```
8 3
1 3
2 4
6 7
```

1, 2, 3, 4, 6, 7 => 6 distinct toys that he plays with.

cnt[i] => how many days toy i was used.
ans = count of i where cnt[i] > 0.

prefix[i] => a[0] + a[1] ... a[i].

pref[r] += 1 => update +1 to 0...r
pref[l - 1] -= 1 => update -1 to 0..l-1

// Given an array A, answer Q queries
// Query: Range (or/xor/and) of the subarray [l, r].

1. How to get range xor ?
   - range(1..r) and range(1...l-1)
I can get xor of range(l, r) => range(1..r) ^ range(1..l-1)

2. what about or ?
 - is or invertible ?
   - no
- how to solve this now ?

1, 2 , 3, 4

xor(1, 4)
2^0(bit_0) + 2^1(bit_1) + 2^2(bit_2).....

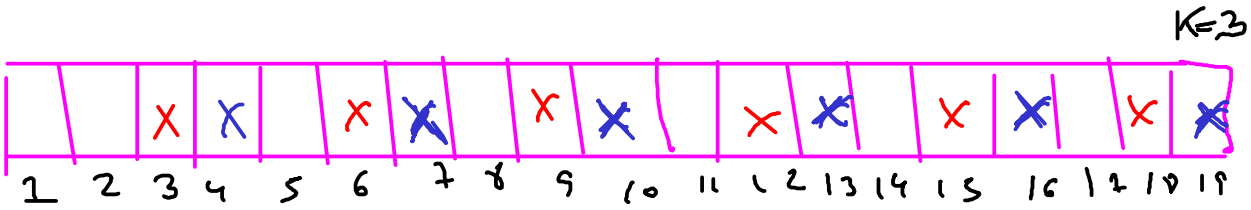
$a_i \le 10^9$ $2^{30} > 10^9$

↳ K (B, t), (l, r)
↳ count of ON bit in this range using prefix sum

OR → will have this bit set
cnt > 0

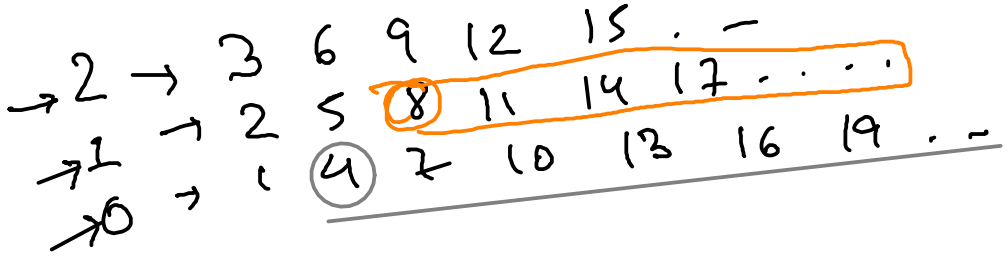
```
// Process Q queries and then print the array
// K is given
// Query(i, c): add c to {i, i + k, i + 2*k ....}
// all queries have common k.
```

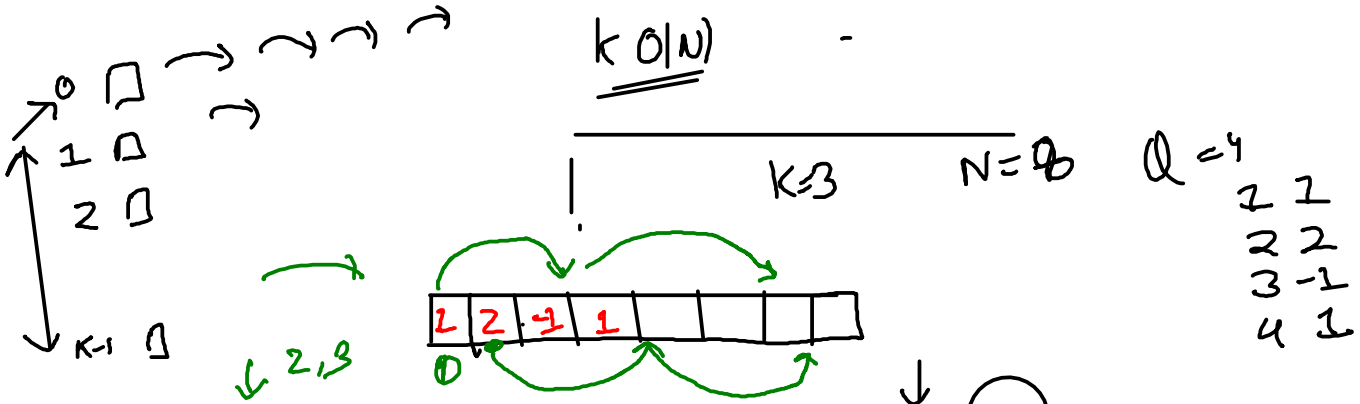
requirement: process query in like O(1) and finally get the answer in O(N).



