The input data for each problem had to be read from the text file and the answer had to be written to the fixed file OUTPUT.TXT. The program had to ask the user for the name of the input file. All the programs were tested on IBM PC 486, 25 MHz computers.

A SQUARE AND A CIRCLE (Estonia)

30 points, run time limit – 10 sec.

The first two rows of the input file contain the co-ordinates of two opposite vertices of a square on the plane, the third row contains the co-ordinates of the centre and radius of a circle (real numbers).

Find the common area of two figures. The result must be written in the first line of the text file with two digits after a decimal point.

Example

```
INPUT                   OUTPUT
0 5.0                    46.64
10.00 -5
3.0 1 4.000
```

The input data for each problem had to be read from the text file and the answer had to be written to the fixed file OUTPUT.TXT. The program had to ask the user for the name of the input file. All the programs were tested on IBM PC 486, 25 MHz computers.

A NICE SEQUENCE (Latvia)

40 points, run time limit – 90 sec.

There is a sequence of N whole positive numbers a_1, a_2, . . . , a_N.

It is said that the sequence b_1, b_2, ..., b_L is a sub-sequence of the given sequence, if

1) for each i ($1 \le i \le L$) there exists such j ($1 \le j \le N$) that $b_i= a_j$;
2) for each i_1 and i_2 such that $i_1 < i_2$, $b_{i1} = a_{j1}$, $b_{i2} = a_{j2}$, the relation $j_1< j_2$ is valid.

It is said that the sequence b_1, b_2, ..., b_L is non-increasing, if for each i ($1 \le i < L$) $b_i \ge b_{i+1}$.
It is said that the sequence b_1, b_2, ..., b_L is non-decreasing, if for each i ($1 \le i < L$) $b_i \le b_{i+1}$.

The given sequence is called NICE, if we can choose K ($K > 0$) sub-sequences of this sequence such that:

1) each sub-sequence has at least 2 members,
2) each sub-sequence is either non-increasing or non-decreasing,
3) each member of the given sequence belongs directly to one sub-sequence.

Find out the least possible K for the given sequence.

Input Data

The input data are written to the text file and contain the value of N ($1 \le N \le 25$) in the first line and the values of sequence members in other N lines (one number per line, for each j ($1 \le a_j \le 100$)).

Output

The output data must be written to the text file and must contain:

1) value of K in the first line,
2) one example of dividing the given sequences to K sub-sequences (i. e. there must be N lines, each containing a_j and *index* ($1 \le$ *index* $\le K$) of sub-sequence to which a_j belongs).

If the given sequence is not NICE, the output file must contain only 0 in the first line.

Example 1

```
INPUT        OUTPUT        COMMENT

6            2             Sub-sequences are
12           12  1         12, 33, 33
33           33  1         and
97           97  2         97, 18, 15
18           18  2
15           15  2
33           33  1
```

Example 2

```
INPUT        OUTPUT
1            0
88
```

Example 3

```
INPUT         OUTPUT
4             1
77            77  1
22            22  1
22            22  1
11            11  1
```

The input data for each problem had to be read from the text file and the answer had to be written to the fixed file OUTPUT.TXT. The program had to ask the user for the name of the input file. All the programs were tested on IBM PC 486, 25 MHz computers.

LOGICAL EXPRESSIONS (Lithuania)
30 points, run time limit – 20 sec.

There are five logical expressions called *R1*, *R2*, *R3*, *R4*, *R5*. Only the logical variables *a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*, the logical operations

# (not)

& (and)

+ (or)

and round brackets can be used in the expressions. Priorities for the operations are as written above. A different order of executing the operations may be defined with round brackets.

The operations are defined by such a table

x	*y*	*#x*	*x&y*	*x+y*
False	*False*	*True*	*False*	*False*
False	*True*	*True*	*False*	*True*
True	*False*	*False*	*False*	*True*
True	*True*	*False*	*True*	*True*

It has been assumed that there are some expressions which may be expressed by the means of a single logical operation with one or two arguments:

$R_i = R_j$ (R_i is equivalent to R_j, i.e. R_i gives the same values as R_j for every set of variable values);

$R_i = \# R_j$ (R_i is equivalent to the negation of R_j);

$R_i = R_j \& R_k$ (conjunction);

$R_i = R_j + R_k$ (disjunction); where ($i \neq k$, $i \neq j$).

A basic set of expressions is a set of such expressions by which any other expression not contained in this set may be represented (expressed), as defined above.

Write a program to find:

a) any least basic set of expressions;

b) any representation (expression) of each expression (which does not belong to the basic set of expressions). The representation should be obtained using any single logical operation with one or two arguments, i.e. expressions from the basic set.

Input

The input data are written in the text file (one expression in one line) ***R1*** is in the first line, etc. There are no spaces in the expressions. The input data are assumed to be correct. The length of each line is less than 256.

Output

The output data should be written to the text file. The file should contain 5 lines, one line for each expression. Write the names of the basic expressions to the first lines of the output file. The rest of the lines should contain representations (expressions) of all other expressions (i.e., expressions not belonging to the basic set of expressions) in the form of equalities (equivalencies), as shown in the example below.

Examples

```
INPUT               OUTPUT
a&(b&b)             R1
c+c                 R2
a&b+c               R4
#a                  R3=R1+R2
a&b                 R5=R1
```