UVa Online Judge

11793 Electoral Districts

In a certain democracy, the party in the government is reorganizing the electoral districts in order to benefit them in the next election.

One delegate is elected for each district. And, as the regime is *de facto* a two-party system, they just need one more vote than the opposition for each district.

The country is divided into a square mesh of $N \times N$ zones; these zones are grouped into electoral districts. There are N districts, and each district is composed by N adjacent zones, which means each zone in the district has a common border (on the left, right, up or down) with at least another zone in the same district.

Suppose the party in the government is A, and the party in the opposition is B. From previous elections, we know the number of votes for each party in each zone. Two $N \times N$ matrices are available, one with the number of votes for party A, and another one for the votes for party B.

For each district, we take the sum of the votes of its zones. Then, the party with more total votes in that district gains a delegate. If the number of votes is equal for A and B, then no delegate is assigned.

Party A has hired you to compute the largest possible difference they can obtain —in number of delegates— with an adequate arrangement of the districts.

Input

The input begins with a line where the number of test cases (T) is indicated. The data for each test case appear in successive lines. For each test case, the first line contains the dimension of the country (N, with maximum value 5, and the country has N by N zones). Following, there are $2N$ lines, each line with N integers between 1 and 1000, separated by a space. The first N lines contain the votes for party A, and the other N lines the votes for party B.

Output

The output consists of T lines, one for each test case. For each case, the maximum achievable difference between the number of delegates obtained by party A with respect to party B is represented.

Sample Input

```
4
2
2 3
2 4
3 1
2 3
3
2 3 4
1 3 2
2 3 5
3 4 1
3 2 1
2 1 3
3
1 2 1
```

```
2 1 2
1 2 1
2 1 2
1 2 1
2 1 2
3
1 1 1
1 1 1
1 1 1
2 2 2
2 2 2
2 2 2
```

Sample Output

```
2
2
1
-3
```