CODER'S CUP 2018 – FRESHMEN ROUND QUALIFIER A

QUESTION # 1.

Pashmak and Flowers

Pashmak decided to give Parmida a pair of flowers from the garden. There are *n* flowers in the garden and the *i*-th of them has a beauty number b_i. Parmida is a very strange girl so she doesn't want to have the two most beautiful flowers necessarily. She wants to have those pairs of flowers that their beauty difference is maximal possible!

Your task is to write a program which calculates two things:

1. The maximum beauty difference of flowers that Pashmak can give to Parmida.
2. The number of ways that Pashmak can pick the flowers. Two ways are considered different if and only if there is at least one flower that is chosen in the first way and not chosen in the second way.

Input Format

The first line of the input contains *n* ($2 \le n \le 2 \cdot 10^5$). In the next line there are *n* space-separated integers $b_1, b_2, ..., b_n$ ($1 \le b_i \le 10^9$).

Output Format

The only line of output should contain two integers. The maximum beauty difference and the number of ways this may happen, respectively.

Examples

Input

```
2
1 2
```

Output

```
1 1
```

Input

```
3
1 4 5
```

Output

```
4 1
```

Input

```
5
3 1 2 3 1
```

Output

```
2 4
```

QUESTION # 2.

Death Note

You received a notebook which is called *Death Note*. This notebook has infinite number of pages. A rule is written on the last page of this notebook. It says: "You have to write names in this notebook during n consecutive days. During the i-th day you have to write exactly ai names.". You got scared (of course you got scared, who wouldn't get scared if he just receives a notebook which is named *Death Note* with some strange rule written in it?).

Of course, you decided to follow this rule. When you calmed down, you came up with a strategy how you will write names in the notebook. You have calculated that each page of the notebook can contain exactly mm names. You will start writing names from the first page. You will write names on the current page as long as the limit on the number of names on this page is not exceeded. When the current page is over, you turn the page. Note that you *always* turn the page when it ends, it doesn't matter if it is the last day or not. If after someday the current page still can hold at least one name, during the next day you will continue writing the names from the current page.

Now you are interested in the following question: how many times will you turn the page during each day? You are interested in the number of pages you will turn each day from 1 to n.

Input

The first line of the input contains two integers n, m ($1 \le n \le 2 \cdot 10^5$, $1 \le m \le 10^9$) — the number of days you will write names in the notebook and the number of names which can be written on each page of the notebook.

The second line contains n integers a1,a2,…,an ($1 \le ai \le 10^9$), where ai means the number of names you will write in the notebook during the i-th day.

Output

Print exactly n integers t1,t2,…,tn, where ti is the number of times you will turn the page during the i-th day.

Examples

Input

```
3 5
3 7 9
```

Output

```
0 2 1
```

Input

```
4 20
10 9 19 2
```

Output

0 0 1 1

Input

1 100
99

Output

0

QUESTION # 3:

HAPPY LADYBUGS

Happy Ladybugs is a board game having the following properties:

- The board is represented by a string b, of length n. The ith character of the string,b[i] , denotes the ith cell of the board.
    - If b[i] is an underscore (i.e., _), it means the ith cell of the board is empty.
    - If b[i] is an uppercase English alphabetic letter (ascii[A-Z]), it means the ith cell contains a ladybug of color b[i].
    - String b will not contain any other characters.
- A ladybug is *happy* only when its left or right adjacent cell (i.e., b[i+1] and b[i-1]) is occupied by another ladybug having the same color.
- In a single move, you can move a ladybug from its current position to any empty cell.

Given the values of n and b for g games of Happy Ladybugs, determine if it's possible to make all the ladybugs happy. For each game, print YES on a new line if all the ladybugs can be made happy through some number of moves. Otherwise, print NO.

As an example, b = [YYR_B_BR]. You can move the rightmost B and R to make b = [YYRRBB__] and all the ladybugs are happy.

Input Format

The first line contains an integer g, the number of games.

The next g pairs of lines are in the following format:

- The first line contains an integer n, the number of cells on the board.
- The second line contains a string b describing the n cells of the board.

Constraints

I ≤ g,n ≤ 100

B[i] belongs to {_, ascii[A-Z]}

Output Format

For each game, print YES on the new line if it is possible to make all the ladybugs happy. Otherwise, print NO.

Examples

Input

```
4
7
RBY_YBR
6
X_Y__X
2
__
6
B_RRBR
```

Output

```
YES
NO
YES
YES
```

Input

```
5
5
AABBC
7
AABBC_C
1
_
10
DD__FQ_QQF
6
AABCBC
```

Output

```
NO
YES
YES
YES
NO
```

QUESTION # 4:

Maximal Intersection

You are given n segments on a number line; each endpoint of every segment has integer coordinates. Some segments can degenerate to points. Segments can intersect with each other, be nested in each other or even coincide.

The intersection of a sequence of segments is such a maximal set of points (not necessarily having integer coordinates) that each point lies within every segment from the sequence. If the resulting set isn't empty, then it always forms some continuous segment. The length of the intersection is the length of the resulting segment or 0 in case the intersection is an empty set.

For example, the intersection of segments [1;5] and [3;10] is [3;5] (length 2), the intersection of segments [1;5] and [5;7] is [5;5] (length 0) and the intersection of segments [1;5] and [6;6] is an empty set (length 0).

Your task is to remove exactly one segment from the given sequence in such a way that the intersection of the remaining (n−1) segments has the maximal possible length.

Input

The first line contains a single integer n ($2 \le n \le 3 \cdot 10^5$) — the number of segments in the sequence.

Each of the next n lines contains two integers li and ri ($0 \le li \le ri \le 10^9$) — the description of the i-th segment.

Output

Print a single integer — the maximal possible length of the intersection of (n−1) remaining segments after you remove exactly one segment from the sequence.

Examples

Input

```
4
1 3
2 6
0 4
3 3
```

Output

```
1
```

Input

```
5
2 6
1 3
```

0 4

1 20

0 4

Output

2

Input

3

4 5

1 2

9 20

Output

0

Note

In the first example you should remove the segment [3;3], the intersection will become [2;3] (length 1). Removing any other segment will result in the intersection [3;3] (length 0).
In the second example you should remove the segment [1;3] or segment [2;6], the intersection will become [2;4] (length 2) or [1;3] (length 2), respectively. Removing any other segment will result in the intersection [2;3] (length 1).
In the third example the intersection will become an empty set no matter the segment you remove.