Task: DEV
Developer

TORUŃ 2025

BOI 2025, Day 2. Available memory: 512 MB. *2025.04.27*

You are in charge of developing new properties in the suburbs of Toruń. You have already decided that there will be one main street and n properties numbered from 1 to n along the street. The area is somewhat hilly, and the elevation of the i-th property is a_i centimetres.

It turns out that no one wants to buy a property that is on a *slope*. Formally, for elevations $a_1, a_2, \ldots, a_n$, a slope is a contiguous subsequence $a_{i-1}, a_i, \ldots, a_j, a_{j+1}$ with $2 \leq i \leq j \leq n-1$ such that either (i) $a_{i-1} < a_i = a_{i+1} = \ldots = a_j < a_{j+1}$, or (ii) $a_{i-1} > a_i = a_{i+1} = \ldots = a_j > a_{j+1}$. Intuitively, a slope is a contiguous range of properties at positions $i-1, i, i+1, \ldots, j, j+1$, where the elevations of all properties at positions $i, i+1, \ldots, j$ are equal to some h, and h is strictly between a_{i-1} and a_{j+1}.

You are able to increase or decrease the elevation of any property by any integer, but of course you want to minimise the overall effort. Your task is to determine the minimal total change in elevation such that there are no slopes at all. That is, you want to find elevations $b_1, b_2, \ldots, b_n$ without slopes such that $|a_1 - b_1| + |a_2 - b_2| + \ldots + |a_n - b_n|$ is as small as possible. The elevations b_i must be integers (in particular, they don't have to be positive), and there are no other constraints on b_i.

Input

The first line contains an integer n ($1 \leq n \leq 2 \cdot 10^5$) denoting the number of properties along the street.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 10^9$), where the i-th integer a_i is the initial elevation of the i-th property.

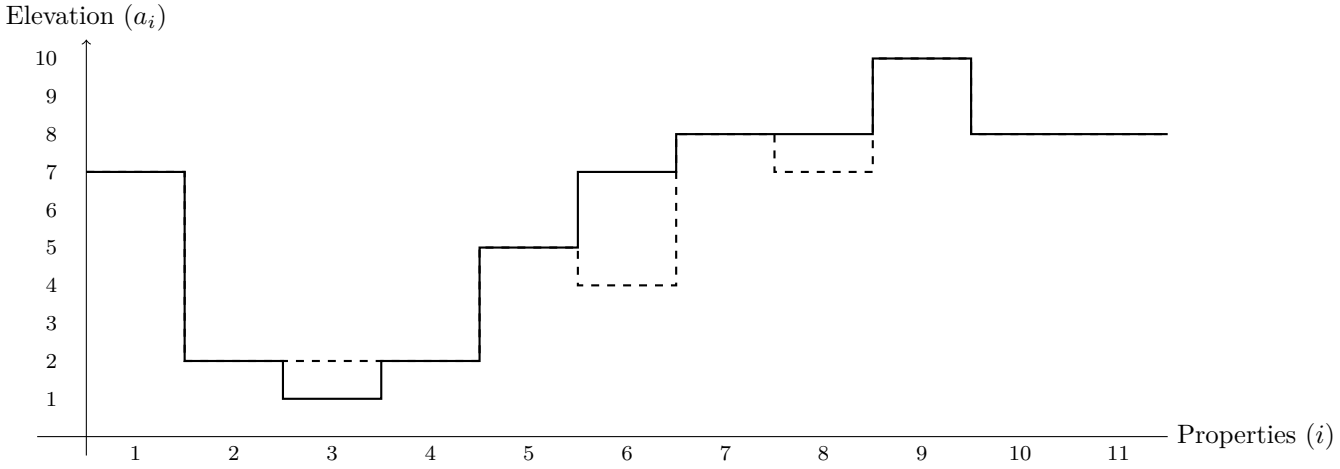
Output

You should output the minimal total change in elevation to ensure that there are no slopes.

Example

For the input data:	the correct result is:
11 7 2 1 2 5 7 8 8 10 8 8	5

This is illustrated below. The dashed lines represent the changed elevations without slopes b_i of their corresponding properties.



Scoring

Subtask	Constraints	Points
1	$n \leq 5$ and $a_i \leq 10$	4
2	$n \leq 2000$	13
3	$a_i \leq 10$	8
4	$a_i < a_{i+1}$	19
5	$n \leq 2 \cdot 10^4$	29
6	No additional constraints.	27

