Problem A. MaxSum (lucky sum—1)

Input file:	`stdin`
Output file:	`stdout`
Time limit:	2 seconds
Memory limit:	64 megabytes

There is a rectangular table with N rows and M columns. There is an integer in each cell. It is necessary to pass it from the top to bottom. You can start from any cell in top row, then you can move to one of the “neighbour bottom” cells (in other words, from cell (i, j) you can move to $(i + 1, j$–$1)$ or $(i + 1, j)$, or $(i + 1, j + 1)$; in the case of $j = M$ last movement becomes impossible, and in the case of $j = 1$ — the first one) and you should end route in any cell of the bottom row.

Write a program that will find the greatest possible *lucky sum* of the values of passed cells among all admissible paths. Everyone knows that the lucky numbers are positive integers which decimal representation containing only the lucky digits 4 and 7. For example, the numbers 47, 744, 4 are lucky and 0, 5, 17, 467 are not. Please note that the sum itself should be lucky, not the individual values of passed cells.

Input

The first line contains N and M — the number of rows and number of columns ($1 \leq N$, $M \leq 77$), then each of the N lines contains exactly M space-separated integers (each is within the range $0 \leq a_{ij} \leq 77$) — values of table’s cells.

Output

Print a single integer — the maximum possible *lucky sum*. In case of no admissible paths with lucky sum print line “`impossible`” (without quotes, small Latin characters).

Examples

stdin	stdout

`3 4`
`8 2 10 14`
`22 2 15 25`
`1 14 9 1`
	`44`