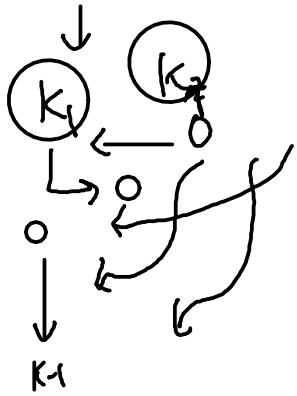
Update 3 11 4 x

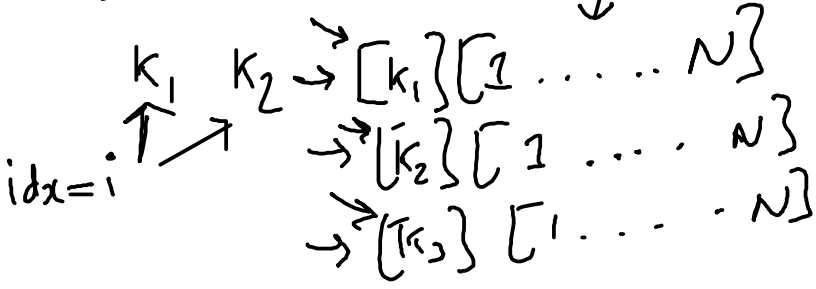
all those indices that have same $(i-1)\%k$ lie in one group.






```
for(int i = 1; i <= n; i++) {
    A[i] = suff[i];
    if(i + k <= n) {
        suff[i + k] += suff[i];
    }
}
```





Queries → Update(i, k, c)

$k_1 \rightarrow$ { 3

$k_2 \rightarrow$ { 3

Complexity

$O(N * \text{Number of } k_s)$

kT →

1. $k > \sqrt{N}$
   ↳ $\sqrt{N}$ indices at once

2. $k < \sqrt{N}$

$\sqrt{N}$ [ ]

update

```
// Process Q queries and then print the array
// query(l, r): add (j - l + 1) to j in [l, r]
// AP type update on range.
```

[l, r] → a[l] += 1

a[l+1] += 2

a[l+2] += 3

a[r] += . . .

(l, r) k

acc[i]

Update → acc[i]

find

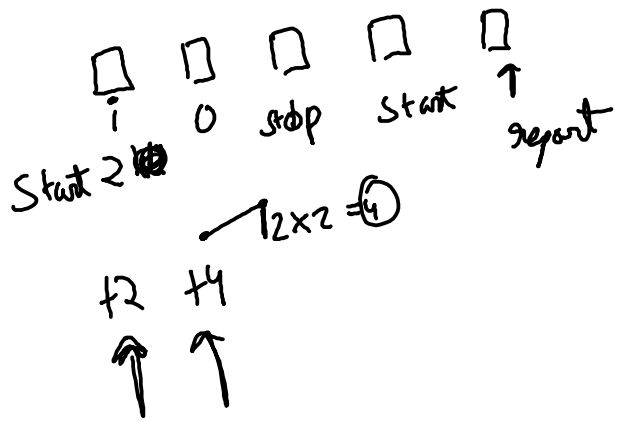
+1 −1

Update(), A[]

