Educational Codeforces Round 9

A. Grandma Laura and Apples

1 second, 256 megabytes

Grandma Laura came to the market to sell some apples. During the day she sold all the apples she had. But grandma is old, so she forgot how many apples she had brought to the market.

She precisely remembers she had n buyers and each of them bought exactly half of the apples she had at the moment of the purchase and also she gave a half of an apple to some of them as a gift (if the number of apples at the moment of purchase was odd), until she sold all the apples she had.

So each buyer took some integral positive number of apples, but maybe he didn't pay for a half of an apple (if the number of apples at the moment of the purchase was odd).

For each buyer grandma remembers if she gave a half of an apple as a gift or not. The cost of an apple is p (the number p is even).

Print the total money grandma should have at the end of the day to check if some buyers cheated her.

Input

The first line contains two integers n and p $(1 \leq n \leq 40, 2 \leq p \leq 1000)$ — the number of the buyers and the cost of one apple. It is guaranteed that the number p is even.

The next n lines contains the description of buyers. Each buyer is described with the string `half` if he simply bought half of the apples and with the string `halfplus` if grandma also gave him a half of an apple as a gift.

It is guaranteed that grandma has at least one apple at the start of the day and she has no apples at the end of the day.

Output

Print the only integer a — the total money grandma should have at the end of the day.

Note that the answer can be too large, so you should use 64-bit integer type to store it. In `C++` you can use the `long long` integer type and in `Java` you can use `long` integer type.

input

```
2 10
half
halfplus
```

output

```
15
```

input

```
3 10
halfplus
halfplus
halfplus
```

output

```
55
```

In the first sample at the start of the day the grandma had two apples. First she sold one apple and then she sold a half of the second apple and gave a half of the second apple as a present to the second buyer.

B. Alice, Bob, Two Teams

1.5 seconds, 256 megabytes

Alice and Bob are playing a game. The game involves splitting up game pieces into two teams. There are n pieces, and the i-th piece has a strength p_i.

The way to split up game pieces is split into several steps:

1. First, Alice will split the pieces into two different groups A and B. This can be seen as writing the assignment of teams of a piece in an n character string, where each character is A or B.
2. Bob will then choose an arbitrary prefix or suffix of the string, and flip each character in that suffix (i.e. change A to B and B to A). He can do this step at most once.
3. Alice will get all the pieces marked A and Bob will get all the pieces marked B.

The strength of a player is then the sum of strengths of the pieces in the group.

Given Alice's initial split into two teams, help Bob determine an optimal strategy. Return the maximum strength he can achieve.

Input

The first line contains integer n $(1 \le n \le 5 \cdot 10^5)$ — the number of game pieces.

The second line contains n integers p_i $(1 \le p_i \le 10^9)$ — the strength of the i-th piece.

The third line contains n characters A or B — the assignment of teams after the first step (after Alice's step).

Output

Print the only integer a — the maximum strength Bob can achieve.

input

```
5
1 2 3 4 5
ABABA
```

output

```
11
```

input

```
5
1 2 3 4 5
AAAAA
```

output

```
15
```

input

```
1
1
B
```

output

```
1
```

In the first sample Bob should flip the suffix of length one.

In the second sample Bob should flip the prefix or the suffix (here it is the same) of length 5.

In the third sample Bob should do nothing.

C. The Smallest String Concatenation

3 seconds, 256 megabytes

You're given a list of n strings $a_1, a_2, ..., a_n$. You'd like to concatenate them together in some order such that the resulting string would be lexicographically smallest.

Given the list of strings, output the lexicographically smallest concatenation.

Input

The first line contains integer n — the number of strings $(1 \le n \le 5 \cdot 10^4)$.

Each of the next n lines contains one string a_i $(1 \le |a_i| \le 50)$ consisting of only lowercase English letters. The sum of string lengths will not exceed $5 \cdot 10^4$.

Output

Print the only string a — the lexicographically smallest string concatenation.

input

```
4
abba
abacaba
bcd
er
```

output

```
abacabaabbabcder
```

input

```
5
x
xx
xxa
xxaa
xxaaa
```

output

```
xxaaaxxaaxxaxxx
```

input

```
3
c
cb
cba
```

output

```
cbacbc
```

D. Longest Subsequence

2 seconds, 256 megabytes

You are given array a with n elements and the number m. Consider some subsequence of a and the value of least common multiple (LCM) of its elements. Denote LCM as l. Find any longest subsequence of a with the value $l \le m$.

A subsequence of a is an array we can get by erasing some elements of a. It is allowed to erase zero or all elements.

The LCM of an empty array equals 1.

Input

The first line contains two integers n and m $(1 \le n, m \le 10^6)$ — the size of the array a and the parameter from the problem statement.

The second line contains n integers a_i $(1 \le a_i \le 10^9)$ — the elements of a.

Output

In the first line print two integers l and k_{max} $(1 \le l \le m, 0 \le k_{max} \le n)$ — the value of LCM and the number of elements in optimal subsequence.

In the second line print k_{max} integers — the positions of the elements from the optimal subsequence in the ascending order.

Note that you can find and print any subsequence with the maximum length.

input

```
7 8
6 2 9 2 7 2 3
```

output

```
6 5
1 2 4 6 7
```

input

```
6 4
2 2 2 3 3 3
```

output

```
2 3
1 2 3
```

E. Thief in a Shop

5 seconds, 512 megabytes

A thief made his way to a shop.

As usual he has his lucky knapsack with him. The knapsack can contain k objects. There are n kinds of products in the shop and an infinite number of products of each kind. The cost of one product of kind i is a_i.

The thief is greedy, so he will take exactly k products (it's possible for some kinds to take several products of that kind).

Find all the possible total costs of products the thief can nick into his knapsack.

Input

The first line contains two integers n and k $(1 \le n, k \le 1000)$ — the number of kinds of products and the number of products the thief will take.

The second line contains n integers a_i $(1 \le a_i \le 1000)$ — the costs of products for kinds from 1 to n.

Output

Print the only line with all the possible total costs of stolen products, separated by a space. The numbers should be printed in the ascending order.

input

```
3 2
1 2 3
```

output

```
2 3 4 5 6
```

input

```
5 5
1 1 1 1 1
```

output

```
5
```

input

```
3 3
3 5 11
```

output

```
9 11 13 15 17 19 21 25 27 33
```

F. Magic Matrix

5 seconds, 512 megabytes

You're given a matrix A of size $n \times n$.

Let's call the matrix with nonnegative elements magic if it is symmetric (so $a_{ij} = a_{ji}$), $a_{ii} = 0$ and $a_{ij} \le max(a_{ik}, a_{jk})$ for all triples i, j, k. Note that i, j, k do not need to be distinct.

Determine if the matrix is magic.

As the input/output can reach very huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in `Java`.

Input

The first line contains integer n $(1 \le n \le 2500)$ — the size of the matrix A.

Each of the next n lines contains n integers a_{ij} $(0 \le a_{ij} < 10^9)$ — the elements of the matrix A.

Note that the given matrix not necessarily is symmetric and can be arbitrary.

Output

Print "`MAGIC`" (without quotes) if the given matrix A is magic. Otherwise print "`NOT MAGIC`".

input

```
3
0 1 2
1 0 2
2 2 0
```

output

```
MAGIC
```

input

```
2
0 1
2 3
```

output

```
NOT MAGIC
```

input

```
4
0 1 2 3
1 0 3 4
2 3 0 5
3 4 5 0
```

output

```
NOT MAGIC
```