E. Bits of merry old England

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Another feature of Shakespeare language is that the variables are named after characters of plays by Shakespeare, and all operations on them (value assignment, output etc.) look like a dialog with other characters. New values of variables are defined in a rather lengthy way, so a programmer should try to minimize their usage.

You have to print the given sequence of n integers. To do this, you have m variables and two types of operations on them:

- `variable=integer`
- `print(variable)`

Any of the m variables can be used as `variable`. Variables are denoted by lowercase letters between "`a`" and "`z`", inclusive. Any integer number can be used as `integer`.

Let's say that the penalty for using first type of operations equals to the number of set bits in the number `integer`. There is no penalty on using second type of operations. Find and output the program which minimizes the penalty for printing the given sequence of numbers.

Input

The first line of input contains integers n and m ($1 \le n \le 250$, $1 \le m \le 26$). The second line contains the sequence to be printed. Each element of the sequence is an integer between 1 and 10^9, inclusive. The sequence has to be printed in the given order (from left to right).

Output

Output the number of lines in the optimal program and the optimal penalty. Next, output the program itself, one command per line. If there are several programs with minimal penalty, output any of them (you have only to minimize the penalty).

Examples

input

```
7 2
1 2 2 4 2 1 2
```

output

```
11 4
b=1
print(b)
a=2
print(a)
print(a)
b=4
print(b)
print(a)
b=1
print(b)
print(a)
```

input

```
6 3
1 2 3 1 2 3
```

output

```
9 4
c=1
print(c)
b=2
print(b)
a=3
print(a)
print(c)
print(b)
print(a)
```