A. Vasya and Digital Root

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya has recently found out what a digital root of a number is and he decided to share his knowledge with you.

Let's assume that $S(n)$ is the sum of digits of number n, for example, $S(4098) = 4 + 0 + 9 + 8 = 21$. Then the digital root of number n equals to:

1. $dr(n) = S(n)$, if $S(n) < 10$;
2. $dr(n) = dr(S(n))$, if $S(n) \geq 10$.

For example, $dr(4098) = dr(21) = 3$.

Vasya is afraid of large numbers, so the numbers he works with are at most 10^{1000}. For all such numbers, he has proved that $dr(n) = S(S(S(S(n))))$ $(n \leq 10^{1000})$.

Now Vasya wants to quickly find numbers with the given digital root. The problem is, he hasn't learned how to do that and he asked you to help him. You task is, given numbers k and d, find the number consisting of exactly k digits (the leading zeroes are not allowed), with digital root equal to d, or else state that such number does not exist.

Input

The first line contains two integers k and d $(1 \leq k \leq 1000;\ 0 \leq d \leq 9)$.

Output

In a single line print either any number that meets the requirements (without the leading zeroes) or "`No solution`" (without the quotes), if the corresponding number does not exist.

The chosen number must consist of exactly k digits. We assume that number 0 doesn't contain any leading zeroes.

Examples

input

```
4 4
```

output

```
5881
```

input

```
5 1
```

output

```
36172
```

input

```
1 0
```

output

```
0
```

Note

For the first test sample $dr(5881) = dr(22) = 4$.

For the second test sample $dr(36172) = dr(19) = dr(10) = 1$.

B. Vasya and Public Transport

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya often uses public transport. The transport in the city is of two types: trolleys and buses. The city has n buses and m trolleys, the buses are numbered by integers from 1 to n, the trolleys are numbered by integers from 1 to m.

Public transport is not free. There are 4 types of tickets:

1. A ticket for one ride on some bus or trolley. It costs c_1 burles;
2. A ticket for an unlimited number of rides on some bus or on some trolley. It costs c_2 burles;
3. A ticket for an unlimited number of rides on all buses or all trolleys. It costs c_3 burles;
4. A ticket for an unlimited number of rides on all buses and trolleys. It costs c_4 burles.

Vasya knows for sure the number of rides he is going to make and the transport he is going to use. He asked you for help to find the minimum sum of burles he will have to spend on the tickets.

Input

The first line contains four integers c_1, c_2, c_3, c_4 $(1 \le c_1, c_2, c_3, c_4 \le 1000)$ — the costs of the tickets.

The second line contains two integers n and m $(1 \le n, m \le 1000)$ — the number of buses and trolleys Vasya is going to use.

The third line contains n integers a_i $(0 \le a_i \le 1000)$ — the number of times Vasya is going to use the bus number i.

The fourth line contains m integers b_i $(0 \le b_i \le 1000)$ — the number of times Vasya is going to use the trolley number i.

Output

Print a single number — the minimum sum of burles Vasya will have to spend on the tickets.

Examples

input

```
1 3 7 19
2 3
2 5
4 4 4
```

output

```
12
```

input

```
4 3 2 1
1 3
798
1 2 3
```

output

```
1
```

input

```
100 100 8 100
3 5
7 94 12
100 1 47 0 42
```

output

```
16
```

Note

In the first sample the profitable strategy is to buy two tickets of the first type (for the first bus), one ticket of the second type (for the second bus) and one ticket of the third type (for all trolleys). It totals to $(2 \cdot 1) + 3 + 7 = 12$ burles.

In the second sample the profitable strategy is to buy one ticket of the fourth type.

In the third sample the profitable strategy is to buy two tickets of the third type: for all buses and for all trolleys.

C. Vasya and Robot

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya has n items lying in a line. The items are consecutively numbered by numbers from 1 to n in such a way that the leftmost item has number 1, the rightmost item has number n. Each item has a weight, the i-th item weights w_i kilograms.

Vasya needs to collect all these items, however he won't do it by himself. He uses his brand new robot. The robot has two different arms — the left one and the right one. The robot can consecutively perform the following actions:

1. Take the leftmost item with the left hand and spend $w_i \cdot l$ energy units (w_i is a weight of the leftmost item, l is some parameter). If the previous action was the same (left-hand), then the robot spends extra Q_l energy units;
2. Take the rightmost item with the right hand and spend $w_j \cdot r$ energy units (w_j is a weight of the rightmost item, r is some parameter). If the previous action was the same (right-hand), then the robot spends extra Q_r energy units;

Naturally, Vasya wants to program the robot in a way that the robot spends as little energy as possible. He asked you to solve this problem. Your task is to find the minimum number of energy units robot spends to collect all items.

Input

The first line contains five integers n, l, r, Q_l, Q_r ($1 \le n \le 10^5$; $1 \le l, r \le 100$; $1 \le Q_l, Q_r \le 10^4$).

