# A. Tokitsukaze and Enhancement

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

Tokitsukaze is one of the characters in the game "Kantai Collection". In this game, every character has a common attribute — health points, shortened to HP.

In general, different values of HP are grouped into $4$ categories:

- Category $A$ if HP is in the form of $(4n+1)$, that is, when divided by $4$, the remainder is $1$;
- Category $B$ if HP is in the form of $(4n+3)$, that is, when divided by $4$, the remainder is $3$;
- Category $C$ if HP is in the form of $(4n+2)$, that is, when divided by $4$, the remainder is $2$;
- Category $D$ if HP is in the form of $4n$, that is, when divided by $4$, the remainder is $0$.

The above-mentioned $n$ can be any integer.

These $4$ categories ordered from highest to lowest as $A > B > C > D$, which means category $A$ is the highest and category $D$ is the lowest.

While playing the game, players can increase the HP of the character. Now, Tokitsukaze wants you to increase her HP by at most $2$ (that is, either by $0$, $1$ or $2$). How much should she increase her HP so that it has the highest possible category?

### Input

The only line contains a single integer $x$ ($30 \leq x \leq 100$) — the value Tokitsukaze's HP currently.

### Output

Print an integer $a$ ($0 \leq a \leq 2$) and an uppercase letter $b$ ($b \in \{A, B, C, D\}$), representing that the best way is to increase her HP by $a$, and then the category becomes $b$.

Note that the output characters are case-sensitive.

### Examples

**input**
```
33
```
**output**
```
0 A
```

**input**
```
98
```
**output**
```
1 B
```

### Note

For the first example, the category of Tokitsukaze's HP is already $A$, so you don't need to enhance her ability.

For the second example:

- If you don't increase her HP, its value is still $98$, which equals to $(4 \times 24 + 2)$, and its category is $C$.
- If you increase her HP by $1$, its value becomes $99$, which equals to $(4 \times 24 + 3)$, and its category becomes $B$.
- If you increase her HP by $2$, its value becomes $100$, which equals to $(4 \times 25)$, and its category becomes $D$.

Therefore, the best way is to increase her HP by $1$ so that the category of her HP becomes $B$.

# B. Tokitsukaze and Mahjong

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

Tokitsukaze is playing a game derivated from Japanese mahjong. In this game, she has three tiles in her hand. Each tile she owns is a suited tile, which means it has a suit (**manzu**, **pinzu** or **souzu**) and a number (a digit ranged from $1$ to $9$). In this problem, we use one digit and one lowercase letter, which is the first character of the suit, to represent a suited tile. All possible suited tiles are represented as `1m`, `2m`, ..., `9m`, `1p`, `2p`, ..., `9p`, `1s`, `2s`, ..., `9s`.

In order to win the game, she must have at least one **mentsu** (described below) in her hand, so sometimes she should draw extra suited tiles. After drawing a tile, the number of her tiles increases by one. She can draw any tiles she wants, including those already in her hand.

Do you know the minimum number of extra suited tiles she needs to draw so that she can win?

Here are some useful definitions in this game:

- A **mentsu**, also known as meld, is formed by a **koutsu** or a **shuntsu**;
- A **koutsu**, also known as triplet, is made of three identical tiles, such as `[1m, 1m, 1m]`, however, `[1m, 1p, 1s]` or `[1m, 4m, 7m]` is NOT a **koutsu**;
- A **shuntsu**, also known as sequence, is made of three sequential numbered tiles in the same suit, such as `[1m, 2m, 3m]` and `[5s, 7s, 6s]`, however, `[9m, 1m, 2m]` or `[1m, 2p, 3s]` is NOT a **shuntsu**.

Some examples:

- `[2m, 3p, 2s, 4m, 1s, 2s, 4s]` — it contains no **koutsu** or **shuntsu**, so it includes no **mentsu**;
- `[4s, 3m, 3p, 4s, 5p, 4s, 5p]` — it contains a **koutsu**, `[4s, 4s, 4s]`, but no **shuntsu**, so it includes a **mentsu**;
- `[5p, 5s, 9m, 4p, 1s, 7p, 7m, 6p]` — it contains no **koutsu** but a **shuntsu**, `[5p, 4p, 6p]` or `[5p, 7p, 6p]`, so it includes a **mentsu**.

