# Codeforces Round #471 (Div. 2)

## A. Feed the cat

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

After waking up at $hh$:$mm$, Andrew realised that he had forgotten to feed his only cat for yet another time (guess why there's only one cat). The cat's current hunger level is $H$ points, moreover each minute without food increases his hunger by $D$ points.

At any time Andrew can visit the store where tasty buns are sold (you can assume that is doesn't take time to get to the store and back). One such bun costs $C$ roubles and decreases hunger by $N$ points. Since the demand for bakery drops heavily in the evening, there is a special $20\%$ discount for buns starting from $20{:}00$ (note that the cost might become rational). Of course, buns cannot be sold by parts.

Determine the minimum amount of money Andrew has to spend in order to feed his cat. The cat is considered fed if its hunger level is less than or equal to zero.

### Input

The first line contains two integers $hh$ and $mm$ ($00 \le hh \le 23$, $00 \le mm \le 59$) — the time of Andrew's awakening.

The second line contains four integers $H$, $D$, $C$ and $N$ ($1 \le H \le 10^5$, $1 \le D, C, N \le 10^2$).

### Output

Output the minimum amount of money to within three decimal digits. You answer is considered correct, if its absolute or relative error does not exceed $10^{-4}$.

Formally, let your answer be $a$, and the jury's answer be $b$. Your answer is considered correct if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-4}$.

**Examples**

| input |
| --- |
| 19 00<br>255 1 100 1 |
| **output** |
| 25200.0000 |

| input |
| --- |
| 17 41<br>1000 6 15 11 |
| **output** |
| 1365.0000 |

### Note

In the first sample Andrew can visit the store at exactly $20{:}00$. The cat's hunger will be equal to $315$, hence it will be necessary to purchase $315$ buns. The discount makes the final answer $25200$ roubles.

In the second sample it's optimal to visit the store right after he wakes up. Then he'll have to buy $91$ bins per $15$ roubles each and spend a total of $1365$ roubles.

## B. Not simply beatiful strings

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

Let's call a string *adorable* if its letters can be realigned in such a way that they form two consequent groups of equal symbols (note that different groups must contain different symbols). For example, *ababa* is *adorable* (you can transform it to *aaabb*, where the first three letters form a group of $a$-s and others — a group of $b$-s), but *cccc* is not since in each possible consequent partition letters in these two groups coincide.

You're given a string $s$. Check whether it can be split into two non-empty subsequences such that the strings formed by these subsequences are *adorable*. Here a subsequence is an arbitrary set of indexes of the string.

### Input

The only line contains $s$ ($1 \le |s| \le 10^5$) consisting of lowercase latin letters.

### Output
Print «Yes» if the string can be split according to the criteria above or «No» otherwise.

Each letter can be printed in arbitrary case.

**Examples**

| input |
| --- |
| ababa |
| **output** |
| Yes |

| input |
| --- |
| zzcxx |
| **output** |
| Yes |

| input |
| --- |
| yeee |
| **output** |
| No |

### Note
In sample case two *zzcxx* can be split into subsequences *zc* and *zxx* each of which is *adorable*.

There's no suitable partition in sample case three.

## C. Sad powers

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

You're given $Q$ queries of the form $(L, R)$.

For each query you have to find the number of such $x$ that $L \le x \le R$ and there exist integer numbers $a > 0, p > 1$ such that $x = a^p$.

### Input
The first line contains the number of queries $Q$ $(1 \le Q \le 10^5)$.

The next $Q$ lines contains two integers $L, R$ each $(1 \le L \le R \le 10^{18})$.

### Output
Output $Q$ lines — the answers to the queries.

**Example**

| input |
| --- |
| 6<br>1 4<br>9 9<br>5 7<br>12 29<br>137 591<br>1 1000000 |
| **output** |
| 2<br>1<br>0<br>3<br>17<br>1111 |

### Note
In query one the suitable numbers are $1$ and $4$.

## D. Scissors

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

output: standard output

Jenya has recently acquired quite a useful tool — $k$-scissors for cutting strings. They are generally used for cutting out two non-intersecting substrings of length $k$ from an arbitrary string $s$ (its length should be at least $2 \cdot k$ in order to perform this operation) and concatenating them afterwards (preserving the initial order). For example, with the help of $2$-scissors you can cut $ab$ and $de$ out of $abcde$ and concatenate them into $abde$, but not $ab$ and $bc$ since they're intersecting.

It's a nice idea to test this tool before using it in practice. After looking through the papers, Jenya came up with two strings $s$ and $t$. His question is whether it is possible to apply his scissors to string $s$ such that the resulting concatenation contains $t$ as a substring?

### Input

The first line contains three integers $n$, $m$, $k$ ($2 \leq m \leq 2 \cdot k \leq n \leq 5 \cdot 10^5$) — length of $s$, length of $t$ and the aforementioned scissors' parameter correspondingly.

The next two lines feature $s$ and $t$ consisting of lowercase latin letters.

### Output

If there is no answer, print «No».

Otherwise print «Yes» and two integers $L$ and $R$ denoting the indexes where cutted substrings start ($1$-indexed). If there are several possible answers, output any.

