Segment-Tree Drill Session

$N, Q \le 10^5$
$A_i \le 10^9$

Q: Arr : [2 3 2 5 7 2 2 9 5]
$\quad\quad\quad\quad$ 0 1 2 3 4 5 6 7 8

In there are 'Q' queries from [L, R] → give majority element [L, R]

Ex: [L=2, R=6] → 2

$(freq) > \left(\frac{R-L+1}{2}\right)$

Majority concept using Boyer-Moore Algo :-

1. We keep a candidate and a counter
2. start with the first element as the candidate and cnt = 1.
3. Traverse the array :
   - If current element == maj → cnt++
   - Else → decrement cnt
   - If cnt become 0 → change candidate to current element & reset cnt = 1
4. Finally return maj.

```
public:
    int majorityElement(vector<int>& nums) {
        int cnt = 1;
        int maj = nums[0];
        for(int i=1;i<nums.size();i++){
            if(nums[i]==maj){
                cnt++;
            }else if(cnt>0){
```

```
int maj = nums[0];
for(int i=1;i<nums.size();i++){
    if(nums[i]==maj){
        cnt++;
    }else if(cnt>0){
        cnt--;
    }else{
```

```
        }else if(cnt>0){
            cnt--;
        }else{
            maj = nums[i];
            cnt = 1;
        }
    }
    return maj;
}
```

```cpp
#define F first
#define S second

int n;
int arr[100100];
pair<int,int> t[400400];

pair<int,int> combine(pair<int,int> a,pair<int,int> b){
    if(a.S<b.S)swap(a,b);
    if(a.F==b.F)return {a.F,a.S+b.S};
    else return {a.F, a.S-b.S};
}

void build(int id,int l,int r){
    if(l==r){
        t[id] = {arr[l],1};
        return;
    }
    int mid = (l+r)/2;
    build(2*id,l,mid);
    build(2*id+1,mid+1,r);
    t[id] = combine(t[2*id],t[2*id+1]);
}
```

```cpp
void build(int id,int l,int r){
    if(l==r){
        t[id] = {arr[l],1};
        return;
    }
    int mid = (l+r)/2;
    build(2*id,l,mid);
    build(2*id+1,mid+1,r);
    t[id] = combine(t[2*id],t[2*id+1]);
}

pair<int,int> query(int id,int l,int r,int lq,int rq){
    if(lq>r||l>rq)return {0,0};
    if(lq<=l&&r<=rq) return t[id];
    int mid = (l+r)/2;
    return combine(query(id<<1,l,mid,lq,rq), query(id<<1|1,mid+1,r,lq,rq));
}
```

```
void solve(){
    cin>>n;
    map<int,vector<int>> valpos;
    for(int i=0;i<n;i++){
        cin>>arr[i];
        valpos[arr[i]].push_back(i);
    }
    build(1,0,n-1);
    int q;
    cin>>q;
    while(q--){
        int l,r;
        cin>>l>>r;
        auto ans = query(1,0,n-1,l,r);
        int maj = ans.F;
        // <= r - <l
        int cnt = upper_bound(valpos[maj].begin(),valpos[maj].end(),r) -
                  lower_bound(valpos[maj].begin(),valpos[maj].end(),l);
        if(cnt*2 > (r-l+1)){
            cout<<maj<<endl;
        }else{
            cout<<-1<<endl;
        }
    }
}
```

```cpp
        int maj = arr[random_val(l,r)];
            // <= r - <l
        int cnt = upper_bound(valpos[maj].begin(),valpos[maj].end(),r) -
                    lower_bound(valpos[maj].begin(),valpos[maj].end(),
                    l);
        if(cnt*2 > (r-l+1)){
            found = 1;
            break;
        }
    }
    if(found){
        cout<<maj<<endl;
    }else{
        cout<<-1<<endl;
    }
  }
}
```

```
}

signed main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    srand(time(0));
    int t=1;
    //cin>>t;
    while(t--){
        solve();
    }
```

spoj.com/problems/GSS4/

Pager Templa... Tally - Google She... Youtube FlatGenie AZ Lecture Slides... VG List | AlgoZenit... Figma DSA Course Struct... Meet Discussion -... AZ Process Docs -... Placement Cell Dat

