1031 Insecure in Prague

Prague is a dangerous city for developers of cryptographic schemes. In 2001, a pair of researchers in Prague announced a security flaw in the famous PGP encryption protocol. In Prague in 2003 , a flaw was discovered in the SSL/TLS (Secure Sockets Layer and Transport Layer Security) protocols. However, Prague's reputation for being tough on cryptographic protocols hasn't stopped the part-time amateur cryptographer and full-time nutcase, Immanuel Kant-DeWitt (known to his friends as "I. Kant-DeWitt"), from bringing his latest encryption scheme to Prague. Here's how it works:

A plain text message p of length n is to be transmitted. The sender chooses an integer $m \geq 2n$, and integers s, t, i, and j, where $0 \leq s, t, i, j < m$ and $i < j$. The scheme works as follows: m is the length of the transmitted ciphertext string, c. Initially, c contains m empty slots. The first letter of p is placed in position s of c. The k-th letter, $k \geq 2$, is placed by skipping over i empty slots in c after the $(k-1)$-st letter, wrapping around to the beginning of c if necessary. Slots already containing letters are not counted as empty. For instance, if the message is PRAGUE, if $s = 1$, $i = 6$, and $m = 15$, then the letters are placed in c as follows:

A	P		U					R	G			E		
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Starting with the first empty slot in or after position t in string c, the plain text message is entered again, but this time skipping j empty slots between letters. For instance, if $t = 0$ and $j = 8$, the second copy of p is entered as follows (beginning in position 2, the first empty slot starting from $t = 0$):

A	P	**P**	U	**R**		**A**	**U**	R	G	**E**	**G**	E		
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Finally, any remaining unfilled slots in c are filled in with randomly chosen letters:

A	P	P	U	R	**A**	A	U	R	G	E	G	E	**W**	**E**
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Kant-DeWitt believes that the duplication of the message, combined with the use of random letters, will confuse decryption schemes based upon letter frequencies and that, without knowledge of s and i, no one can figure out what the original message is. Your job is to try to prove him wrong. Given a number of ciphertext strings (and no additional information), you will determine the longest possible message that could have been encoded using the Kant-DeWitt method.

Input

A number of ciphertext strings, one per line. Each string will consist only of upper case alphabetic letters, with no leading or trailing blanks; each will have length between 2 and 40.

Input for the last test case is followed by a line consisting of the letter '`X`'.

Output

For each input ciphertext string, print the longest string that could be encrypted in the ciphertext. If more than one string has the longest length, then print '`Codeword not unique`'. Follow the format of the sample output given below.

Sample Input

```
APPURAAURGEGEWE
ABABABAB
THEACMPROGRAMMINGCONTEST
X
```

Sample Output

```
Code 1: PRAGUE
Code 2: Codeword not unique
Code 3: Codeword not unique
```