**Examples**

| input |
| --- |
| 7 4 3<br>baabaab<br>aaaa |
| **output** |
| Yes<br>1 5 |

| input |
| --- |
| 6 3 2<br>cbcbcb<br>bcc |
| **output** |
| Yes<br>2 5 |

| input |
| --- |
| 7 5 3<br>aabbaaa<br>aaaaa |
| **output** |
| No |

### Note

In the first sample case you can cut out two substrings starting at $1$ and $5$. The resulting string *baaaab* contains *aaaa* as a substring.

In the second sample case the resulting string is *bccb*.

## E. Icicles

time limit per test: 2.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrew's favourite Krakozyabra has recenly fled away and now he's eager to bring it back!

At the moment the refugee is inside an icy cave with $n$ icicles dangling from the ceiling located in integer coordinates numbered from $1$ to $n$. The distance between floor and the $i$-th icicle is equal to $a_i$.

Andrew is free to choose an arbitrary integer point $T$ in range from $1$ to $n$ inclusive and at time instant $0$ launch a sound wave spreading into both sides (left and right) at the speed of one point per second. Any icicle touched by the wave starts falling at the same speed (that means that in a second the distance from floor to icicle decreases by one but cannot become less that zero). While distance from icicle to floor is more than zero, it is considered passable; as soon as it becomes zero, the icicle blocks the path and prohibits passing.

Krakozyabra is initially (i.e. at time instant $0$) is located at point $\frac{1}{2}$ and starts running in the right direction at the speed of one point per second. You can assume that events in a single second happen in the following order: first Krakozyabra changes its position, and only then the sound spreads and icicles fall; in particular, that means that if Krakozyabra is currently at point $i - \frac{1}{2}$ and the falling (i.e. already touched by the sound wave) icicle

at point $i$ is $1$ point from the floor, then Krakozyabra will pass it and find itself at $i + \frac{1}{2}$ and only after that the icicle will finally fall and block the path.

Krakozyabra is considered entrapped if there are fallen (i.e. with $a_i = 0$) icicles both to the left and to the right of its current position. Help Andrew find the minimum possible time it takes to entrap Krakozyabra by choosing the optimal value of $T$ or report that this mission is impossible.

### Input

The first line contains the number of icicles $n$ ($2 \le n \le 10^5$).

The next line contains $n$ space-separated numbers $a_i$ ($1 \le a_i \le 10^5$) — the distances from floor to icicles.

### Output

Print an only integer — the minimum time it takes to entrap Krakozyabra between two fallen icicles. If it is impossible, print $-1$.

**Examples**

| input |
| --- |
| 5<br>1 4 3 5 1 |
| **output** |
| 3 |

| input |
| --- |
| 4<br>1 2 1 1 |
| **output** |
| 2 |

| input |
| --- |
| 2<br>2 1 |
| **output** |
| 3 |

| input |
| --- |
| 2<br>1 2 |
| **output** |
| -1 |

### Note

In sample case one it's optimal to launch the sound wave from point $3$. Then in two seconds icicles $1$ and $5$ will start falling, and in one more seconds they will block the paths. Krakozyabra will be located at $3\frac{1}{2}$ at that time. Note that icicle number $3$ will also be fallen, so there will actually be two icicles blocking the path to the left.

In sample case two it is optimal to launch the wave from point $2$ and entrap Krakozyabra in $2$ seconds.

In sample case four the answer is impossible.

## F. Heaps

time limit per test: 2.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You're given a tree with $n$ vertices rooted at $1$.

We say that there's a $k$-ary heap of depth $m$ located at $u$ if the following holds:

- For $m = 1$ $u$ itself is a $k$-ary heap of depth $1$.
- For $m > 1$ vertex $u$ is a $k$-ary heap of depth $m$ if **at least** $k$ of its children are $k$-ary heaps of depth **at least** $m - 1$.

Denote $dp_k(u)$ as maximum depth of $k$-ary heap in the subtree of $u$ (including $u$). Your goal is to compute $\sum_{k=1}^{n} \sum_{u=1}^{n} dp_k(u)$.

### Input

The first line contains an integer $n$ denoting the size of the tree ($2 \le n \le 3 \cdot 10^5$).

The next $n - 1$ lines contain two integers $u$, $v$ each, describing vertices connected by $i$-th edge.

It's guaranteed that the given configuration forms a tree.

## Output

Output the answer to the task.

### Examples

| input |
| --- |
| 4<br>1 3<br>2 3<br>4 3 |
| **output** |
| 21 |

| input |
| --- |
| 4<br>1 2<br>2 3<br>3 4 |
| **output** |
| 22 |

## Note

Consider sample case one.

For $k \geq 3$ all $dp_k$ will be equal to $1$.

For $k = 2$ $dp_k$ is $2$ if $u \in 1, 3$ and $1$ otherwise.

For $k = 1$ $dp_k$ values are $(3, 1, 2, 1)$ respectively.

To sum up, $4 \cdot 1 + 4 \cdot 1 + 2 \cdot 2 + 2 \cdot 1 + 3 + 1 + 2 + 1 = 21$.