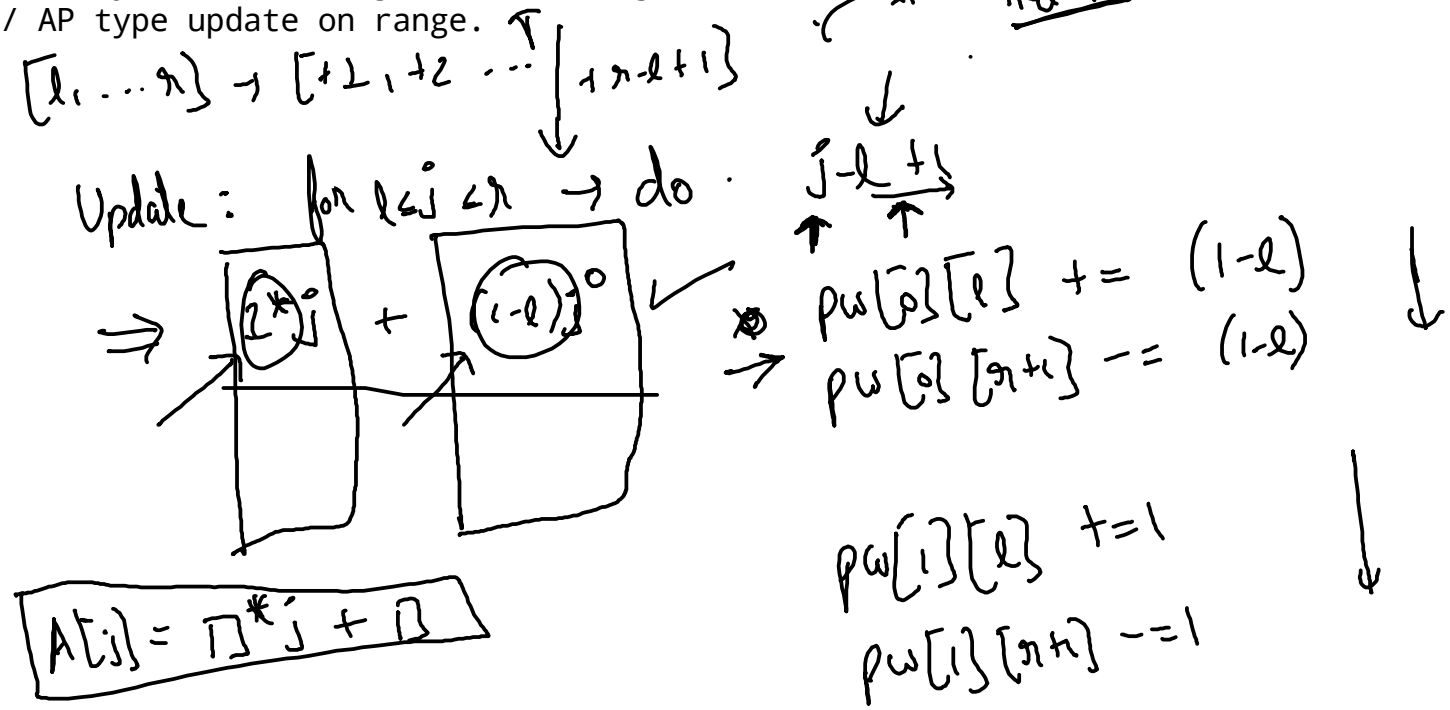
up[l] += 1

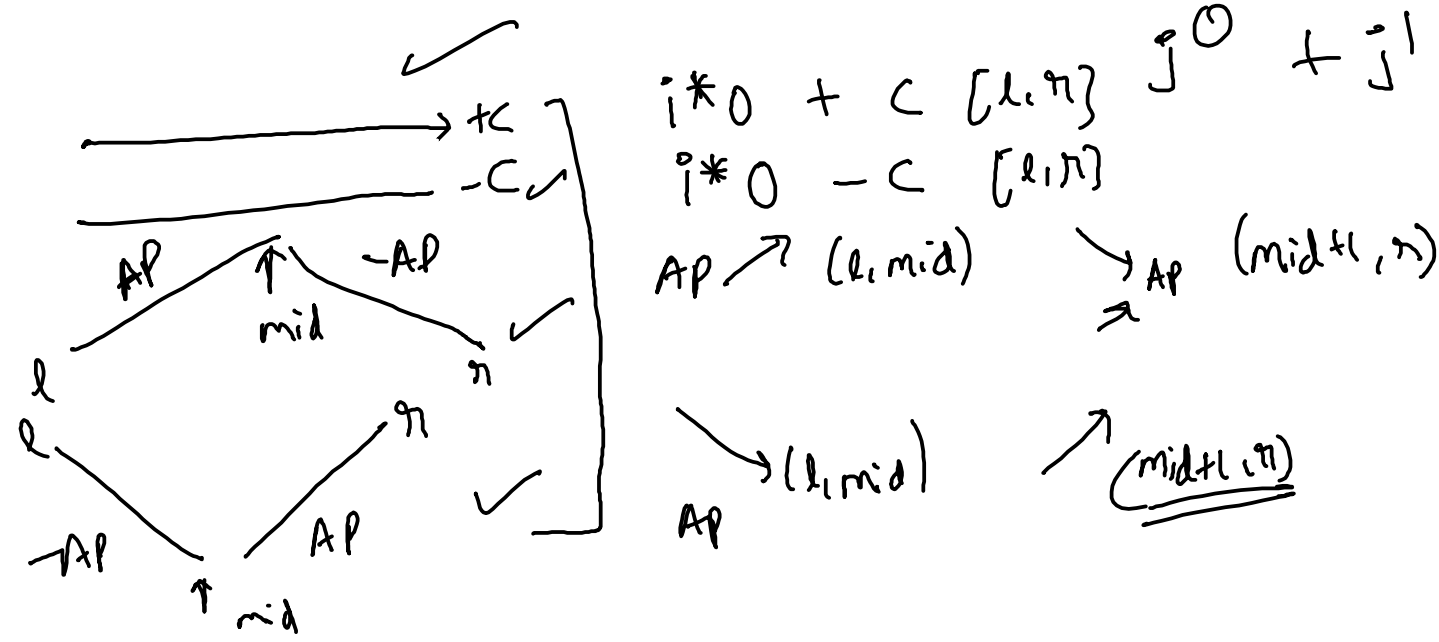
up[r+1] −= 1

(l, r) (r+1) −




```
// Process Q queries and then print the array
// Query(l, r): add (j - l + 1) to j in [l, r]
// AP type update on range.
```





Count the number of pairs of indices (i,j) such that i<j and aj-ai=j-i.

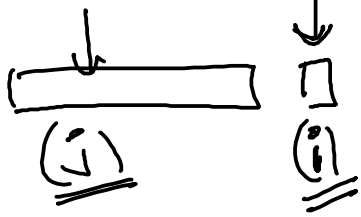
$a_j - a_i = j - i$

$(a_j - j) = (a_i - i)$

ans += freq[val]

val = $a_i - i$

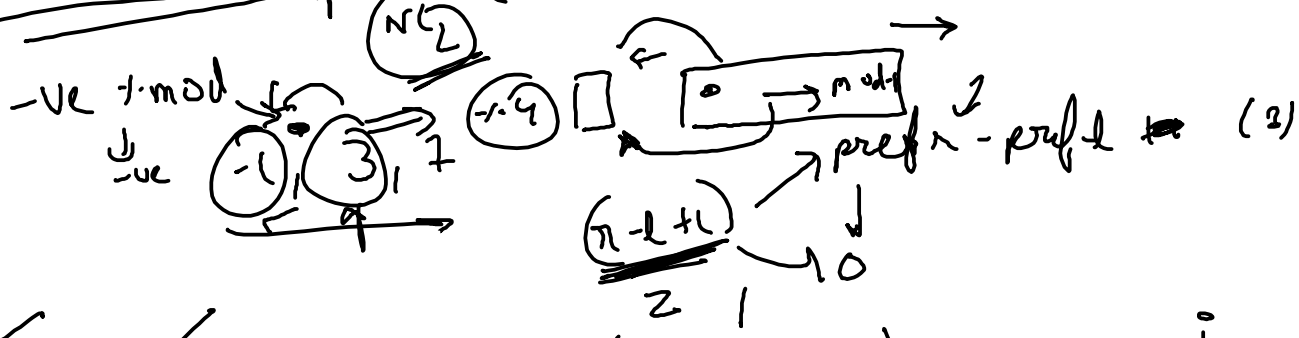
How many j had same val.



