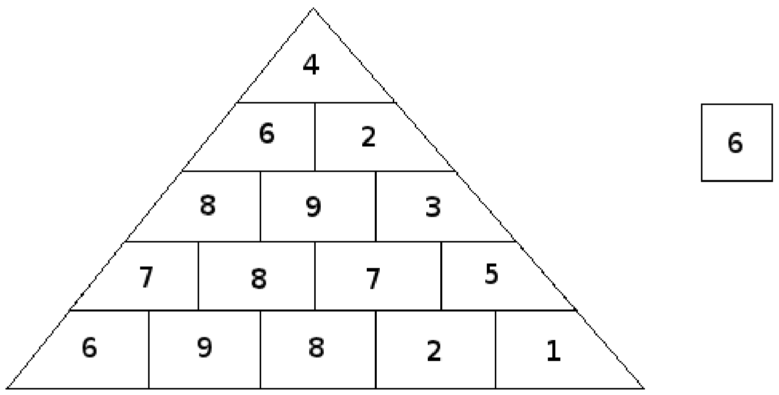
Hopscootch

Input File: *standard input*
Output File: *standard output*

Time and Memory Limits: 1 second, 32 MB

Puny human! All you humies played Hopscotch as a kid, but did you ever play *Hopscootch*? Well I did, and **I** was the greatest *Hopscootch* player to have ever lived until I took an arrow to the knee.

Here's how *Hopscootch* works. We play on a pyramid with N rows. The top row has one number, while the bottom row has N numbers. You must walk from the single number at the top of the pyramid to any of the numbers on the bottom row. In each of your $N - 1$ steps, you must go to one of the two adjacent numbers in the row below your current position (that is, you can only walk "down" the pyramid). Every number you step on (including your starting and finishing numbers) increases your score by its value – but beware that the negative ones will reduce your score! If you're gonna beat me, you gotta end up with a higher score than me.



Since you're just a human playing for your first time, let's make it easier. I'll give you a list of K extra numbers you can use. Before you start your walk, you can choose to replace some of the numbers in the pyramid with numbers in this list. Of course you can only use each extra number at most once!

What's the highest score **you** can get, humie?

Input

The first line will contain two space-separated integers N and K. The second line will contain the space-separated list of K extra single-use numbers that can be substitued into the pyramid. The next N lines will then contain the numbers in the pyramid. The ith of these lines will contain i integers, indicating the numbers on row i.

Output

Output should consist of a single integer: the maximum possible score of a *Hopscootch* walk.

Sample Input 1

```
5 1
6
4
6 2
8 9 3
7 8 7 5
6 9 8 2 1
```

Sample Output 1

```
38
```

Sample Input 2

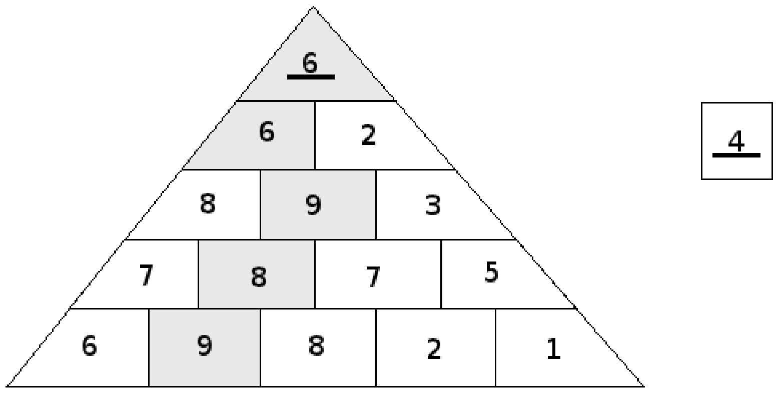
```
6 3
-1 3 -2
3
1 2
4 1 2
5 5 15 -1
-3 -5 -3 -4 -3
11 18 14 2 25 -6
```

Sample Output 2

```
50
```

Explanation

The first sample case is the example given earlier:



We can solve this case by replacing the 4 with the single-use 6, then walking down the pyramid taking the shaded path. This gives a total of 38.

In the second sample case, one of the optimal solutions is to first replace the -4 with the single-use 3, then walk down the pyramid taking numbers 3, 2, 2, 15, 3, 25. This gives a total of 50.

Subtasks & Constraints

In all subtasks, the values of the numbers in the pyramid and in the extra list will be integers between $-1\,000\,000$ and $1\,000\,000$, inclusive.

- For Subtask 1 (40 points), $1 \leq N \leq 100$ and $K = 1$.
- For Subtask 2 (40 points), $1 \leq N \leq 100$ and $1 \leq K \leq N$.
- For Subtask 3 (20 points), $1 \leq N \leq 300$ and $1 \leq K \leq N$.