The second line contains n integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 100$).

Output

In the single line print a single number — the answer to the problem.

Examples

input

```
3 4 4 19 1
42 3 99
```

output

```
576
```

input

```
4 7 2 3 9
1 2 3 4
```

output

```
34
```

Note

Consider the first sample. As $l = r$, we can take an item in turns: first from the left side, then from the right one and last item from the left. In total the robot spends $4 \cdot 42 + 4 \cdot 99 + 4 \cdot 3 = 576$ energy units.

The second sample. The optimal solution is to take one item from the right, then one item from the left and two items from the right. In total the robot spends $(2 \cdot 4) + (7 \cdot 1) + (2 \cdot 3) + (2 \cdot 2 + 9) = 34$ energy units.

D. Game with Strings

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Given an $n \times n$ table T consisting of lowercase English letters. We'll consider some string s *good* if the table contains a correct path corresponding to the given string. In other words, good strings are all strings we can obtain by moving from the left upper cell of the table only to the right and down. Here's the formal definition of correct paths:

Consider rows of the table are numbered from 1 to n from top to bottom, and columns of the table are numbered from 1 to n from left to the right. Cell (r, c) is a cell of table T on the r-th row and in the c-th column. This cell corresponds to letter $T_{r, c}$.

A path of length k is a sequence of table cells $[(r_1, c_1), (r_2, c_2), \dots, (r_k, c_k)]$. The following paths are correct:

1. There is only one correct path of length 1, that is, consisting of a single cell: $[(1, 1)]$;
2. Let's assume that $[(r_1, c_1), \dots, (r_m, c_m)]$ is a correct path of length m, then paths $[(r_1, c_1), \dots, (r_m, c_m), (r_m + 1, c_m)]$ and $[(r_1, c_1), \dots, (r_m, c_m), (r_m, c_m + 1)]$ are correct paths of length $m + 1$.

We should assume that a path $[(r_1, c_1), (r_2, c_2), \dots, (r_k, c_k)]$ corresponds to a string of length k: $T_{r_1, c_1} + T_{r_2, c_2} + \dots + T_{r_k, c_k}$.

Two players play the following game: initially they have an empty string. Then the players take turns to add a letter to the end of the string. After each move (adding a new letter) the resulting string must be good. The game ends after $2n - 1$ turns. A player wins by the following scenario:

1. If the resulting string has strictly more letters "a" than letters "b", then the first player wins;
2. If the resulting string has strictly more letters "b" than letters "a", then the second player wins;
3. If the resulting string has the same number of letters "a" and "b", then the players end the game with a draw.

Your task is to determine the result of the game provided that both players played optimally well.

Input

The first line contains a single number n $(1 \le n \le 20)$.

Next n lines contain n lowercase English letters each — table T.

Output

In a single line print string "`FIRST`", if the first player wins, "`SECOND`", if the second player wins and "`DRAW`", if the game ends with a draw.

Examples

input

```
2
ab
cd
```

output

```
DRAW
```

input

```
2
xa
ay
```

output

```
FIRST
```

input

```
3
aab
bcb
bac
```

output

```
DRAW
```

Note

Consider the first sample:

Good strings are strings: a, ab, ac, abd, acd.

The first player moves first and adds letter a to the string, as there is only one good string of length 1. Then the second player can add b or c and the game will end with strings abd or acd, correspondingly. In the first case it will be a draw (the string has one a and one b), in the second case the first player wins. Naturally, in this case the second player prefers to choose letter b and end the game with a draw.

Consider the second sample:

Good strings are: x, xa, xay.

We can see that the game will end with string xay and the first player wins.

E. Vasya and Beautiful Arrays

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya's got a birthday coming up and his mom decided to give him an array of positive integers a of length n.

Vasya thinks that an array's beauty is the greatest common divisor of all its elements. His mom, of course, wants to give him as beautiful an array as possible (with largest possible beauty). Unfortunately, the shop has only one array a left. On the plus side, the seller said that he could decrease some numbers in the array (no more than by k for each number).

The seller can obtain array b from array a if the following conditions hold: $b_i > 0;\ 0 \le a_i - b_i \le k$ for all $1 \le i \le n$.

Help mom find the maximum possible beauty of the array she will give to Vasya (that seller can obtain).

Input

The first line contains two integers n and k $(1 \le n \le 3\cdot10^5; 1 \le k \le 10^6)$. The second line contains n integers a_i $(1 \le a_i \le 10^6)$ — array a.

Output

In the single line print a single number — the maximum possible beauty of the resulting array.

Examples

input

```
6 1
3 6 10 12 13 16
```

output

```
3
```

input

```
5 3
8 21 52 15 77
```

output

```
7
```

Note

In the first sample we can obtain the array:

$3\ \ 6\ \ 9\ \ 12\ \ 12\ \ 15$

In the second sample we can obtain the next array:

$7\ \ 21\ \ 49\ \ 14\ \ 77$