MATHEMATICAL GRAMMAR SCHOOL CUP
June 27, 2019
https://arena.petlja.org/en-US/competition/mgcup2019

Information

Grading	TakeBest
Penalty	PenaltyDisabled
MaxNoSubmissions	8 per task
SourceLimit	100 KB

TASK1 MG numbers(English)

Time limit: 2.4 seconds
Memory limit: 40 MB

MG arrays are special series of positive integers such that the greatest common divisor of all the elements is greater than 1.
You are given an array consisting of ***N*** positive integers.
It is possible to formulate some queries directed to that array.

Each query could be one of the two types:
1. Change the value at position ***X*** in the array to ***V***.
2. Determine the number of MG contiguous subarrays contained in the interval [L, R] of the array.

The first line of input contains the numbers ***N*** and ***Q*** ($1 \leq N$, $Q \leq 10^5$), representing respectively the number of elements in the array and the number of queries.
In the next line, there are ***N*** natural numbers A_i ($1 \leq A_i \leq 10^9$) that represent the numbers in the given array.

Each of the following Q lines contains a query of the following form:
- The first number in the line is number 1 or number 2 and represents the type of the query.
- If the query is of type 1, two numbers follow, ***X*** ($1 \leq X \leq N$) and ***V*** ($1 \leq V \leq 10^9$) from the problem formulation.
- If the query is of type 2, two numbers follow, ***L*** and ***R*** ($1 \leq L \leq R \leq N$) that represent the left and right interval boundary.

Write the program that print on the standard output the number of MG contiguous subarrays from the task (for each query of type 2).

Input 1
5 1
8 4 3 9 1
2 2 5
Output 1
4
Clarification:
The interval from the second to the fifth position consists of numbers (4, 3, 9, 1).
There are following MG contiguous subarrays (denoted with square brackets):
[4] 3 9 1
4 [3] 9 1
4 3 [9] 1
4 [3 9] 1

Input 2
4 3
2 2 2 2
2 1 4
1 2 3
2 1 4

Output 2
10
5

TASK 2 LED DISCO STROBE

Time limit:0.1 seconds Memory limit: 4 MB

There are ***N*** LED disco strobes (1≤***N***≤50) numbered from 1 to ***N***.
For each LED disco strobe it is known whether it's turned on or off.
You can change the state of LED disco strobe ***i*** if LED disco strobe ***i+1*** is turned on and LED disco strobe ***i+2***, ***i+3,…***,***N*** are turned off. This rule is not targeted toward LED disco strobe N, which can be switched on or off voluntarily.

Write the program that prints the minimum number of switches you need to make in order to turn off all the LED disco strobes.
The standard input contains a single string of ***N*** binary digits (0 or 1). Each LED disco strobe is represented by a binary digit.
A LED disco strobe that is switched off is represented by a 0, and one that is switched on by a 1.

Input 1
1111
Output 1
10

Input 2
1000
Output 2
15

Input 3
10001000
Output 3
240

MATHEMATICAL GRAMMAR SCHOOL CUP
June 27, 2019 (SOLUTIONS)

1.
Let us change the value of position *i*.
Let's find the leftmost position *j* such that $gcd(j, i) = A_i$, *where* $gcd(l, r)$ is the greatest common divisor of the range $Al, Al + 1, \ldots, Ar$.
From now, we are sure that the $gcd(j - 1, r)$ will decrease.
Let's find the leftmost position j_2, such that $gcd(j_2, i) = gcd(j - 1, i)$.
If we do this until $gcd(j_{k-1}, i) = 1$, we will have $O(\log 10^9)$ such ranges can we in worst case.

We will solve this task using a segment tree where we will store the following data for each interval:

● greatest common divisor of all prefixes P of the given array (as an array of pairs (g d)), where g represents one of the values P_i , and d represent the number of appearances of that value in array P

● greatest common divisor of all suffixes S as an array of pairs (g d)), where g represents one of the values S_i , and d represent the number of appearances of that value in array S

● the number of MG contiguous subsequences in that interval.

2. Let us note from the input 2 that we need 2^N-1 moves in order to turn off a sequence `1000...00` of `N` LEDs.
So, we can have greedy solution in order to always start from the *least significant bit*.

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
    ios::sync_with_stdio(false);
    string s;
    cin >> s;
    long long moves = 0, LSB = 0;
    for (char x : s)
    {
        LSB ^= (x – '0');  //mirroring
        moves = 2 * moves + lsb;
    }
    cout << moves << endl;
}
```