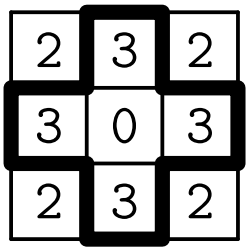
Problem A. Tube Master II

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Yuuka is playing "Tube Master". The game field is divided into $n \times m$ cells and $(n+1) \times (m+1)$ crossings connected by $(n+1) \times m$ horizontal tubes and $n \times (m+1)$ vertical ones. The cells are conveniently labeled with (i, j) for $1 \le i \le n$, $1 \le j \le m$, and the crossings are labeled with (i, j) for $1 \le i \le (n+1)$, $1 \le j \le (m+1)$. Additionally, each cell (i, j) contains an integer $count_{i,j}$.



The above figure shows a game field with $n = m = 3$ (the third sample).

Yuuka decides to use some of the tubes. However, the game poses several weird restrictions.

1. Either 0 or 2 tubes connected to each crossing are used.
2. No two consecutive horizontal tubes are used simultaneously, and no consecutive vertical tubes are used simultaneously. Two tubes are consecutive if and only if they share the same crossing.
3. Exactly $count_{i,j}$ tubes surrounding cell (i, j) are used.

Using the tube connecting crossing (i, j) and $(i, j+1)$ costs $a_{i,j}$, and using the tube connecting crossing (i, j) and $(i+1, j)$ costs $b_{i,j}$. Yuuka would like to find a configuration satisfying the above constrains with the minimum possible total cost.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains two integers n and m ($1 \le n, m \le 100$).

The i-th of the following n lines contains m integers $count_{i,1}, count_{i,2}, \ldots, count_{i,m}$ ($0 \le count_{i,j} \le 4$).

The i-th of the next $(n+1)$ lines contains m integers $a_{i,1}, a_{i,2}, \ldots, a_{i,m}$.

The i-th of the last n lines contains $(m+1)$ integers $b_{i,1}, b_{i,2}, \ldots, b_{i,m+1}$.

The constraints are: $1 \le a_{i,j}, b_{i,j} \le 10^9$.

It is guaranteed that the total sum of $n \cdot m$ in all test cases does not exceed 10^4.

Output

For each test case, output an integer which denotes the minimum cost of the configuration. If there is no valid configuration, output "`-1`" instead.

Example

standard input	standard output
1 3 4 2 4 1 1 1 1 1 1 1 1 1 1 1 2 3 3 1 1 1 1 1 1 1 3 3 2 3 2 3 0 3 2 3 2 1 2 3 4 5 6 7 8 9 11 12 13 1 2 3 4 5 6 7 8 9 10 11 12	8 -1 79