A. Almost Prime

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

A number is called almost prime if it has exactly two distinct prime divisors. For example, numbers 6, 18, 24 are almost prime, while 4, 8, 9, 42 are not. Find the amount of almost prime numbers which are between 1 and n, inclusive.

Input

Input contains one integer number n ($1 \le n \le 3000$).

Output

Output the amount of almost prime numbers between 1 and n, inclusive.

Examples

input

```
10
```

output

```
2
```

input

```
21
```

output

```
8
```

B. Regular Bracket Sequence

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

A bracket sequence is called regular if it is possible to obtain correct arithmetic expression by inserting characters «+» and «1» into this sequence. For example, sequences «(())()», «()» and «(()(()))» are regular, while «)(», «(()» and «(()))(» are not.

One day Johnny got bracket sequence. He decided to remove some of the brackets from it in order to obtain a regular bracket sequence. What is the maximum length of a regular bracket sequence which can be obtained?

Input

Input consists of a single line with non-empty string of «(» and «)» characters. Its length does not exceed 10^6.

Output

Output the maximum possible length of a regular bracket sequence.

Examples

input

```
(()))(
```

output

```
4
```

input

```
((()())
```

output

```
6
```

C. Parquet

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

Once Bob decided to lay a parquet floor in his living room. The living room is of size $n \times m$ metres. Bob had planks of three types: a planks 1×2 meters, b planks 2×1 meters, and c planks 2×2 meters. Help Bob find out, if it is possible to parquet the living room with such a set of planks, and if it is possible, find one of the possible ways to do so. Bob doesn't have to use all the planks.

Input

The first input line contains 5 space-separated integer numbers n, m, a, b, c ($1 \le n, m \le 100, 0 \le a, b, c \le 10^4$), n and m — the living room dimensions, a, b and c — amount of planks 1×2, 2×1 и 2×2 respectively. It's not allowed to turn the planks.

Output

If it is not possible to parquet the room with such a set of planks, output `IMPOSSIBLE`. Otherwise output one of the possible ways to parquet the room — output n lines with m lower-case Latin letters each. Two squares with common sides should contain the same letters, if they belong to one and the same plank, and different letters otherwise. Different planks can be marked with one and the same letter (see examples). If the answer is not unique, output any.

Examples

input
```
2 6 2 2 1
```
output
```
aabcca
aabdda
```

input
```
1 1 100 100 100
```
output
```
IMPOSSIBLE
```

input
```
4 4 10 10 10
```
output
```
aabb
aabb
bbaa
bbaa
```

D. Tickets

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

As a big fan of Formula One, Charlie is really happy with the fact that he has to organize ticket sells for the next Grand Prix race in his own city. Unfortunately, the finacial crisis is striking everywhere and all the banknotes left in his country are valued either 10 euros or 20 euros. The price of all tickets for the race is 10 euros, so whenever someone comes to the ticket store only with 20 euro banknote Charlie must have a 10 euro banknote to give them change. Charlie realize that with the huge deficit of banknotes this could be a problem. Charlie has some priceless information but couldn't make use of it, so he needs your help. Exactly $n + m$ people will come to buy a ticket. n of them will have only a single 10 euro banknote, and m of them will have only a single 20 euro banknote. Currently Charlie has k 10 euro banknotes, which he can use for change if needed. All $n + m$ people will come to the ticket store in random order, all orders are equiprobable. Return the probability that the ticket selling process will run smoothly, i.e. Charlie will have change for every person with 20 euro banknote.

Input

The input consist of a single line with three space separated integers, n, m and k ($0 \le n, m \le 10^5$, $0 \le k \le 10$).

Output

Output on a single line the desired probability with at least 4 digits after the decimal point.

Examples

input

```
5 3 1
```

output

```
0.857143
```

input

```
0 5 5
```

output

```
1
```

input

```
0 1 0
```

output

```
0
```

E. Multithreading

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

You are given the following concurrent program. There are N processes and the i-th process has the following pseudocode:

repeat n_i times
  $y_i := y$
  $y := y_i + 1$
end repeat

Here y is a shared variable. Everything else is local for the process. All actions on a given row are atomic, i.e. when the process starts executing a row it is never interrupted. Beyond that all interleavings are possible, i.e. every process that has yet work to do can be granted the rights to execute its next row. In the beginning $y = 0$. You will be given an integer W and n_i, for $i = 1, \dots, N$. Determine if it is possible that after all processes terminate, $y = W$, and if it is possible output an arbitrary schedule that will produce this final value.

Input

In the first line of the input you will be given two space separated integers N ($1 \le N \le 100$) and W ($-10^9 \le W \le 10^9$). In the second line there are N space separated integers n_i ($1 \le n_i \le 1000$).

Output

On the first line of the output write Yes if it is possible that at the end $y = W$, or No otherwise. If the answer is No then there is no second line, but if the answer is Yes, then on the second line output a space separated list of integers representing some schedule that leads to the desired result. For more information see note.

Examples

input
```
1 10
11
```
output
```
No
```

input
```
2 3
4 4
```
output
```
Yes
1 1 2 1 2 2 2 2 2 1 2 1 1 1 1 2
```

input
```
3 6
1 2 3
```
output
```
Yes
1 1 2 2 2 2 3 3 3 3 3 3
```

Note

For simplicity, assume that there is no repeat statement in the code of the processes, but the code from the loop is written the correct amount of times. The processes are numbered starting from 1. The list of integers represent which process works on its next instruction at a given step. For example, consider the schedule 1 2 2 1 3. First process 1 executes its first instruction, then process 2 executes its first two instructions, after that process 1 executes its second instruction, and finally process 3 executes its first instruction. The list must consists of exactly $2 \cdot \Sigma_{i = 1 \dots N} n_i$ numbers.