Note that the order of tiles is unnecessary and you can assume the number of each type of suited tiles she can draw is infinite.

### Input

The only line contains **three** strings — the tiles in Tokitsukaze's hand. For each string, the first character is a digit ranged from $1$ to $9$ and the second character is `m`, `p` or `s`.

### Output

Print a single integer — the minimum number of extra suited tiles she needs to draw.

#### Examples

**input**
```
1s 2s 3s
```
**output**
```
0
```

**input**
```
9m 9m 9m
```
**output**
```
0
```

**input**
```
3p 9m 2p
```
**output**
```
1
```

### Note

In the first example, Tokitsukaze already has a **shuntsu**.

In the second example, Tokitsukaze already has a **koutsu**.

In the third example, Tokitsukaze can get a **shuntsu** by drawing one suited tile — `1p` or `4p`. The resulting tiles will be `[3p, 9m, 2p, 1p]` or `[3p, 9m, 2p, 4p]`.

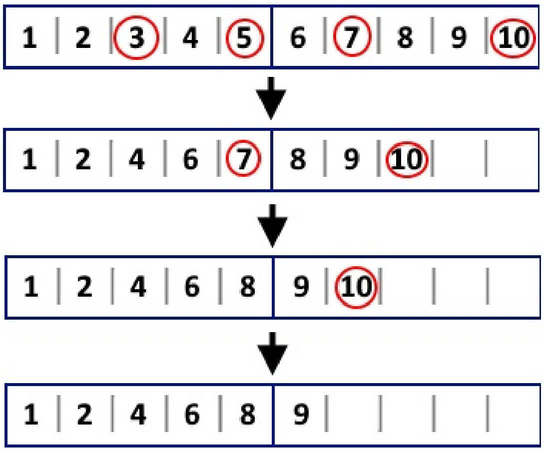
## C. Tokitsukaze and Discard Items

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

Recently, Tokitsukaze found an interesting game. Tokitsukaze had $n$ items at the beginning of this game. However, she thought there were too many items, so now she wants to discard $m$ ($1 \le m \le n$) special items of them.

These $n$ items are marked with indices from $1$ to $n$. In the beginning, the item with index $i$ is placed on the $i$-th position. Items are divided into several pages orderly, such that each page contains exactly $k$ positions and the last positions on the last page may be left empty.

Tokitsukaze would do the following operation: focus on the first special page that contains at least one special item, and at one time, Tokitsukaze would discard all special items on this page. After an item is discarded or moved, its old position would be empty, and then the item below it, if exists, would move up to this empty position. The movement may bring many items forward and even into previous pages, so Tokitsukaze would keep waiting until all the items stop moving, and then do the operation (i.e. check the special page and discard the special items) repeatedly until there is no item need to be discarded.




Consider the first example from the statement: $n=10$, $m=4$, $k=5$, $p=[3,5,7,10]$. The are two pages. Initially, the first page is *special* (since it is the first page containing a special item). So Tokitsukaze discards the special items with indices $3$ and $5$. After, the first page remains to be special. It contains $[1,2,4,6,7]$, Tokitsukaze discards the special item with index $7$. After, the second page is special (since it is the first page containing a special item). It contains $[9,10]$, Tokitsukaze discards the special item with index $10$.

Tokitsukaze wants to know the number of operations she would do in total.

### Input

The first line contains three integers $n$, $m$ and $k$ ($1 \le n \le 10^{18}$, $1 \le m \le 10^5$, $1 \le m, k \le n$) — the number of items, the number of special items to be discarded and the number of positions in each page.

The second line contains $m$ distinct integers $p_1, p_2, \ldots, p_m$ ($1 \le p_1 < p_2 < \ldots < p_m \le n$) — the indices of special items which should be discarded.