Count pairs where (Pref[i] - Pref[j]) % mod = (i - j) % mod

$Pref[i] \% mod - pref[j] \% mod = i \% mod - j \% mod$

$(pref[i] - i) \% mod = (pref[j] - j) \% mod$



Given a Binary string, count substrings with freq_1 > freq_0.

$pref[r] - pref[l-1] > \frac{r-l+1}{2}$

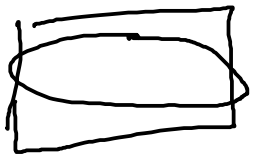
Count of 1s in my range

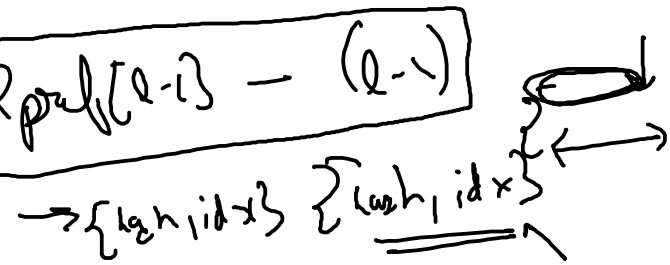
$2pref[r] - 2pref[l-1] > r - l + 1$

$val \rightarrow 2pref[r] - r > 2pref[l-1] - (l-1)$

Val ⇒ 2pref - i

ans += Count of j that have val < val - i





query[hash-me, inf}