The aim of data compression is to reduce redundancy in stored or communicated data. This increases effective data density and speeds up data transfer rates. One possible method to compress any binary message is the following:

> Replace any maximal sequence of n 1's with the binary version of n whenever it shortens the length of the message.

For example, the compressed form of the data **11111111**00100**1111111111111111**0011 becomes **1000**00100**1111**0011. The original data is 32 bits long while the compressed data is only 17 bits long.

The drawback of this method is that sometimes the decompression process yields more than one result for the original message, making it impossible to obtain the original message. Write a program that determines if the original message can be obtained from the compressed data when the length of the original message (L), the number of 1's in the original message (N) and the compressed data are given. The original message will be no longer than 16 Kbytes and the compressed data will be no longer than 40 bits.

Input

The input file contains several test cases. Each test case has two lines. The first line contains L and N and the second line contains the compressed data.

The last case is followed by a line containing two zeroes.

Output

For each test case, output a line containing the case number (starting with 1) and a message '`YES`', '`NO`' or '`NOT UNIQUE`'. '`YES`' means that the original message can be obtained. '`NO`' means that the compressed data has been corrupted and the original message cannot be obtained. '`NOT UNIQUE`' means that more than one message could have been the original message. Follow the format shown in the sample output.

Sample Input

```
32 26
10000010011110011
9 7
1010101
14 14
111111
00
```

Sample Output

```
Case #1: YES
Case #2: NOT UNIQUE
Case #3: NO
```