### Output

Print a single integer — the number of operations that Tokitsukaze would do in total.

#### Examples

**input**
```
10 4 5
3 5 7 10
```
**output**
```
3
```

**input**
```
13 4 5
7 8 9 10
```
**output**
```
1
```

### Note

For the first example:

- In the first operation, Tokitsukaze would focus on the first page $[1,2,3,4,5]$ and discard items with indices $3$ and $5$;
- In the second operation, Tokitsukaze would focus on the first page $[1,2,4,6,7]$ and discard item with index $7$;
- In the third operation, Tokitsukaze would focus on the second page $[9,10]$ and discard item with index $10$.

For the second example, Tokitsukaze would focus on the second page $[6,7,8,9,10]$ and discard all special items at once.

## D. Tokitsukaze, CSL and Stone Game

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

Tokitsukaze and CSL are playing a little game of stones.

In the beginning, there are $n$ piles of stones, the $i$-th pile of which has $a_i$ stones. The two players take turns making moves. Tokitsukaze moves first. On each turn the player chooses a nonempty pile and removes exactly one stone from the pile. A player loses if all of the piles are empty before his turn, or if after removing the stone, two piles (possibly empty) contain the same number of stones. Supposing that both players play optimally, who will win the game?

Consider an example: $n=3$ and sizes of piles are $a_1=2$, $a_2=3$, $a_3=0$. It is impossible to choose the empty pile, so Tokitsukaze has two choices: the first and the second piles. If she chooses the first pile then the state will be $[1,3,0]$ and it is a good move. But if she chooses the second pile then the state will be $[2, 2, 0]$ and she immediately loses. So the only good move for her is to choose the first pile.

Supposing that both players always take their best moves and never make mistakes, who will win the game?

Note that even if there are two piles with the same number of stones at the beginning, Tokitsukaze may still be able to make a valid first move. It is only necessary that there are no two piles with the same number of stones after she moves.

### Input

The first line contains a single integer $n$ ($1 \le n \le 10^5$) — the number of piles.

The second line contains $n$ integers $a_1, a_2, \ldots, a_n$ ($0 \le a_1, a_2, \ldots, a_n \le 10^9$), which mean the $i$-th pile has $a_i$ stones.

### Output

Print "`sjfnb`" (without quotes) if Tokitsukaze will win, or "`cslnb`" (without quotes) if CSL will win. Note the output characters are case-sensitive.

#### Examples

**input**
```
1
0
```
**output**
```
cslnb
```

**input**
```
2
1 0
```
**output**
```
cslnb
```

**input**
```
2
2 2
```
**output**
```
sjfnb
```

**input**
```
3
2 3 1
```
**output**
```
sjfnb
```

### Note

In the first example, Tokitsukaze cannot take any stone, so CSL will win.

In the second example, Tokitsukaze can only take a stone from the first pile, and then, even though they have no stone, these two piles will have the same number of stones, which implies CSL will win.

In the third example, Tokitsukaze will win. Here is one of the optimal ways:

- Firstly, Tokitsukaze can choose the first pile and take a stone from that pile.
- Then, CSL can only choose the first pile, because if he chooses the second pile, he will lose immediately.
- Finally, Tokitsukaze can choose the second pile, and then CSL will have no choice but to lose.

In the fourth example, they only have one good choice at any time, so Tokitsukaze can make the game lasting as long as possible and finally win.

## E. Tokitsukaze and Duel

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

"Duel!"

Betting on the lovely princess Claris, the duel between Tokitsukaze and Quailty has started.

There are $n$ cards in a row. Each card has two sides, one of which has color. At first, some of these cards are with color sides facing up and others are with color sides facing down. Then they take turns flipping cards, in which Tokitsukaze moves first. In each move, one should choose exactly $k$ consecutive cards and flip them to the same side, which means to make their color sides all face up or all face down. If all the color sides of these $n$ cards face the same direction after one's move, the one who takes this move will win.

Princess Claris wants to know who will win the game if Tokitsukaze and Quailty are so clever that they won't make mistakes.

