D. Fence

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

John Doe has a crooked fence, consisting of n rectangular planks, lined up from the left to the right: the plank that goes i-th $(1 \le i \le n)$ (from left to right) has width 1 and height h_i. We will assume that the plank that goes i-th $(1 \le i \le n)$ (from left to right) has index i.

A *piece of the fence* from l to r $(1 \le l \le r \le n)$ is a sequence of planks of wood with indices from l to r inclusive, that is, planks with indices $l, l+1, ..., r$. The width of the piece of the fence from l to r is value $r - l + 1$.

Two pieces of the fence from l_1 to r_1 and from l_2 to r_2 are called matching, if the following conditions hold:

- the pieces do not intersect, that is, there isn't a single plank, such that it occurs in both pieces of the fence;
- the pieces are of the same width;
- for all i $(0 \le i \le r_1 - l_1)$ the following condition holds: $h_{l_1+i} + h_{l_2+i} = h_{l_1} + h_{l_2}$.

John chose a few pieces of the fence and now wants to know how many distinct matching pieces are for each of them. Two pieces of the fence are distinct if there is a plank, which belongs to one of them and does not belong to the other one.

Input

The first line contains integer n $(1 \le n \le 10^5)$ — the number of wood planks in the fence. The second line contains n space-separated integers $h_1, h_2, ..., h_n$ $(1 \le h_i \le 10^9)$ — the heights of fence planks.

The third line contains integer q $(1 \le q \le 10^5)$ — the number of queries. Next q lines contain two space-separated integers l_i and r_i $(1 \le l_i \le r_i \le n)$ — the boundaries of the i-th piece of the fence.

Output

For each query on a single line print a single integer — the number of pieces of the fence that match the given one. Print the answers to the queries in the order, in which the queries are given in the input.

Examples

input Copy

```
10
1 2 2 1 100 99 99 100 100 100
6
1 4
1 2
3 4
1 5
9 10
10 10
```

output Copy

```
1
2
2
0
```

```
2
9
```