Coding

A binary code for an alphabet of 2^N symbols is a bijection between the 2^N symbols and 2^N binary codewords. For example, in the table below 3 different binary codes are presented for a 4-symbol alphabet (**a**,**b**,**c**,**d**).

Symbol	Code 1	Code 2	Code 3
a	00	0	1
b	01	10	10
c	10	110	100
d	11	111	1000

A code is said to be prefix-free if none of the codewords is a prefix of another codeword. For example, in the table above, codes 1 and 2 are prefix-free. However, code 3 is not prefix-free. Prefix-free codes are widely used, as encoding and decoding becomes very simple.
For this problem, given **N** and a message containing **M** alphabet symbols, the task is to find a prefix-free code for the entire alphabet (including symbols possibly not present in the message) that minimizes the number of necessary bits to represent the message. For example, let **N**=2, with symbols (a,b,c,d), and the message "**a a a a b b b b a a a a c c d d**"

The message encoded with codes 1 and 2 above becomes, respectively:

- 00 00 00 00 01 01 01 01 00 00 00 00 10 10 11 11, for a total of 32 bits.
- 0 0 0 0 10 10 10 10 0 0 0 0 110 110 111 111, for a total of 28 bits.

It is possible to show that no prefix-free code can encode the message above in less than 28 bits.

Input

The input contains several test cases. Each test case has two lines. The first line of a test case contains two integers **N**, **M** separated by a single space (1 ≤ **N** ≤ 15, 1 ≤ **M** ≤ 106, **D** ≤ 15).

On the second line are **M** integers X_i, 0 ≤ **Xi** ≤ 2^N – 1, representing the message to be encoded. The end of the input is marked by a case with **N**=**M**=0. This case must not be processed.

Output

For each test case, print a single line with one integer, the minimum number of bits necessary to encode the message using a prefix-free code.

Example

Input:
```
2 16
0 0 0 0 1 1 1 1 0 0 0 0 2 2 3 3
0 0
```

Output:
```
28
```