Matrix Words

Given an NxN matrix of characters. We start at position (1,1) and want to reach (N,N) in exactly 2N-1 moves. Each move consists of movement in one of the four standard directions. As we move, we collect the characters found in our positions forming a string. We now constrain our attention to all paths that do not cross the diagonal of the matrix. However the parts of the path can be on the diagonal line. These paths can be classified into two partitions; the paths that lie above and paths that lie below the diagonal. Each path is represented by a string of characters formed by the ordered concatenation of characters found on the way. If we consider the set of all valid paths, (both upper and lower) get their corresponding strings, sort them all in alphabetical order, we obtain the (ordered) master set. Note that the master set might contain duplicates, and all strings in the master set consist of exactly 2N-1 characters. Let M be the total number of strings in the master set, given an integer I, we need to find the string with index = I (modulo M) within the master set.
If Master Set = { “A”,”B”,”B”,”C” } (although this set can never be a master set!)
I=0 produces “A”, while I=2 and I=5, produces “B”.

Constraints:
N<=30.
I<=10^{18}. ‘I’ will fit into a 64-bit integer.

Input

T-number of test cases
N I
Next is the NxN matrix of characters, N characters per line.
All characters are between ‘A’-‘Z’ (only uppercase).

Output

For each test case output the corresponding string sought for in the master set.

Example

Sample Input:

```
2
3 18
DAA
BDA
BBD
3 18
DAA
ADA
AAD
```

Sample Output:

```
DBBBD
DADAD
```

Explanation:
Test case I: Master Set = { “DAAAD”, “DADAD”,”DBBBD”,”DBDBD”}

Test case II: Master Set = { “DAAAD”,”DAAAD”,”DADAD”,”DADAD”}