GSS4 - Can you answer these queries IV

#tree

You are given a sequence A of N (N ≤ 100,000) positive integers. Their sum will be less than 10^{18}. On this sequence you have to apply M (M ≤ 100,000) operations:

- (A) For given x, y, for each elements between the x-th and the y-th ones (inclusively, counting from 1), modify it to its positive square root (rounded down to the nearest integer).
- (B) For given x, y, query the sum of all the elements between the x-th and the y-th ones (inclusively, counting from 1) in the sequence.

Input

Multiple test cases, please proceed them one by one. Input terminates by EOF.

For each test case:

The first line contains an integer N. The following line contains N integers, representing the sequence $A_1 .. A_N$. The third line contains an integer M. The next M lines contain the operations in the form "i x y". i = 0 denotes the modify operation, i = 1 denotes the query operation.

Output

Q- GSS4 - Can you answer these queries - IV

→ Each segment tree nodes stores:

- F → sum of the range
- S → count of elements in the range that are greater than 1

→ leaf ~~nodes~~ node stores the array value.

→ If value > 1, set $s = 1$, else $s = 0$.

→ Apply $A[i] = [\sqrt{A[i]}]$ for range $[L, R]$

→ If $s = 0$, stop recursion.

→ otherwise update leaf nodes & recompute sum & s.

```
const lli maxn=100100;
lli n,q,a,b,c;
ii t[4*maxn];
lli arr[maxn];
ii combine(ii a,ii b){
        ii ans;
        ans.F=b.F+a.F;
        ans.S=a.S&b.S;
        return ans;
}
void build(int id,int l,int r){
        if(l==r){
                t[id].F=arr[l];
                if(arr[l]==1)t[id].S=1;
                else t[id].S=0;
                return;
        }
        lli mid=(l+r)>>1;
        build(id<<1,l,mid);
        build(id<<1|1,mid+1,r);
        t[id]=combine(t[id<<1],t[id<<1|1]);
}
void update(int id,int l,int r,int lq,int rq){
        if(l>r||lq>r||l>rq||t[id].S)return;
        if(l==r){
                t[id].F=sqrt(t[id].F);
                if(t[id].F==1)t[id].S=1;
                return;
        }
        lli mid=(l+r)>>1;
        update(id<<1,l,mid,lq,rq);
```

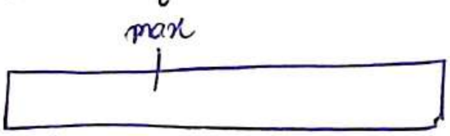
```
}
void update(int id,int l,int r,int lq,int rq){
        if(l>r||lq>r||l>rq||t[id].S)return;
        if(l==r){
                t[id].F=sqrt(t[id].F);
                if(t[id].F==1)t[id].S=1;
                return;
        }
        lli mid=(l+r)>>1;
        update(id<<1,l,mid,lq,rq);
        update(id<<1|1,mid+1,r,lq,rq);
        t[id]=combine(t[id<<1],t[id<<1|1]);
}

lli query(int id,int l,int r,int lq,int rq){
        if(l>r||l>rq||lq>r)return 0;
        if(lq<=l&&r<=rq){
                return t[id].F;
        }
        lli mid=(l+r)>>1;
        return query(id<<1,l,mid,lq,rq)+query(id<<1|1,mid+1,r,lq,rq);
}
```

```
}

void solve(){
        fr(i,n)cin>>arr[i];
        build(1,0,n-1);
        cin>>q;
        fr(i,q){
                cin>>a>>b>>c;
                if(b>c)swap(b,c);
                if(a==0){
                        update(1,0,n-1,b-1,c-1);
                }
                else{
                        cout<<query(1,0,n-1,b-1,c-1)<<endl;
                }
        }


```

Q: Child & Sequence

$a[i] = a[i] \% x$

If max of range $<$ x (L,R), will it change any element ?



max $> x$

↳ let it flow on both sides.

