Problem Set 13 15-295 Spring 2019

A. Equivalent Strings **1 Point**

2 seconds, 256 megabytes

Today on a lecture about strings Gerald learned a new definition of string equivalency. Two strings a and b of equal length are called *equivalent* in one of the two cases:

1. They are equal.
2. If we split string a into two halves of the same size a_1 and a_2, and string b into two halves of the same size b_1 and b_2, then one of the following is correct:
   a. a_1 is equivalent to b_1, and a_2 is equivalent to b_2
   b. a_1 is equivalent to b_2, and a_2 is equivalent to b_1

As a home task, the teacher gave two strings to his students and asked to determine if they are equivalent.

Gerald has already completed this home task. Now it's your turn!

Input

The first two lines of the input contain two strings given by the teacher. Each of them has the length from 1 to $200\,000$ and consists of lowercase English letters. The strings have the same length.

Output

Print "`YES`" (without the quotes), if these two strings are equivalent, and "`NO`" (without the quotes) otherwise.

input

```
aaba
abaa
```

output

```
YES
```

input

```
aabb
abab
```

output

```
NO
```

In the first sample you should split the first string into strings "`aa`" and "`ba`", the second one — into strings "`ab`" and "`aa`". "`aa`" is equivalent to "`aa`"; "`ab`" is equivalent to "`ba`" as "`ab`" = "`a`" + "`b`", "`ba`" = "`b`" + "`a`".

In the second sample the first string can be splitted into strings "`aa`" and "`bb`", that are equivalent only to themselves. That's why string "`aabb`" is equivalent only to itself and to string "`bbaa`".

B. Suspects **1.5 Points**

2 seconds, 256 megabytes

As Sherlock Holmes was investigating a crime, he identified n suspects. He knows for sure that exactly one of them committed the crime. To find out which one did it, the detective lines up the suspects and numbered them from 1 to n. After that, he asked each one: "Which one committed the crime?". Suspect number i answered either "The crime was committed by suspect number a_i", or "Suspect number a_i didn't commit the crime". Also, the suspect could say so about himself ($a_i = i$).

Sherlock Holmes understood for sure that exactly m answers were the truth and all other answers were a lie. Now help him understand this: which suspect lied and which one told the truth?

Input

The first line contains two integers n and m ($1 \le n \le 10^5, 0 \le m \le n$) — the total number of suspects and the number of suspects who told the truth. Next n lines contain the suspects' answers. The i-th line contains either "+a_i" (without the quotes), if the suspect number i says that the crime was committed by suspect number a_i, or "-a_i" (without the quotes), if the suspect number i says that the suspect number a_i didn't commit the crime (a_i is an integer, $1 \le a_i \le n$).

It is guaranteed that at least one suspect exists, such that if he committed the crime, then exactly m people told the truth.

Output

Print n lines. Line number i should contain "`Truth`" if suspect number i has told the truth for sure. Print "`Lie`" if the suspect number i lied for sure and print "`Not defined`" if he could lie and could tell the truth, too, depending on who committed the crime.

input
1 1 +1
output
Truth

input
3 2 -1 -2 -3
output
Not defined Not defined Not defined

input
4 1 +2 -3 +4 -1
output
Lie Not defined Lie Not defined

The first sample has the single person and he confesses to the crime, and Sherlock Holmes knows that one person is telling the truth. That means that this person is telling the truth.

In the second sample there are three suspects and each one denies his guilt. Sherlock Holmes knows that only two of them are telling the truth. Any one of them can be the criminal, so we don't know for any of them, whether this person is telling the truth or not.

In the third sample the second and the fourth suspect defend the first and the third one. But only one is telling the truth, thus, the first or the third one is the criminal. Both of them can be criminals, so the second and the fourth one can either be lying or telling the truth. The first and the third one are lying for sure as they are blaming the second and the fourth one.

C. Double Profiles **2 Points**

3 seconds, 256 megabytes

You have been offered a job in a company developing a large social network. Your first task is connected with searching profiles that most probably belong to the same user.

