Periodic function, trip 5

Solar cycle predictions are used by various agencies and many industry groups. The solar cycle is important for determining the lifetime of satellites in low-Earth orbit, as the drag on the satellites correlates with the solar cycle [...]. (NOAA)

(Solar Cycle)

Sunspot Number Progression : Observed data through May 2008 ; Dec 2012 ; Nov 2014 ; Jun 2016

The goal of the problem is to propose a perfect prediction center, with not so weak constraints.

Let us consider periodic functions from **Z** to **R**.

```
def f(x): return [4, -6, 7][x%3] # (with Python notations)
# 4, -6, 7, 4, -6, 7, 4, -6, 7, 4, -6, 7, 4, -6, 7, ...
```

For example, f is a 3-periodic function, with $f(0) = f(3) = f(6) = f(9) = 4$.
With a simplified notation we will denote f as [4, -6, 7].
For two periodic functions (with integral period), the quotient of periods will be rational, in that case it can be shown that the sum of the functions is also a periodic function. Thus, the set of all such functions is a vector space over **R**.

For that problem, we consider a function that is the sum of several periodic functions all with as period an integer N at maximum. You will be given some starting values, you'll have to find new ones.

Input

On the first line, you will be given an integer N.
On the second line, you will be given integers y : the first (0-indexed) $N \times N$ values of a periodic function f that is sum of periodic functions all with as period an integer N at maximum.
On the third line, you will be given $N \times N$ integers x.

Output

Print $f(x)$ for all required x. See sample for details.

Example

Input:
3
15 3 17 2 16 4 15 3 17

10 100 1000 10000 100000 1000000 10000000 100000000 1000000000

Output:
16 16 16 16 16 16 16 16 16

Explanation

For example *f* can be seen as the sum of three periodic functions : [10] + [5, -8] + [0, 1, 2] (with simplified notations ; periods are 1,2 and 3)
In that case *f*(10) = [10][10%1] + [5, -8][10%2] + [0, 1, 2][10%3] = 10 + 5 + 1 = 16, and so on.

Constraints

N < 258
abs(y) < 10^9
0 <= x < 10^9

For PERIOD4 you can have AC with $O(N^6)$ method, for PERIOD3 the awaited solution is about $\pi^6/27$ faster.
For PERIOD5 a new complexity is awaited.

Informations

You can safely assume output fit in a signed 32bit container. There's 6 input files, with increasing value of *N*. My modest C code ended in 1.27s ; no optimization. Some details (#i, *N*, TL, t) : (#0, around 50, 1s, 0s), (#1, around 50, 1s, 0s), (#2, around 100, 1s, 0.04s), (#3, around 150, 3s, 0.14s), (#4, around 200, 7s, 0.36s), (#5, around 250, 15s, 0.74s).
You may first try the medium edition PERIOD3. **Have fun ;-)**

(Edit 2017-02-11 ; compiler update ; here ×2 speedup) Some updated details (#i, *N*, TL, t) : (#0, around 50, 1s, 0s), (#1, around 50, 1s, 0s), (#2, around 100, 1s, 0.02s), (#3, around 150, 3s, 0.07s), (#4, around 200, 7s, 0.18s), (#5, around 250, 15s, 0.36s).