Codeforces Beta Round #89 (Div. 2)

A. String Task

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

Petya started to attend programming lessons. On the first lesson his task was to write a simple program. The program was supposed to do the following: in the given string, consisting if uppercase and lowercase Latin letters, it:

- deletes all the vowels,
- inserts a character "`.`" before each consonant,
- replaces all uppercase consonants with corresponding lowercase ones.

Vowels are letters "A", "O", "Y", "E", "U", "I", and the rest are consonants. The program's input is exactly one string, it should return the output as a single string, resulting after the program's processing the initial string.

Help Petya cope with this easy task.

Input

The first line represents input string of Petya's program. This string only consists of uppercase and lowercase Latin letters and its length is from 1 to 100, inclusive.

Output

Print the resulting string. It is guaranteed that this string is not empty.

Sample test(s)

input
tour
output
.t.r

input
Codeforces
output
.c.d.f.r.c.s

input
aBAcAba
output
.b.c.b

B. Present from Lena

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

Vasya's birthday is approaching and Lena decided to sew a patterned handkerchief to him as a present. Lena chose digits from 0 to n as the pattern. The digits will form a rhombus. The largest digit n should be located in the centre. The digits should decrease as they approach the edges. For example, for $n = 5$ the handkerchief pattern should look like that:

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

Your task is to determine the way the handkerchief will look like by the given n.

Input

The first line contains the single integer n ($2 \le n \le 9$).

Output

Print a picture for the given n. You should strictly observe the number of spaces before the first digit on each line. Every two adjacent digits in the same line should be separated by exactly one space. There should be no spaces after the last digit at the end of each line.

Sample test(s)

input
2

output
```
    0
  0 1 0
0 1 2 1 0
  0 1 0
    0
```
|

input
3

output
```
      0
    0 1 0
  0 1 2 1 0
0 1 2 3 2 1 0
  0 1 2 1 0
    0 1 0
      0
```
|

C. Fancy Number

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

A car number in Berland consists of exactly n digits. A number is called beautiful if it has at least k equal digits. Vasya wants to change the digits in his car's number so that the number became beautiful. To replace one of n digits Vasya has to pay the sum of money, equal to the absolute difference between the old digit and the new one.

Help Vasya: find the minimum sum of money he should pay to make the number of his car beautiful. You should also find the resulting beautiful number. If there are several such numbers, then print the lexicographically minimum one.

Input

The first line contains two space-separated integers n and k ($2 \le n \le 10^4, 2 \le k \le n$) which represent how many digits the number has and how many equal digits a beautiful number should have. The second line consists of n digits. It describes the old number of Vasya's car. It is guaranteed that the number contains no spaces and only contains digits.

Output

On the first line print the minimum sum of money Vasya needs to change the number. On the second line print the car's new number. If there are several solutions, print the lexicographically minimum one.

Sample test(s)

input
6 5 898196
output
4 888188

input
3 2 533
output
0 533

input
10 6 0001112223
output
3 0000002223

Note

In the first sample replacing the second digit with an "8" costs $|9 - 8| = 1$. Replacing the fifth digit with an "8" costs the same. Replacing the sixth digit costs $|6 - 8| = 2$. As a result, Vasya will pay $1 + 1 + 2 = 4$ for a beautiful number "888188".

The lexicographical comparison of strings is performed by the < operator in modern programming languages. The string x is lexicographically smaller than the string y, if there exists such i ($1 \le i \le n$), that $x_i < y_i$, and for any j ($1 \le j < i$) $x_j = y_j$. The strings compared in this problem will always have the length n.

D. Caesar's Legions

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

Gaius Julius Caesar, a famous general, loved to line up his soldiers. Overall the army had n_1 footmen and n_2 horsemen. Caesar thought that an arrangement is **not** beautiful if somewhere in the line there are strictly more that k_1 footmen standing successively one after another, or there are strictly more than k_2 horsemen standing successively one after another. Find the number of *beautiful* arrangements of the soldiers.

Note that all $n_1 + n_2$ warriors should be present at each arrangement. All footmen are considered indistinguishable among themselves. Similarly, all horsemen are considered indistinguishable among themselves.

Input

The only line contains four space-separated integers n_1, n_2, k_1, k_2 ($1 \le n_1, n_2 \le 100, 1 \le k_1, k_2 \le 10$) which represent how many footmen and horsemen there are and the largest acceptable number of footmen and horsemen standing in succession, correspondingly.

Output

Print the number of beautiful arrangements of the army modulo 100000000 (10^8). That is, print the number of such ways to line up the soldiers, that no more than k_1 footmen stand successively, and no more than k_2 horsemen stand successively.

Sample test(s)

input
2 1 1 10
output
1

input
2 3 1 2
output
5

input
2 4 1 1
output
0

Note

Let's mark a footman as `1`, and a horseman as `2`.

In the first sample the only beautiful line-up is: `121`

In the second sample 5 beautiful line-ups exist: `12122`, `12212`, `21212`, `21221`, `22121`

E. Bertown roads

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bertown has n junctions and m bidirectional roads. We know that one can get from any junction to any other one by the existing roads.

As there were more and more cars in the city, traffic jams started to pose real problems. To deal with them the government decided to make the traffic one-directional on all the roads, thus easing down the traffic. Your task is to determine whether there is a way to make the traffic one-directional so that there still is the possibility to get from any junction to any other one. If the answer is positive, you should also find one of the possible ways to orient the roads.

Input

The first line contains two space-separated integers n and m ($2 \le n \le 10^5$, $n - 1 \le m \le 3 \cdot 10^5$) which represent the number of junctions and the roads in the town correspondingly. Then follow m lines, each containing two numbers which describe the roads in the city. Each road is determined by two integers a_i and b_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$) — the numbers of junctions it connects.

It is guaranteed that one can get from any junction to any other one along the existing bidirectional roads. Each road connects different junctions, there is no more than one road between each pair of junctions.

Output

If there's no solution, print the single number 0. Otherwise, print m lines each containing two integers p_i and q_i — each road's orientation. That is the traffic flow will move along a one-directional road from junction p_i to junction q_i. You can print the roads in any order. If there are several solutions to that problem, print any of them.

Sample test(s)

input

```
6 8
1 2
2 3
1 3
4 5
4 6
5 6
2 4
3 5
```

output

```
1 2
2 3
3 1
4 5
5 6
6 4
4 2
3 5
```

input

```
6 7
1 2
2 3
1 3
4 5
4 6
5 6
2 4
```

output

```
0
```