Problem #4 - Matrix Encryption

Source File: **encrypt.{c|cpp|pas}**
Input File: **encrypt.in**
Output File: **encrypt.out**

A simple yet highly effective method of data encryption involves breaking a message into pieces of length n (the final piece is padded with blanks if necessary), where n is a positive integer, and then multiplying each piece, as a column vector, by an $n \times n$ non-singular integer matrix (there are a few important restrictions on the formulation of the matrix, but these are not important for this problem). The resulting column vectors form the encoded message. Since it is possible that the resulting column vectors will contain values not representable in the message alphabet, each element is mapped to its equivalent value (mod)k, where k is the number of characters in the message alphabet. (The decryption process is similar but involves the inverse of the encryption matrix.)

For this problem, consider messages which contain only uppercase letters, spaces, and three punctuation characters: commas, periods, and question marks. Number the letters A...Z as $1 \ldots 26$, respectively, the space 27, the comma 28, the period 29, and the question mark 30. Now consider encrypting the message "SEND HELP." using the encryption matrix

$$M = \begin{bmatrix} 1 & 1 & 1 \\ 0 & -1 & 2 \\ 1 & 3 & -5 \end{bmatrix}.$$

Since M is a 3×3 matrix, the message will be encoded three characters at a time. The first three characters are "SEN", represented by the numbers 19 5 14 according to the above scheme. Treating these numbers as a column vector, we compute

$$M \begin{bmatrix} 19 \\ 5 \\ 14 \end{bmatrix} = \begin{bmatrix} 38 \\ 23 \\ -36 \end{bmatrix}.$$

To map the elements of the resulting column vector back to the range $1 \ldots 30$, 30 is repeatedly added or subtracted to or from each element as necessary until each element is in the proper range (this is a slightly modified form of the modulus operation). Thus, the resulting column vector above becomes 8 23 24, which represents the string "HWX".

Repeating this process with the next three characters in the message, "D H", yields an encrypted string of "ISO". Repeating again with "ELP" yields "CTU". Finally, we take the final character (a period) along with enough blanks to pad to three characters, and perform the final encoding. So, encoding ". " (a period followed by 2 spaces) yields "W E". So, the final encrypted message is "HWXISOCTUW E".

The input file for this problem consists of several messages to be encrypted as just demonstrated. The first line of the input file is the dimension d of the matrix which will be used to encrypt the first message (the maximum dimension which will be used in the input file is 10). The next d lines are the rows of the matrix, and the line immediately following the matrix is the message to be encrypted (maximum length is 80 characters). Following this is another matrix and message, and then another, and so on. The end of the input is indicated by a 0 for the dimension of the encryption matrix. The output file should consist of the encrypted text of each message, one message per line, enclosed in single quotes.

Sample input file:

```
3
1 1 1
0 -1 2
1 3 -5
SEND HELP.
5
1 2 3 4 5
2 1 4 3 5
-1 0 5 3 1
0 2 -3 5 1
5 4 3 0 -2
TO BE, OR NOT TO BE?  THAT IS THE QUESTION.
1
19
ACM REGIONAL PROGRAMMING CONTEST
0
```

The output for the above sample input should be:

```
'HWXISOCTUW E'
'NNFXUDBHF.LDGE?ETJMI,JHEALYS.ADXMCWCRWBNMP,MN'
'S GCLEMUOZSRCDLOMLSGGUZMC OZTEAT'
```