### Input

The first line contains two integers $n$ and $k$ ($1 \le k \le n \le 10^5$).

The second line contains a single string of length $n$ that only consists of $0$ and $1$, representing the situation of these $n$ cards, where the color side of the $i$-th card faces up if the $i$-th character is $1$, or otherwise, it faces down and the $i$-th character is $0$.

### Output

Print "`once again`" (without quotes) if the total number of their moves can exceed $10^9$, which is considered a draw.

In other cases, print "`tokitsukaze`" (without quotes) if Tokitsukaze will win, or "`quailty`" (without quotes) if Quailty will win.

Note that the output characters are case-sensitive, and any wrong spelling would be rejected.

#### Examples

**input**

```
4 2
0101
```

**output**

```
quailty
```

**input**

```
6 1
010101
```

**output**

```
once again
```

**input**

```
6 5
010101
```

**output**

```
tokitsukaze
```

**input**

```
4 1
0011
```

**output**

```
once again
```

### Note

In the first example, no matter how Tokitsukaze moves, there would be three cards with color sides facing the same direction after her move, and Quailty can flip the last card to this direction and win.

In the second example, no matter how Tokitsukaze moves, Quailty can choose the same card and flip back to the initial situation, which can allow the game to end in a draw.

In the third example, Tokitsukaze can win by flipping the leftmost five cards up or flipping the rightmost five cards down.

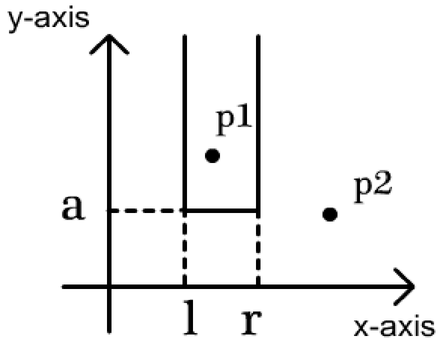
The fourth example can be explained in the same way as the second example does.

## F. Tokitsukaze and Strange Rectangle

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are $n$ points on the plane, the $i$-th of which is at $(x_i, y_i)$. Tokitsukaze wants to draw a strange rectangular area and pick all the points in the area.

The strange area is enclosed by three lines, $x = l$, $y = a$ and $x = r$, as its left side, its bottom side and its right side respectively, where $l$, $r$ and $a$ can be any real numbers satisfying that $l < r$. The upper side of the area is boundless, which you can regard as a line parallel to the $x$-axis at infinity. The following figure shows a strange rectangular area.




A point $(x_i, y_i)$ is in the strange rectangular area if and only if $l < x_i < r$ and $y_i > a$. For example, in the above figure, $p_1$ is in the area while $p_2$ is not.

Tokitsukaze wants to know how many different non-empty sets she can obtain by picking all the points in a strange rectangular area, where we think two sets are different if there exists at least one point in one set of them but not in the other.

### Input

The first line contains a single integer $n$ ($1 \leq n \leq 2 \times 10^5$) — the number of points on the plane.

The $i$-th of the next $n$ lines contains two integers $x_i$, $y_i$ ($1 \leq x_i, y_i \leq 10^9$) — the coordinates of the $i$-th point.

All points are distinct.

### Output

Print a single integer — the number of different non-empty sets of points she can obtain.

#### Examples

**input**

```
3
1 1
1 2
1 3
```

**output**

```
3
```

**input**

```
3
1 1
2 1
3 1
```

**output**

```
6
```

**input**

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

**output**

```
6
```

### Note

For the first example, there is exactly one set having $k$ points for $k = 1, 2, 3$, so the total number is $3$.

For the second example, the numbers of sets having $k$ points for $k = 1, 2, 3$ are $3$, $2$, $1$ respectively, and their sum is $6$.

For the third example, as the following figure shows, there are

- $2$ sets having one point;
- $3$ sets having two points;
- $1$ set having four points.

Therefore, the number of different non-empty sets in this example is $2 + 3 + 0 + 1 = 6$.