Combination Lock

Farmer John's cows keep escaping from his farm and causing mischief. To try and prevent them from leaving, he purchases a fancy combination lock to keep his cows from opening the pasture gate.

Knowing that his cows are quite clever, Farmer John wants to make sure they cannot easily open the lock by simply trying many different combinations. The lock has three dials, each numbered 1..N ($1 \le N \le 100$), where 1 and N are adjacent since the dials are circular. There are two combinations that open the lock, one set by Farmer John, and also a "master" combination set by the lock maker.

The lock has a small tolerance for error, however, so it will open even if the numbers on the dials are each within at most 2 positions of a valid combination.

For example, if Farmer John's combination is (1,2,3) and the master combination is (4,5,6), the lock will open if its dials are set to (1,3,5) (since this is close enough to Farmer John's combination) or to (2,4,8) (since this is close enough to the master combination). Note that (1,5,6) would not open the lock, since it is not close enough to any one single combination.

Given Farmer John's combination and the master combination, please determine the number of distinct settings for the dials that will open the lock. Order matters, so the setting (1,2,3) is distinct from (3,2,1).

PROGRAM NAME: combo

INPUT FORMAT:

Line 1:	The integer N.
Line 2:	Three space-separated integers, specifying Farmer John's combination.
Line 3:	Three space-separated integers, specifying the master combination (possibly the same as Farmer John's combination).

SAMPLE INPUT (file combo.in):

```
50
1 2 3
5 6 7
```

INPUT DETAILS:

Each dial is numbered 1..50. Farmer John's combination is (1,2,3), and the master combination is (5,6,7).

OUTPUT FORMAT:

Line 1: The number of distinct dial settings that will open the lock.

SAMPLE OUTPUT (file combo.out):

```
249
```

SAMPLE OUTPUT EXPLANATION

Here's a list:

```
1,1,1  2,2,4  3,4,2  4,4,5  5,4,8  6,5,6  7,5,9  3,50,2  50,1,4
1,1,2  2,2,5  3,4,3  4,4,6  5,4,9  6,5,7  7,6,5  3,50,3  50,1,5
1,1,3  2,3,1  3,4,4  4,4,7  5,5,5  6,5,8  7,6,6  3,50,4  50,2,1
1,1,4  2,3,2  3,4,5  4,4,8  5,5,6  6,5,9  7,6,7  3,50,5  50,2,2
1,1,5  2,3,3  3,4,6  4,4,9  5,5,7  6,6,5  7,6,8  49,1,1  50,2,3
1,2,1  2,3,4  3,4,7  4,5,5  5,5,8  6,6,6  7,6,9  49,1,2  50,2,4
1,2,2  2,3,5  3,4,8  4,5,6  5,5,9  6,6,7  7,7,5  49,1,3  50,2,5
1,2,3  2,4,1  3,4,9  4,5,7  5,6,5  6,6,8  7,7,6  49,1,4  50,3,1
1,2,4  2,4,2  3,5,5  4,5,8  5,6,6  6,6,9  7,7,7  49,1,5  50,3,2
1,2,5  2,4,3  3,5,6  4,5,9  5,6,7  6,7,5  7,7,8  49,2,1  50,3,3
1,3,1  2,4,4  3,5,7  4,6,5  5,6,8  6,7,6  7,7,9  49,2,2  50,3,4
1,3,2  2,4,5  3,5,8  4,6,6  5,6,9  6,7,7  7,8,5  49,2,3  50,3,5
1,3,3  3,1,1  3,5,9  4,6,7  5,7,5  6,7,8  7,8,6  49,2,4  50,4,1
1,3,4  3,1,2  3,6,5  4,6,8  5,7,6  6,7,9  7,8,7  49,2,5  50,4,2
1,3,5  3,1,3  3,6,6  4,6,9  5,7,7  6,8,5  7,8,8  49,3,1  50,4,3
1,4,1  3,1,4  3,6,7  4,7,5  5,7,8  6,8,6  7,8,9  49,3,2  50,4,4
1,4,2  3,1,5  3,6,8  4,7,6  5,7,9  6,8,7  1,50,1 49,3,3  50,4,5
1,4,3  3,2,1  3,6,9  4,7,7  5,8,5  6,8,8  1,50,2 49,3,4  49,50,1
1,4,4  3,2,2  3,7,5  4,7,8  5,8,6  6,8,9  1,50,3 49,3,5  49,50,2
1,4,5  3,2,3  3,7,6  4,7,9  5,8,7  7,4,5  1,50,4 49,4,1  49,50,3
2,1,1  3,2,4  3,7,7  4,8,5  5,8,8  7,4,6  1,50,5 49,4,2  49,50,4
2,1,2  3,2,5  3,7,8  4,8,6  5,8,9  7,4,7  2,50,1 49,4,3  49,50,5
2,1,3  3,3,1  3,7,9  4,8,7  6,4,5  7,4,8  2,50,2 49,4,4  50,50,1
2,1,4  3,3,2  3,8,5  4,8,8  6,4,6  7,4,9  2,50,3 49,4,5  50,50,2
2,1,5  3,3,3  3,8,6  4,8,9  6,4,7  7,5,5  2,50,4 50,1,1  50,50,3
2,2,1  3,3,4  3,8,7  5,4,5  6,4,8  7,5,6  2,50,5 50,1,2  50,50,4
2,2,2  3,3,5  3,8,8  5,4,6  6,4,9  7,5,7  3,50,1 50,1,3  50,50,5
2,2,3  3,4,1  3,8,9  5,4,7  6,5,5  7,5,8
```

Submit a solution:

Choose File | No file chosen | Send it in!

Training Gateway | Comment or Question