Problem G

Playfair Cipher

Input: Standard Input
Output: Standard Output

The Playfair cipher was invented more than one and a half century ago by Sir Charles Wheatsone, one of the pioneers of the electric telegraph. It was made popular by Baron Playfair of St Andrews, a friend of Wheatstone, and used by British gouvernment institutions upto the first world war. The cipher's main advantages were that it is quite easy to use, both for encryption and decryption, and that it was relatively secure compared to other ciphers of that age. Nowadays, however, it would stand no chance against high speed computers, and that is what you are going to prove in this task.

To encrypt a message, the text is first converted to groups of two capitals, called digraphs. This is done as follows:

1. Convert all letters in the text to uppercase and omit all non-alphabetic characters.
2. Replace all letters 'J' by 'I'.
3. Form digraphs, but avoid having twice the same letter in a digraph. Insert an extra 'X' between the identical letters if necessary. If the repeated letter is an 'X', insert a 'Q' instead.
4. If the last digraph would be incomplete, append an extra 'X' to the text (or a 'Q' if the last letter in the text is an 'X').

Consider the following message: "Programming in C and Pascal is easy; I will learn Java next year." The digraph representation would be:

```
PR OG RA MX MI NG IN CA ND PA SC AL IS EA SY IW IL LX LE AR NI AV AN EX TY EA RX
```

Note the extra 'X' between the two 'M's of 'programming'. There is no extra 'X' between the two 'L's of 'will', because they are in different digraphs, but there is one between 'will' and 'learn'. There is also an extra 'X' at the end of the message. The 'J' in 'Java' is replaced by an 'I'. To illustrate the exceptions for the letter 'X' in the original text, consider the message "I am an ex-xenophobe, attempting to relax!". This becomes:

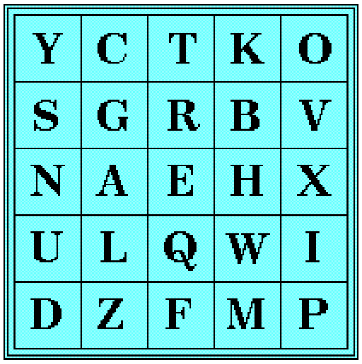
```
IA MA NE XQ XE NO PH OB EA TX TE MP TI NG TO RE LA XQ
```

The next stage is the replacement of each digraph by an other digraph according to the following rules:

- The uppercase letters, 'J' excluded, are placed in a 5X5 square in some predetermined order. This is the key for the encryption.
- If the two letters of the digraph are in the same row of the square, replace them by the letters occuring at the immediate right of each one. If one of the letters is in the rightmost column, replace it by the letter in the first column of the same row (wrap around).
- If the two letters of the digraph are in the same column of the square, replace them by the letters occuring immediately below each one. Wrap around to the same column in the top row if one of them is in the bottom row.
- If the letters are neither in the same row nor the same column, replace the first letter by the letter in the same row as the first letter and the same column as the second letter. Replace the second letter by the letter in the same row as the second letter and the same column as the first letter.

The resulting digraphs form the encrypted code.

Consider the key:



Y	C	T	K	O
S	G	R	B	V
N	A	E	H	X
U	L	Q	W	I
D	Z	F	M	P

The first digraph of the first example above is 'PR'. Since the letters are not in the same row or column, 'P' is replaced by 'F' and 'R' is replaced by 'V'. The replacement digraph is 'FV'.

Similarly the following digraphs, 'OG', 'RA', 'MX','MI', 'NG' and 'IN', are replaced by resp. 'CV', 'GE', 'PH', 'PW', 'AS' and 'UX'.

In the next digraph, 'CA', the letters occur in the same column, so 'C' is replaced by 'G' and 'A' is replaced by 'L', resulting in the new digraph 'GL'.

