# X Diagonal

**Time limit:** 0.6s **Memory limit:** 512M
Java (latest): 1.5s
PyPy 3: 1.5s

Given a $N$ by $N$ matrix, find the **maximum** sum of a main diagonal (/) and an antidiagonal (\), and if they intersect at a cell, subtract that cell because it is only counted once.

Note: For this problem, Python users are recommended to use PyPy over CPython.

## Constraints

$3 \leq N \leq 2000$

The elements in the matrix are between $1$ and $10\,000$, inclusive.

### Subtask 1 [40%]

$3 \leq N \leq 25$

### Subtask 2 [60%]

No additional constraints.

## Input Specification

The first line of input contains a single integer, $N$, the number of rows and columns in the matrix.

The next $N$ lines of input contain $N$ space separated integers, the elements in the matrix.

## Output Specification

Output a single integer, the **maximum** sum of a main diagonal (/) and an antidiagonal (\), minus their intersection if they intersect at a cell (intersection has integer coordinates inside the matrix).

## Sample Input 1

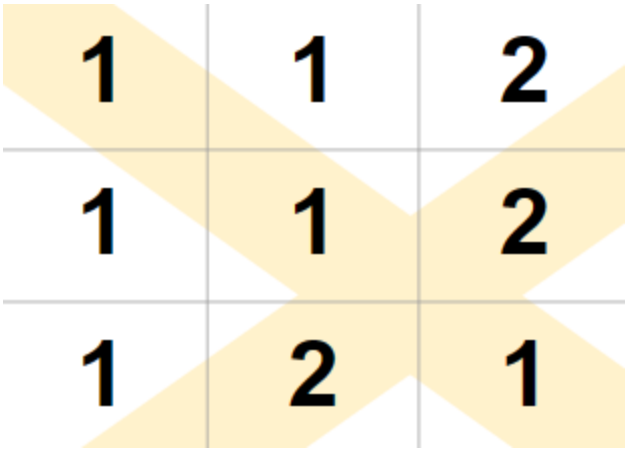
```
3
1 1 2
1 1 2
1 2 1
```

## Sample Output 1

```
7
```

## Explanation for Sample Output 1

The 2 diagonals don't overlap so we just add their values: $(1+1+1)+(2+2)=7$




## Sample Input 2

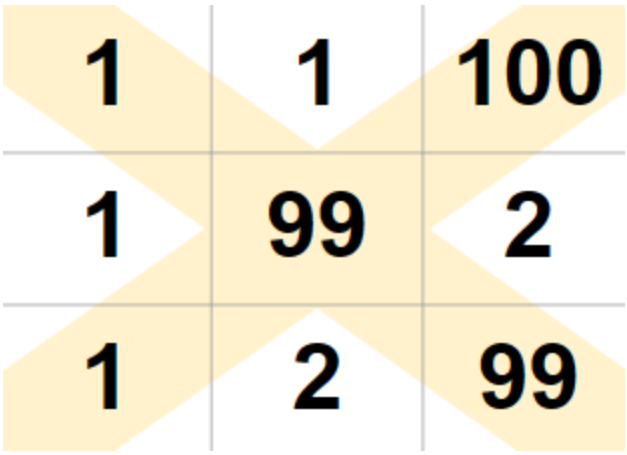
```
3
1 1 100
1 99 2
1 2 99
```

## Sample Output 2

```
300
```

## Explanation for Sample Output 2

The 2 diagonals overlap so we subtract the intersection, as it is only counted once:
$(1+99+100)+(1+99+99)-99=300$




## Sample Input 3

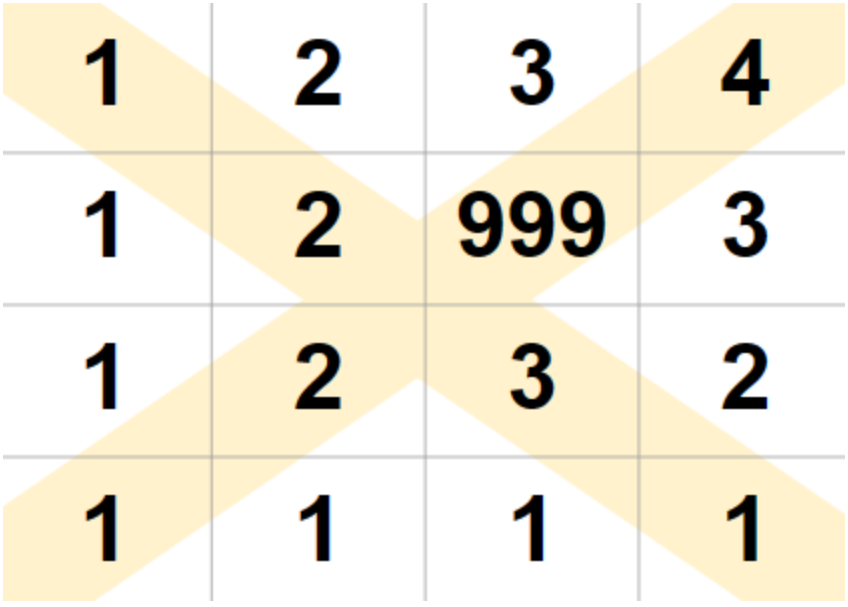
```
4
1 2 3 4
1 2 999 3
1 2 3 2
1 1 1 1
```

## Sample Output 3

```
1013
```

## Explanation for Sample Output 3

The 2 diagonals don't overlap so we just add their values: $(1 + 2 + 999 + 4) + (1 + 2 + 3 + 1) = 1013$



## Sample Input 4

```
4
1 1 9 1
1 1 1 9
1 1 1 1
1 1 1 9
```

## Sample Output 4

```
27
```

## Explanation for Sample Output 4

The corners also count as a diagonals. $(9 + 9) + 9 = 27$