

Jump Game

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

points: 1 · problem code: pondojumpgame · author: jszqlew

Problem Statement

You are given a list of positive integers a of length n . Each integer represents the distance a trampoline will send you to the right. In particular, if you stand on the i^{th} integer, then you will bounce to coordinate $i + a_i$ (this will keep happening until you bounce past all the trampolines).

You are asked to perform operations of the following form.

- Type 1: 1 i x - Adjust the i^{th} trampoline to bounce x to the right. OR
- Type 2: 2 i - Determine the number of bounces before you make it past all the trampolines if you started on the i^{th} trampoline.

After adjusting a trampoline, the trampoline will be adjusted permanently and all future operations should take that into account.

Input Statement

Your first line will contain n and q , representing the length of a and the number of operations respectively. Your next line will contain n space-separated integers representing the values of a . Your next q lines will contain 2 or 3 integers each, representing the operations described. If the line starts with a 1, then you will receive 3 integers, 1, i , and x . If the line starts with a 2, then you will receive 2 integers, 2 and i .

Output Statement

For each type 2 query, you should output a new line with an integer, representing the number of bounces before you make it past all the trampolines for the corresponding type 2 operation.

You should output these in the order the operations occurred in the input.

Constraints

- $1 \leq n \leq 10^4$
- $1 \leq q \leq 10^4$
- Queries are 1-indexed.
- $1 \leq a_i \leq 10^4$

Sample Cases

Input 1

```
10 7
1 1 1 1 1 1 1 1 1 1
1 1 5
2 1
2 2
1 1 1
2 1
1 5 10000
2 3
```

Output 1

```
6
9
10
3
```

Explanation 1

The array is initially: `1 1 1 1 1 1 1 1 1 1` After performing `1 1 5` it becomes: `5 1 1 1 1 1 1 1 1 1` After querying `2 1`, we know it will take `6` jumps to exit the trampolines: You start at `1`, jump to `6`, then to `7`, then to `8`, then to `9`, then to `10`, then past the trampolines. After querying `2 2` we know that it will take `9` jumps, (jump `1` to the right every time starting from the 2^{nd} trampoline). The rest of the operations follow...