:

- $\boxed{1 \ k \ x}$: Gán phần tử thứ k thành x .
- $\boxed{2 \ l \ r}$: Tính tổng các phần tử trong đoạn $[l, r]$.
- $\boxed{3 \ l \ r}$: Tìm phần tử có giá trị nhỏ nhất trong đoạn $[l, r]$.
- $\boxed{4 \ l \ r}$: Tìm phần tử có giá trị lớn nhất trong đoạn $[l, r]$.

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 10^5$).
- Dòng thứ hai gồm n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$).
- q dòng tiếp theo, mỗi dòng chứa một truy vấn.

Dữ liệu ra.

- Với mỗi truy vấn loại 2, 3 hoặc 4, in ra một dòng chứa kết quả tương ứng.

Ví dụ 1

Input

```
8 10
5 -1 3 7 -4 2 6 -8
2 1 8
3 2 5
4 4 7
1 5 10
2 3 6
1 8 4
2 6 8
3 1 4
1 2 -9
4 1 3
```

Output

```
10
-4
7
22
12
-1
5
```

2 SEGPERS - Persistent Segment Tree

Cho dãy số nguyên a_1, a_2, \dots, a_n . Ban đầu, tồn tại phiên bản 1 của dãy, chính là dãy ban đầu. Mỗi truy vấn cập nhật sẽ tạo ra một phiên bản mới của dãy.

Xử lý q truy vấn thuộc một trong các dạng sau:

- $1 \ v \ i \ k$: Từ phiên bản v , tạo ra một phiên bản mới bằng cách tăng phần tử thứ i lên k đơn vị ($|k| \leq 10^9$).
- $2 \ v \ l \ r$: Tính tổng các phần tử trong đoạn $[l, r]$ của phiên bản v .
- $3 \ v \ l \ r$: Tìm giá trị nhỏ nhất trong đoạn $[l, r]$ của phiên bản v .
- $4 \ v \ l \ r$: Tìm giá trị lớn nhất trong đoạn $[l, r]$ của phiên bản v .

Các phiên bản được đánh số liên tiếp từ 1 theo thứ tự xuất hiện (các truy vấn loại 1).

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 10^5$).
- Dòng thứ hai gồm n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$).
- q dòng tiếp theo, mỗi dòng chứa một truy vấn như mô tả ở trên.

Dữ liệu ra.

- Với mỗi truy vấn loại 2, 3 hoặc 4, in ra một dòng chứa kết quả tương ứng.

Ví dụ 2

Input

```
6 9
2 -5 4 1 7 -3
2 1 1 6
1 1 2 3
2 2 1 6
3 2 3 6
4 2 1 4
1 2 4 -2
2 3 4 6
3 1 1 6
4 3 1 6
```

Output

```
6
9
-3
4
3
-5
7
```

3 SEGPERSLAZY - Lazy Persistent Segment Tree

Cho dãy số nguyên a_1, a_2, \dots, a_n . Ban đầu, tồn tại phiên bản 1 của dãy, chính là dãy ban đầu. Mỗi truy vấn cập nhật sẽ tạo ra một phiên bản mới của dãy.

Xử lý q truy vấn thuộc một trong các dạng sau:

- $1\ v\ l\ r\ k$: Từ phiên bản v , tạo ra một phiên bản mới bằng cách tăng mỗi phần tử trong đoạn $[l, r]$ lên k đơn vị.
- $2\ v\ l\ r$: Tính tổng các phần tử trong đoạn $[l, r]$ của phiên bản v .
- $3\ v\ l\ r$: Tìm giá trị nhỏ nhất trong đoạn $[l, r]$ của phiên bản v .
- $4\ v\ l\ r$: Tìm giá trị lớn nhất trong đoạn $[l, r]$ của phiên bản v .

Các phiên bản được đánh số liên tiếp từ 1 theo thứ tự xuất hiện (các truy vấn loại 1).

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 10^5$).
- Dòng thứ hai gồm n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$).
- q dòng tiếp theo, mỗi dòng chứa một truy vấn như mô tả ở trên.

Dữ liệu ra.

- Với mỗi truy vấn loại 2, 3 hoặc 4, in ra một dòng chứa kết quả tương ứng.

Ví dụ 2

Input

```
6 9
2 -5 4 1 7 -3
2 1 1 6
1 1 2 5 3
2 2 1 6
3 2 3 6
4 2 1 4
1 2 4 6 -2
2 3 4 6
3 1 1 6
4 3 1 6
```

Output

```
6
18
-3
7
-3
-5
7
```

4 WAVELET - Wavelet Tree

Cho dãy số nguyên a_1, a_2, \dots, a_n . Hãy xử lý q truy vấn thuộc một trong các dạng sau:

- $\boxed{1 \ l \ r \ k}$: Phần tử lớn thứ k trong đoạn $[l, r]$. Nếu không tồn tại, in ra NO.
- $\boxed{2 \ l \ r \ x}$: Số phần tử $\leq x$ trong đoạn $[l, r]$.
- $\boxed{3 \ l \ r \ x}$: Tổng các phần tử $\leq x$ trong đoạn $[l, r]$.
- $\boxed{4 \ l \ r \ x}$: Số phần tử $= x$ trong đoạn $[l, r]$.