We also encounter the digraph 'EA' with letters in the same row. It's converted to 'HE'. Horizontal wrap around is encountered during the conversion of 'IW', 'IL' and 'EX'. No vertical wrap around occurs in the example, but the digraph 'BM' would convert to 'HK'.

The complete encryption of the first message is:

```
FV CV GE PH PW AS UX GL UY ZX GY LZ UV HE NS UI UQ IA QA EG XU XG EA HN KC HE VE
```

The second message encrypts to:

```
LX ZH AH EI NH XY MX KV HE OE RQ PD OQ AS KY EQ ZL EI
```

Decryption is easy once you know the key. That process is not described here, because I trust you can figure that out for yourself.

In this problem you will implement a so called 'known plaintext attack'. You have a piece of plain text and you also have it's encrypted code. From that information you will have to deduce a key and use that key to decode another piece of encrypted text.

Input

The input contains several cases, the number of which is on the first line. Every case has three parts. The first part is the plaintext and consists of one or more lines of ordinary text. The second part is the code that is the result of encrypting the first part. The third part is code for the text you are to decrypt. The parts are terminated by a hashmark ('#') on a line by itself. Code parts are printed as uppercase digraphs, 20 digraphs on a line, separated by one space. The last line of a code part can contain fewer than 20 digraphs. No code part will contain more than 5000 digraphs.

The keys can differ between cases (of course).

Output

For each case, first output a line "Case x:" where x is the case number (starting from 1). Then output the decrypted code represented as digraphs in the same format as the code parts in the input. Separate the cases by an empty line.

It is guaranteed that the first two parts of each case contain enough information to uniquely decode any possible encoded text.

Sample Input

```
2
Programming in C and Pascal is easy; I will learn Java next year.
#
FV CV GE PH PW AS UX GL UY ZX GY LZ UV HE NS UI UQ IA QA EG
XU XG EA HN KC HE VE
#
LX ZH AH EI NH XY MX KV HE OE RQ PD OQ AS KY EQ ZL EI
#
It is full moon!
Meet me at Hammersmith Bridge tonight.
#
MP PI NZ AZ RN QV UG GD DO GD RQ AR KY GD HD NK PR DA MS OG
UP GK IC QY
#
HL WT UP MC HQ RW PI CX DC ZD HB HG KL PM GI FP SK GE QR MF
MP AR BH HM HA SP DP TC WM DZ PO RL SG MU DC SB OD SM MU CS
UH RX BL MH HG WS DC BH MF KR MZ GT CD PU CS HD GH LK DP CT
GI RZ CD EV KY GD MF IP GT IF KG IC EH TE SD QV QG PR RQ EV
MU HK IF RC CR EQ OU PR SB GE CD PR PI UP DR UE EV FS BH MF
EV FS DA BC MK GI
#
```

Output for Sample Input

```
Case 1:
IA MA NE XQ XE NO PH OB EA TX TE MP TI NG TO RE LA XQ

Case 2:
CR YP TO GR AP HY IS AV ER YF AS CI NA TI NG SU BI EC TA ND
IT HA SA RI CH HI ST OR YI FY OU AR EI NT ER ES TE DI NT HE
PL AY FA IR CI PH ER SA ND MA NY MO RE OT HE RS IC AN ST RO
NG LY RE CO MX ME ND SI MO NS IN GH SC OD EB OX OK TH AT CO
NT AI NS AL LA BO UT TH ES EC RE TH IS TO RY OF CO DE SA ND
CO DE BR EA KI NG
```

Problemsetter: Joachim Wulff (aka Little Joey)
Special Thanks: Derek Kisman, EPS

EPILOGUE (Not required to solve the problem above)

The plaintext for the second example is:

Cryptography is a very fascinating subject and it has a rich history. If you are interested in the Playfair ciphers and many more others, I can strongly recommend Simon Singhs "Code Book" that contains all about the secret history of codes and code breaking.

The key was taken from Appendix E that contains the explanation of the Playfair cipher.
I owe my fascination for cryptography to Simon Singh.