problem	papir	zmaj
source file	papir.pas papir.c papir.cpp	zmaj.pas zmaj.c zmaj.cpp
input data	stdin	stdin
output data	stdout	stdout
time limit (intel celeron 2.4 ghz)	1 sec	10 sec
memory limit (heap)	64 MB	64 MB
memory limit (stack)	16 MB	16 MB
points	**100**	**100**
	200	

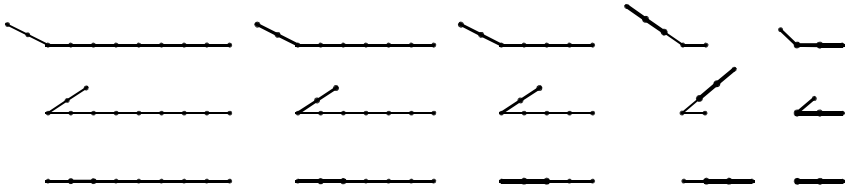
papir

Mickey has a long strip of paper and he wants to fold it several times in order to make it as short as possible.

Strip is of size **1xN cm** and it is **divided into N equal squares**. Each square is colored on both sides either in red or blue color, under the restriction that the two sides of the same square are colored in **different colors**.

Mickey is folding the strip in the following way: in every step he chooses some number K (positive integer **smaller** than the current length of the strip), and at the position of K cm from the left edge of the strip he lifts the left part of the strip and folds it firmly onto the right part of the strip. During the folding, two squares of the **same color must not** be pressed one onto the other.

Example of folding steps corresponding to the solution of the third example:



Write a program that will determine **some sequence of folding steps** so that final folded strip is **as short as possible**.

input data

First line of input contains an integer N, $2 \leq N \leq 10{,}000$, the length of the strip.

Second line contains a sequence of N characters '**c**' ('crveno' i.e. 'red') or '**p**' ('plavo' or 'blue'). The i^{th} character in the sequence denotes the color of the upper side of the i^{th} square on the strip.

output data

First line of output should contain the length of strip at the end of the folding.

Second line should contain the number M, total number of folding steps, and third line should contain M numbers so that i^{th} number denotes the number K at the i^{th} folding step.

Note: solution needs not to be unique.

examples

input	**input**	**input**
5 ccpcc	5 pcpcp	10 ccppccppcp
output	**output**	**output**
5 0	1 4 1 1 1 1	2 5 2 2 2 3 1

Programming language ZMAJ (somewhat similar to BASIC) has 26 variables denoted by lowercase letters of the English alphabet ('a'-'z'). Initial value of all the variables is 0, and during program execution they can store integer values between 0 and 9999 (inclusive). Result of any operation in the program is reduced modulo 10,000 before it is stored into a variable.

Each line of our program contains exactly one of the following commands:

BEGIN	appears only at the beginning of the program.
=	to the variable on the left side of the operator we assign the value of the expression on the right side. Expression consists of one or more addiction and subtraction operations on terms, where each term is either a constant or a variable possibly multiplied by a constant (i.e. a = 2b + 4 - c). Each binary operator will be surrounded by one space character on each side, and unary minus operator is not allowed. Constants are integers between 0 and 999 (inclusive).
REPEAT n	denotes the beginning of the block that is repeated *n* times, $1 \leq n \leq 100{,}000$.
STOP	denotes the end of the REPEAT block or end of program.
PRINT var	prints the current value of variable *var* in the following format 'var = value'.

Write a program that will generate output for the given ZMAJ program.

input data

Input will contain a **legal** program in the programming language ZMAJ, and it will be at most 50 lines long.

Each line of input contains one command and each command will have at most 100 characters.

Blocks of command inside BEGIN...STOP or REPEAT...STOP will be **indented exactly 3 spaces**.

output data

Output should contain all the outputs from PRINT commands during the program execution.

Note: input will be such that the PRINT command will execute at most 20 times.

examples

input

```
BEGIN
   n = 10
   k = 1
   REPEAT 3
      REPEAT 2
         n = n + k
         PRINT n
      STOP
      k = 3 - k
   STOP
STOP
```

output

```
n = 11
n = 12
n = 14
n = 16
n = 17
n = 18
```

input

```
BEGIN
   a = 1
   b = a
   PRINT a
   PRINT b
   REPEAT 10
      c = a + b
      a = b
      b = c
      PRINT c
   STOP
STOP
```

output

```
a = 1
b = 1
c = 2
c = 3
c = 5
c = 8
c = 13
c = 21
c = 34
c = 55
c = 89
c = 144
```

input

```
BEGIN
   x = 1
   REPEAT 4
      REPEAT 99999
         x = 2x
      STOP
      PRINT x
   STOP
STOP
```

output

```
x = 4688
x = 7344
x = 8672
x = 4336
```