The social network contains n registered profiles, numbered from 1 to n. Some pairs there are friends (the "friendship" relationship is mutual, that is, if i is friends with j, then j is also friends with i). Let's say that profiles i and j ($i \neq j$) are *doubles*, if for any profile k ($k \neq i$, $k \neq j$) one of the two statements is true: either k is friends with i and j, or k isn't friends with either of them. Also, i and j can be friends or not be friends.

Your task is to count the number of different unordered pairs (i, j), such that the profiles i and j are doubles. Note that the pairs are unordered, that is, pairs (a, b) and (b, a) are considered identical.

Input

The first line contains two space-separated integers n and m ($1 \le n \le 10^6$, $0 \le m \le 10^6$), — the number of profiles and the number of pairs of friends, correspondingly.

Next m lines contains descriptions of pairs of friends in the format "v u", where v and u ($1 \le v, u \le n$, $v \neq u$) are numbers of profiles that are friends with each other. It is guaranteed that each unordered pair of friends occurs no more than once and no profile is friends with itself.

Output

Print the single integer — the number of unordered pairs of profiles that are doubles.

Please do not use the `%lld` specificator to read or write 64-bit integers in C++. It is preferred to use the `%I64d` specificator.

input

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

output

```
3
```

input

```
3 0
```

output

```
3
```

input

```
4 1
1 3
```

output

```
2
```

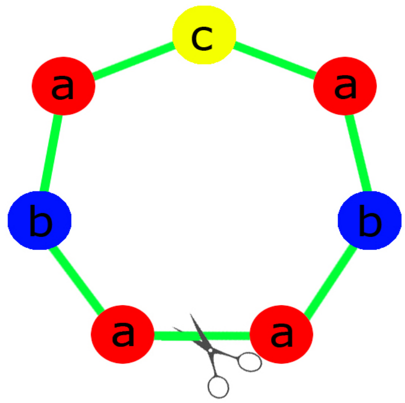
In the first and second sample any two profiles are doubles.

In the third sample the doubles are pairs of profiles $(1, 3)$ and $(2, 4)$.

D. Necklace **2.5 Points**

2 seconds, 256 megabytes

Ivan wants to make a necklace as a present to his beloved girl. A *necklace* is a cyclic sequence of beads of different colors. Ivan says that necklace is *beautiful* relative to the cut point between two adjacent beads, if the chain of beads remaining after this cut is a palindrome (reads the same forward and backward).



Ivan has beads of n colors. He wants to make a necklace, such that it's beautiful relative to as many cuts as possible. He certainly wants to use all the beads. Help him to make the most beautiful necklace.

Input

The first line of the input contains a single number n $(1 \le n \le 26)$ — the number of colors of beads. The second line contains after n positive integers a_i — the quantity of beads of i-th color. It is guaranteed that the sum of a_i is at least 2 and does not exceed $100\,000$.

Output

In the first line print a single number — the maximum number of beautiful cuts that a necklace composed from given beads may have. In the second line print any example of such necklace.

Each color of the beads should be represented by the corresponding lowercase English letter (starting with `a`). As the necklace is cyclic, print it starting from any point.

input
```
3
4 2 1
```
output
```
1
abacaba
```

input
```
1
4
```
output
```
4
aaaa
```

input
```
2
1 1
```
output
```
0
ab
```

In the first sample a necklace can have at most one beautiful cut. The example of such a necklace is shown on the picture.

In the second sample there is only one way to compose a necklace.

E. Tournament Construction **3 Points**

2 seconds, 512 megabytes

Ivan is reading a book about tournaments. He knows that a tournament is an oriented graph with exactly one oriented edge between each pair of vertices. The score of a vertex is the number of edges going outside this vertex.

Yesterday Ivan learned Landau's criterion: there is tournament with scores $d_1 \le d_2 \le \ldots \le d_n$ if and only if $d_1 + \cdots + d_k \geqslant \frac{k(k-1)}{2}$ for all $1 \le k < n$ and $d_1 + d_2 + \cdots + d_n = \frac{n(n-1)}{2}$.

