

# LIBRARY CHECKER

Tác giả: Phan Thành Hưng

February 8, 2026

*Tài liệu gồm 11 bài, có 12 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  $x$  ( $|x| \leq 10^9$ ).
- $\boxed{2 \ l \ r}$ : 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 SEGLAZY - Segment Tree (Lazy propagation)

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

- $\boxed{1 \ l \ r \ k}$ : Tăng mỗi phần tử trong đoạn  $[l, r]$  lên  $k$  đơn vị ( $|k| \leq 10^9$ ).
- $\boxed{2 \ l \ r}$ : 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

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

##### Output

```
3
-3
5
12
1
8
0
```

### 3 QUERY2D - 2D Data Structure (Basic)

Cho lưới ô vuông kích thước  $n \times m$ . Ô nằm tại hàng  $i$ , cột  $j$  có giá trị là  $a_{i,j}$ .

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

- $\boxed{1 \ x \ y \ k}$ : Tăng ô có tọa độ  $(x, y)$  lên  $k$  đơn vị ( $|k| \leq 10^9$ ).
- $\boxed{2 \ x_1 \ y_1 \ x_2 \ y_2}$ : Tính tổng giá trị các ô vuông trong hình chữ nhật từ  $(x_1, y_1)$  tới  $(x_2, y_2)$ .

**Dữ liệu vào.**

- Dòng đầu tiên gồm hai số nguyên dương  $n, m, q$  ( $1 \leq n, m \leq 10^3, 1 \leq q \leq 10^5$ ).
- Dòng thứ  $i$  trong  $n$  dòng tiếp theo chứa  $m$  số nguyên  $a_{i,1}, a_{i,2}, \dots, a_{i,m}$  ( $|a_{i,j}| \leq 10^9$ ).
- $q$  dòng tiếp theo, mỗi dòng chứa một truy vấn.

**Dữ liệu ra.**

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

Ví dụ 1

**Input**

```
4 4 7
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2 1 1 4 4
1 2 2 5
2 2 2 3 3
1 4 4 -10
2 3 3 4 4
1 1 1 3
2 1 1 2 2
```

**Output**

```
136
39
44
22
```

#### 4 SPARSEBIT2D - Fenwick Tree 2D (Compressed)

Cho một lưới ô vuông vô hạn trên mặt phẳng tọa độ. Ban đầu, giá trị tại mọi ô đều bằng 0.

Bạn cần xử lý  $q$  truy vấn, mỗi truy vấn thuộc một trong hai loại sau:

- $\boxed{1 \ x \ y \ k}$ : Tăng ô có tọa độ  $(x, y)$  lên  $k$  đơn vị ( $1 \leq x, y \leq 10^9, |k| \leq 10^9$ ).
- $\boxed{2 \ x_1 \ y_1 \ x_2 \ y_2}$ : Tính tổng giá trị các ô vuông trong hình chữ nhật từ  $(x_1, y_1)$  tới  $(x_2, y_2)$  ( $1 \leq x_1 \leq x_2 \leq 10^9, 1 \leq y_1 \leq y_2 \leq 10^9$ ).

**Dữ liệu vào.**

- Dòng đầu tiên gồm hai số nguyên dương  $q$  ( $1 \leq q \leq 2 \times 10^5$ ).
- $q$  dòng tiếp theo, mỗi dòng chứa một truy vấn.

**Dữ liệu ra.**

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

Ví dụ 1

**Input**

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

**Output**

```
12
11
16
```

---

## 5 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
```

## 6 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 ( $|k| \leq 10^9$ ).
- $2\ v\ l\ r$ : 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 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
```

## 7 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
```



## 8 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
```

---

## 9 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
```

## 10 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:

- $[k\ v_1\ v_2\ \dots\ v_k]$ : 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:  $\sum k \leq 5 \times 10^5$ .

**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
```

## 11 DPOPT - DP Optimization (D&C / CHT)

Cho dãy  $n$  phần tử  $a_1, a_2, \dots, a_n$ . Ta định nghĩa giá trị  $C(l, r)$  của đoạn con liên tiếp từ  $l$  tới  $r$  là bình phương tổng các phần tử trong đoạn đó, cụ thể là  $C(l, r) = (a_l + a_{l+1} + \dots + a_r)^2$ .

Hãy chia dãy thành đúng  $k$  đoạn con liên tiếp sao cho mỗi phần tử thuộc đúng một đoạn con và tổng giá trị tất cả các đoạn đó là nhỏ nhất.

### Dữ liệu vào.

- Dòng đầu tiên gồm hai số nguyên dương  $n, k$  ( $1 \leq n \leq 10^4, 1 \leq k \leq \min(100, n)$ ).
- Dòng tiếp theo gồm  $n$  số nguyên  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^3$ ).

### Dữ liệu ra.

- Gồm một dòng duy nhất chứa tổng giá trị của các đoạn con nhỏ nhất tìm được.

#### Ví dụ 1

**Input**

5 3  
1 1 1 1 1

**Output**

9

**Giải thích**

Ta có thể chia dãy như sau:  $\{1, 1 \mid 1, 1 \mid 1\}$ .

Tổng giá trị là  $2^2 + 2^2 + 1 = 9$ , nhỏ nhất tìm được.

#### Ví dụ 2

**Input**

5 3  
8 9 5 9 7

**Output**

516

**Giải thích**

Ta có thể chia dãy như sau:  $\{8 \mid 9, 5 \mid 9, 7\}$ .

\*\*\* HẾT \*\*\*

---