observation of problem :-

$\because y > x \Rightarrow y \% x \leq y/2$

$x > y/2$

$y \% x < y/2$

$x < y/2$

$y \% x < y/2$

$a_i = \sqrt{a_i} \rightarrow 6$

$a_i = a_i \% x \rightarrow \log(a_i)$

$a_i = \phi(a_i) \rightarrow 2\log(a_i)$

$a_i = a_i / p \rightarrow \log(a_i)$

XOR on Segment

a_i = ~~a[i]~~ ~~two~~ requirements :

1) Lazy Merge → Combine the info (sum & count of values) → 1) from L child to right child ~~to~~ form parent
2) Apply ( ) → Apply square - root operation
3) Push down → Before moving to children , ~~propagate~~ propagate the pending update from the ~~children~~ current node to Left & right child.

CODEFORCES
Sponsored by TON

F. Ant colony

time limit per test: 1 second
memory limit per test: 256 megabytes

Mole is hungry again. He found one ant colony, consisting of n ants, ordered in a row. Each ant i ($1 \leq i \leq n$) has a strength s_i.

In order to make his dinner more interesting, Mole organizes a version of «Hunger Games» for the ants. He chooses two numbers l and r ($1 \leq l \leq r \leq n$) and each pair of ants with indices between l and r (inclusively) will fight. When two ants i and j fight, ant i gets one battle point only if s_i divides s_j (also, ant j gets one battle point only if s_j divides s_i).

After all fights have been finished, Mole makes the ranking. An ant i, with v_i battle points obtained, is going to be freed only if $v_i = r - l$, or in other words only if it took a point in every fight it participated. After that, Mole eats the rest of the ants. Note that there can be many ants freed or even none.

In order to choose the best sequence, Mole gives you t segments $[l_i, r_i]$ and asks for each of them how many ants is he going to eat if those ants fight.

Input

The first line contains one integer n ($1 \leq n \leq 10^5$), the size of the ant colony.

The second line contains n integers $s_1, s_2, \ldots, s_n$ ($1 \leq s_i \leq 10^9$), the strengths of the ants.

The third line contains one integer t ($1 \leq t \leq 10^5$), the number of test cases.

Each of the next t lines contains two integers l_i and r_i ($1 \leq l_i \leq r_i \leq n$), describing one query.

Output

Print to the standard output t lines. The i-th line contains number of ants that Mole eats from the segment $[l_i, r_i]$.

D. The Child and Sequence

time limit per test: 4 seconds

memory limit per test: 256 megabytes

At the children's day, the child came to Picks's house, and messed his house up. Picks was angry at him. A lot of important things were lost, in particular the favorite sequence of Picks.

Fortunately, Picks remembers how to repair the sequence. Initially he should create an integer array $a[1], a[2], ..., a[n]$. Then he should perform a sequence of m operations. An operation can be one of the following:

1. Print operation l, r. Picks should write down the value of $\sum_{i=l}^{r} a[i]$.
2. Modulo operation l, r, x. Picks should perform assignment $a[i] = a[i] \ mod \ x$ for each i ($l \le i \le r$).
3. Set operation k, x. Picks should set the value of $a[k]$ to x (in other words perform an assignment $a[k] = x$).

Can you help Picks to perform the whole sequence of operations?

Input

The first line of input contains two integer: n, m ($1 \le n, m \le 10^5$). The second line contains n integers, separated by space: $a[1], a[2], ..., a[n]$ ($1 \le a[i] \le 10^9$) — initial value of array elements.

Each of the next m lines begins with a number $type$ ($type \in \{1, 2, 3\}$).

- If $type = 1$, there will be two integers more in the line: l, r ($1 \le l \le r \le n$), which correspond the operation 1.
- If $type = 2$, there will be three integers more in the line: l, r, x ($1 \le l \le r \le n$; $1 \le x \le 10^9$), which correspond the operation 2.
- If $type = 3$, there will be two integers more in the line: k, x ($1 \le k \le n$; $1 \le x \le 10^9$), which correspond the operation 3.

Output

For each operation 1, please print a line containing the answer. Notice that the answer may exceed the 32-bit integer.