Now, Ivan wanna solve following problem: given a **set** of numbers $S = \{a_1, a_2, \ldots, a_m\}$, is there a tournament with given set of scores? I.e. is there tournament with sequence of scores $d_1, d_2, \ldots, d_n$ such that if we remove duplicates in scores, we obtain the required set $\{a_1, a_2, \ldots, a_m\}$?

Find a tournament with **minimum** possible number of vertices.

Input

The first line contains a single integer m ($1 \le m \le 31$).

The next line contains m distinct integers $a_1, a_2, \ldots, a_m$ ($0 \le a_i \le 30$) — elements of the set S. It is guaranteed that all elements of the set are distinct.

Output

If there are no such tournaments, print string "=(" (without quotes).

Otherwise, print an integer n — the number of vertices in the tournament.

Then print n lines with n characters — matrix of the tournament. The j-th element in the i-th row should be 1 if the edge between the i-th and the j-th vertices is oriented towards the j-th vertex, and 0 otherwise. The main diagonal should contain only zeros.

input

```
2
1 2
```

output

```
4
0011
1001
0100
0010
```

input

```
2
0 3
```

output

```
6
000111
100011
110001
011001
001101
000000
```

F. Memory and Casinos **3 Points**

4 seconds, 512 megabytes

There are n casinos lined in a row. If Memory plays at casino i, he has probability p_i to win and move to the casino on the right ($i + 1$) or exit the row (if $i = n$), and a probability $1 - p_i$ to lose and move to the casino on the left ($i - 1$) or also exit the row (if $i = 1$).

We say that Memory *dominates* on the interval $i \dots j$ if he completes a walk such that,

- He starts on casino i.
- He never looses in casino i.
- He finishes his walk by winning in casino j.

Note that Memory can still walk left of the 1-st casino and right of the casino n and that always finishes the process.

Now Memory has some requests, in one of the following forms:

- $1\ i\ a\ b$: Set $p_i = \frac{a}{b}$.
- $2\ l\ r$: Print the probability that Memory will *dominate* on the interval $l \dots r$, i.e. compute the probability that Memory will first leave the segment $l \dots r$ after winning at casino r, if she starts in casino l.

It is guaranteed that at any moment of time p is a **non-decreasing sequence**, i.e. $p_i \le p_{i+1}$ for all i from 1 to $n - 1$.

Please help Memory by answering all his requests!

Input

The first line of the input contains two integers n and $q (1 \le n, q \le 100\ 000)$, — number of casinos and number of requests respectively.

The next n lines each contain integers a_i and b_i ($1 \le a_i < b_i \le 10^9$) — $\frac{a_i}{b_i}$ is the probability p_i of winning in casino i.

The next q lines each contain queries of one of the types specified above ($1 \le a < b \le 10^9$, $1 \le i \le n$, $1 \le l \le r \le n$).

It's guaranteed that there will be at least one query of type 2, i.e. the output will be non-empty. Additionally, it is guaranteed that p forms a non-decreasing sequence at all times.

Output

Print a real number for every request of type 2 — the probability that boy will "dominate" on that interval. Your answer will be considered correct if its absolute error does not exceed 10^{-4}.

Namely: let's assume that one of your answers is a, and the corresponding answer of the jury is b. The checker program will consider your answer correct if $|a - b| \le 10^{-4}$.

input

```
3 13
1 3
1 2
2 3
2 1 1
2 1 2
2 1 3
2 2 2
2 2 3
2 3 3
1 2 2 3
2 1 1
2 1 2
2 1 3
2 2 2
2 2 3
2 3 3
```

output

```
0.3333333333
0.2000000000
0.1666666667
0.5000000000
0.4000000000
0.6666666667
0.3333333333
0.2500000000
0.2222222222
0.6666666667
0.5714285714
0.6666666667
```