Task: EXP
Exponents

BOI 2025, Day 2. Available memory: 1024 MB. *2025.04.27*

The famous polymath Nicolaus Copernicus was born and grew up in Toruń in the 15th century. Archaeologists have recently discovered his notebook, and learned that he was fond of using powers of two to store large numbers. In particular, even when he added two powers of two:

$$2^a + 2^b,$$

Copernicus evaluated the result and then rounded up the result to the nearest power of two. That is, he would evaluate $2^a + 2^b$ to $2^{\max(a,b)+1}$. To evaluate a longer expression of the form:

$$2^{b_1} + 2^{b_2} + \ldots + 2^{b_k},$$

he first inserted the brackets to make it well-parenthesised*. For example, an expression $2^5 + 2^4 + 2^4 + 2^4 + 2^5$ can be made well-parenthesised to obtain $((2^5 + 2^4) + (2^4 + (2^4 + 2^5)))$. Finally, he evaluated the result of the obtained well-parenthesised expression, operating on powers of two as described above. Notice that the obtained result might vary depending on how he inserts the brackets. For example, here are two possible ways to evaluate $2^5 + 2^4 + 2^4 + 2^4 + 2^5$:

$$(((2^5 + 2^4) + 2^4) + (2^4 + 2^5)) = ((2^6 + 2^4) + 2^6) = (2^7 + 2^6) = 2^8$$
$$((2^5 + (2^4 + 2^4)) + (2^4 + 2^5)) = ((2^5 + 2^5) + 2^6) = (2^6 + 2^6) = 2^7$$

The first page of the Copernicus' notebook contains only a single expression $2^{a_1} + 2^{a_2} + \ldots + 2^{a_n}$ called the main expression. Later pages of the notebook then reference fragments of the main expression, which are of the form $2^{a_\ell} + 2^{a_{l+1}} + \ldots + 2^{a_r}$, for some $1 \le \ell \le r \le n$.

You are not sure about their meaning, but suspect that you should calculate, for each such fragment, the smallest possible result that can be obtained when evaluating the result as described above for the fragment. Note that each fragment is evaluated independently of the other fragments.

Input

The first line contains two integers n and q ($1 \le n, q \le 300\,000$) denoting the length of the main expression from the first page of the notebook and the number of queries, respectively.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^6$), where the i-th integer a_i denotes the exponent of the i-th power of two in the main expression.

The next q lines describe the queries. Each query consists of two integers ℓ and r ($1 \le \ell \le r \le n$) representing a fragment of the main expression starting at the ℓ-th power of two and ending at the r-th power of two.

Output

You should output q lines. The i-th line should contain the smallest possible result that can be obtained when evaluating the fragment described in the i-th query. You should output only the exponent of the corresponding power of two.

Example

For the input data:

```
8 4
2 4 2 5 4 4 4 5
4 8
1 4
2 5
1 7
```

the correct result is:

```
7
7
7
8
```

*The formal definition of a well-parenthesised expression is as follows: 2^a is a well-parenthesised expression for any non-negative integer a; if E_1 and E_2 are well-parenthesised expressions then so is $(E_1 + E_2)$. No other expressions are well-parenthesised.

Scoring

Subtask	Constraints	Points
1	$n \leq 8$, $q \leq 10$	6
2	$n \leq 200$	8
3	$n, q \leq 2000$	23
4	$a_i \leq 20$	22
5	No additional constraints.	41

Task: GCD
Gingerbread

TORUŃ 2025

BOI 2025, Day 2. Available memory: 256 MB. *2025.04.27*

Toruń has been known for its traditional gingerbread since the Middle Ages. Young Nicolaus would like to buy a set of n boxes with gingerbread cookies in his favourite shop. The shop has very strict rules, though: Nicolaus initially obtains n boxes that are already filled with cookies: the i-th box initially contains a_i of them. Then, Nicolaus can order some extra cookies. He adds extra cookies to some boxes so that the greatest common divisor* of the numbers of cookies in all of the boxes becomes equal to 1. It can be proven that this is always possible.

Help Nicolaus by calculating the smallest number of cookies that need to be added in order to make the greatest common divisor of all the numbers equal to 1.

Input

The first line contains an integer n ($2 \leq n \leq 10^6$), denoting the number of boxes.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^7$), where the i-th integer a_i denotes the initial number of cookies in the i-th box.

Output

Output one line with a single integer denoting the smallest number of cookies that Nicolaus should add to the boxes. If Nicolaus doesn't have to add any cookies to make the greatest common divisor of the numbers equal to 1, output 0.

Example

For the input data:	the correct result is:
3 90 84 140	2

Explanation of the example: Indeed, the greatest common divisor (GCD) of 90, 84, and 140 is 2, so some cookies must be added. If we add only one cookie, we may obtain the quantities 91, 84, 140 with GCD of 7, or 90, 85, 140 with GCD of 5, or 90, 84, 141 with GCD of 3, so this is not enough. After adding two cookies, one to the first box, and one to the second box, we obtain the quantities 91, 85, 140 with GCD of 1; hence the answer is 2. Note that adding both cookies to the first box does not help: we obtain quantities 92, 84, 140 with GCD of 4.

Scoring

Subtask	Constraints	Points
1	$n = 2$	17
2	$n \leq 10$	34
3	$n \leq 1000$	11
4	No additional constraints.	38

*The greatest common divisor (GCD) of multiple numbers is the largest positive integer that divides all of them without remainder.