The input data for each problem should to be read from the text file and the answer should to be written to the fixed file OUTPUT.TXT. The program has to ask the user for the name of the input file. All the programs will be tested on IBM PC 486 DX2, 65 MHz computers.

A CURRENCY TABLE (Lithuania)

20 points, run time limit – 10 sec.

A table of currency ratios can be presented in the following way:

1		2	3	...	*n*
1	1.00	0.50	2.50	...	0.10
2	2.00	1.00	5.00	...	0.20
3	0.40	0.20	1.00	...	0.04
	...	...	...	1.00	...
n	10.00	5.00	25.00	...	1.00

The presented numbers 1, 2, 3, ... , *n* denote currency codes.

The ratio *m* written in the row *r* and the column *c* of the table shows that one unit of the currency *r* is worth *m* units of the currency *c*. For example, the number 0.50 in the table above means that one unit of currency 1 costs half of the unit of currency 2, while the number 2.00 shows a reverse ratio, i.e. that one unit of currency 2 is equal to two units of currency 1.

There is redundancy of information in the table. For example, it is possible to develop the whole table having only one row or column.

Write a program to complete the partly filled in currency ratio table. Assume that the input data are correct and not contradictory.

Input Data

The input file contains a partly filled in currency ratio table. The integer number *n* ($n \le 20$) which denotes the size of the table is written in the first line, followed by the table containing *n* rows. Each row contains *n* real numbers. Zeros are written instead of the missing numbers.

Examples

The file CURRENCY1.DAT contains the table

```
3
1.00   0.50   0.00
0.00   0.00   8.00
0.00   0.00   0.00
```

and CURRENCY2.DAT contains the table

```
3
1.00   0.00   4.00
0.00   1.00   0.00
0.00   0.00   0.00
```

Output Data

Complete the currency table as much as possible and write to the OUTPUT.TXT file. Each row of the table must be written on one line of the file. Write the values with two digits after a decimal point (as it is shown in the examples). Mark zeros for the elements of the table that can not be restored.

The results for sample files:

```
1.00   0.50   4.00
2.00   1.00   8.00
0.25   0.12   1.00

1.00   0.00   4.00
0.00   1.00   0.00
0.25   0.00   1.00
```

The input data for each problem should to be read from the text file and the answer should to be written to the fixed file OUTPUT.TXT. The program has to ask the user for the name of the input file. All the programs will be tested on IBM PC 486 DX2, 65 MHz computers.

AN EXPRESSION (Estonia)
30 points, run time limit – 10 sec.

An arithmetic expression is on the first line of the input file. It may contain the characters

`0..9, a..h, +, -`

where *a*, *b*, *c*, ..., *h* are variables.

Some values of the variables occurring in the expression are given on the remaining lines in the form

`<variable>=<integer>`

If the file contains enough information, then find the value of the expression, otherwise, write in the output file: `UNDEFINED`.

The length of the expression does not exceed 40 characters. The program will be tested with examples where the values of sub-expressions are not too large for a long integer type.

Examples

The files EXPR1.DAT, EXPR2.DAT and EXPR3.DAT contain the lines

```
235ahah+c-12bb    abc          23b
b=1               a=1          b=2
c=-7              b=2
a=10
h=2
```

The answers: `93981, UNDEFINED, 46.`

The input data for each problem should to be read from the text file and the answer should to be written to the fixed file OUTPUT.TXT. The program has to ask the user for the name of the input file. All the programs will be tested on IBM PC 486 DX2, 65 MHz computers.

DOUBLE PRIME NUMBERS (Latvia)

50 points, run time limit – 2 min.

$\{a_i\}$ is the sequence of prime numbers:

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, ...

The elements of the sequence $\{b_i\}$ are built from the elements of the sequence $\{a_i\}$ by *gluing together* elements in pairs:

23, 57, 1113, 1719, 2329, 3137, ...

$\{c_i\}$ is a sub-sequence of $\{b_i\}$, which contains only prime numbers:

$c_1 = 23$; $c_2 = 3137$; ...

Write a program that finds for the given j ($1 \le j \le 150$) the value of the member c_j.

Input Data

The input file contains only one integer j .

Output

The output file OUTPUT.TXT must contain only the value c_j.

Examples

(files PRIM1.DAT, PRIM2.DAT, PRIM3.DAT)

INPUT	OUTPUT
2	3137
1	23
5	167173