The following is an excerpt from section 6.8 (Base64 Content-Transfer-Encoding) of RFC 2045. It gives details of Base64 encoding that is used for encoding of binary attchments in email.

The encoding process represents 24-bit groups of input bits as output strings of 4 encoded characters. Proceeding from left to right, a 24-bit input group is formed by concatenating 3 8bit input groups. These 24 bits are then treated as 4 concatenated 6-bit groups, each of which is translated into a single digit in the base64 alphabet. When encoding a bit stream via the base64 encoding, the bit stream must be presumed to be ordered with the most-significant-bit first. That is, the first bit in the stream will be the high-order bit in the first 8bit byte, and the eighth bit will be the low-order bit in the first 8bit byte, and so on.

Each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the output string. These characters are identified in table below.

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	26	a	52	0	62	+
1	B	27	b	53	1	63	/
2	C	28	c	54	2		
.	.	.	.	.	.	(pad)	=
.	.	.	.	.	.		
24	Y	50	y	60	8		
25	Z	51	z	61	9		

The encoded output stream must be represented in lines of no more than 76 characters each. All line breaks or other characters not found in the table must be ignored by decoding software.

Special processing is performed if fewer than 24 bits are available at the end of the data being encoded. A full encoding quantum is always completed at the end of a body. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups. Padding at the end of the data is performed using the '=' character. Since all base64 input is an integral number of octets, only the following cases can arise:

1. the final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of encoded output will be an integral multiple of 4 characters with no '=' padding,

2. the final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two '=' padding characters,

3. or the final quantum of encoding input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one '=' padding character.

Because it is used only for padding at the end of the data, the occurrence of any '=' characters may be taken as evidence that the end of the data has been reached (without truncation in transit). No such assurance is possible, however, when the number of octets transmitted was a multiple of three and no '=' characters are present.

You are to write a Base64 decoder for a web-based email client to be used in BUET server. However, in this problem, to ease handing multiple datasets, you have to treat the character '#' specially (i.e. NOT ignore it as RFC 2045 says above). See input specification below.

Input

There are multiple datasets, each terminated by the '#' mark. The last dataset, which should not be processed, contains no character other than the terminating '#' mark. Each dataset consists of a valid Base64 encoding of some (possibly binary) data.

Output

Output decoded data for each dataset followed immediately by the '#' mark.

Sample Input

```
VGhpcOlzVGVzdA==
#
QSBUZXNOIElucHV0W3so
KX1d
##
```

Sample Output

```
ThisIsTest#A Test Input[{()}]#
```