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 10^5$).
- Dòng thứ hai gồm n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^6$).
- q dòng tiếp theo, mỗi dòng chứa một truy vấn.

Dữ liệu ra.

- Gồm q dòng, mỗi dòng trả lời kết quả của truy vấn tương ứng.

Ví dụ 1

Input

```
6 8
-5 3 -2 7 -2 4
1 1 6 3
2 1 6 -2
3 1 6 0
4 1 6 -2
1 2 5 5
2 2 5 10
3 2 5 3
4 3 6 7
```

Output

```
3
3
-9
2
NO
4
-1
1
```

5 WAVELET2 - Wavelet Tree 2

Cho một mảng a gồm n số nguyên.

Hãy xử lý q truy vấn, mỗi truy vấn có dạng:

- $\boxed{l\ r\ k}$: Tìm số nguyên x nhỏ nhất sao cho x xuất hiện trong đoạn $[l, r]$ **nhiều hơn** $\frac{r-l+1}{k}$ lần. Nếu không tồn tại số như vậy, in ra -1 .

Dữ liệu vào.

- Dòng đầu gồm hai số nguyên n, q ($1 \leq n, q \leq 3 \times 10^5$).
- Dòng thứ hai gồm n số nguyên a_1, a_2, \dots, a_n ($1 \leq a_i \leq n$).
- q dòng tiếp theo, mỗi dòng gồm ba số nguyên l, r, k ($1 \leq l \leq r \leq n, 2 \leq k \leq 5$).

Dữ liệu ra.

- Với mỗi truy vấn, in ra một dòng chứa đáp án tương ứng.

Ví dụ 1

Input

```
4 2
1 1 2 2
1 3 2
1 4 2
```

Output

```
1
-1
```

Ví dụ 2

Input

```
5 3
1 2 1 3 2
2 5 3
1 2 3
5 5 2
```

Output

```
2
1
2
```

6 DSUROLLBACK - DSU Rollback (Basic)

Cho đồ thị vô hướng gồm n đỉnh. Các đỉnh được đánh số từ 1 tới n . Ban đầu, đồ thị chưa có cạnh nào và các đỉnh $1, 2, \dots, n$ lần lượt mang giá trị a_1, a_2, \dots, a_n .

Hãy xử lý q truy vấn thuộc một trong các dạng sau:

- **0 u v** : Thêm một cạnh mới nối giữa hai đỉnh u và v .
- **1**: Xóa cạnh gần nhất đã được thêm vào từ một thao tác loại 0 mà chưa bị xóa.
- **2**: Đếm số thành phần liên thông trong đồ thị.
- **3 u** : Đếm số đỉnh thuộc cùng thành phần liên thông với đỉnh u (bao gồm cả u).
- **4 u** : Tìm giá trị đỉnh nhỏ nhất trong các đỉnh thuộc cùng thành phần liên thông với u .

Lưu ý: Giữa hai đỉnh có thể có nhiều cạnh nối phân biệt.

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 10^5$).
- Dòng tiếp theo gồm n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$).
- q dòng tiếp theo, mỗi dòng chứa một truy vấn.

Dữ liệu đảm bảo tại mọi thời điểm, số truy vấn loại 0 xuất hiện không ít hơn số truy vấn loại 1.

Dữ liệu ra.

- Với mỗi truy vấn loại 2, 3 hoặc 4, in ra một dòng chứa kết quả tương ứng.

Ví dụ 1

Input

```
5 8
3 -1 4 2 0
2
0 1 2
4 1
0 2 3
3 1
1
2
4 3
```

Output

```
5
-1
3
4
4
```

7 VIRTREE - Virtual Tree (Basic)

Cho cây vô hướng gồm n đỉnh có gốc tại đỉnh 1.

Hãy xử lý q truy vấn. Mỗi truy vấn có dạng:

- $\boxed{k \ v_1 \ v_2 \ \dots \ v_k}$: Cho k đỉnh phân biệt trên cây. Xây dựng **Virtual Tree** chứa tất cả các đỉnh v_i và các LCA cần thiết.

Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương n, q ($1 \leq n, q \leq 2 \times 10^5$).
- $n - 1$ dòng tiếp theo, mỗi dòng gồm hai số u, v mô tả một cạnh của cây.
- q dòng tiếp theo, mỗi dòng chứa một truy vấn.

Dữ liệu đảm bảo: $\sum k \leq 2 \times 10^5$.

Dữ liệu ra.

- Với mỗi truy vấn, in ra:
 - + Dòng đầu: một số d là số đỉnh của Virtual Tree tương ứng.
 - + $d - 1$ dòng tiếp theo: mỗi dòng chứa một cạnh $u \ v$ trên Virtual Tree đó.

Bạn được phép in các cạnh theo thứ tự tùy ý.

Ví dụ 1

Input

```
7 2
1 2
1 3
2 4
2 5
3 6
3 7
3 4 5 6
2 4 7
```

Output

```
5
2 4
2 5
1 2
1 6
3
1 4
1 7
```

*** HẾT ***
