DAY 2 - OVERVIEW

Name of the task:	**REPEATS**	**RECTANGLES**	**CAR PARK**
Input file name:	`repeats.in`	`rect.in`	`carpark.in`
Output file name:	`repeats.out`	`rect.out`	`carpark.out`
Time limit for one test case:	2 seconds	1 second	1 second
Source code file size limit:	1 MB	1 MB	1 MB
Memory limit:	64 MB	64 MB	64 MB
Maximum total points:	100	100	100

Comment which must be in source code first four lines if program is in **PASCAL** :	`{`
	`task: repeats`
	`lang: pascal`
	`}`
		`{`
		`task: rect`
		`lang: pascal`
		`}`
			`{`
			`task: carpark`
			`lang: pascal`
			`}`

Comment which must be in source code first four lines if program is in **C** :	`/*`
	`task: repeats`
	`lang: c`
	`*/`
		`/*`
		`task: rect`
		`lang: c`
		`*/`
			`/*`
			`task: carpark`
			`lang: c`
			`*/`

Comment which must be in source code first four lines if program is in **C++** :	`/*`
	`task: repeats`
	`lang: c++`
	`*/`
		`/*`
		`task: rect`
		`lang: c++`
		`*/`
			`/*`
			`task: carpark`
			`lang: c++`
			`*/`

Precondition for program to be tested:	Source code compiles without errors and passes example test case given in task description	Source code compiles without errors and passes example test case given in task description	Source code compiles without errors and passes example test case given in task description

A string *s* is called an *(k,l)-repeat* if s is obtained by concatenating $k \geq 1$ times some seed string *t* with length $l \geq 1$. For example, the string

$$s = \text{abaabaabaaba}$$

is a (4,3)-repeat with

$$t = \text{aba}$$

as its seed string. That is, the seed string *t* is 3 characters long, and the whole string *s* is obtained by repeating *t* 4 times.

Write a program for the following task: Your program is given a long string *u* consisting of characters 'a' and/or 'b' as input. Your program must find some *(k,l)-repeat* that occurs as substring within *u* with *k* as large as possible. For example, the input string

$$u = \text{babb\underline{abaabaabaabab}}$$

contains the underlined (4,3)-repeat s starting at position 5. Since *u* contains no other contiguous substring with more than 4 repeats, your program must output this underlined substring.

Input

In the first line of the input file `repeats.in` one integer - length of the input string *n* ($1 \leq n \leq 50000$) is given.

The next *n* file lines contain the input string, one character (either 'a' or 'b') per line, in order.

Output

The output file `repeats.out` must consist of three integers, each on its own line. They report the *(k, l)*-repeat your program found as follows:
1. The first line consists of the repeat count *k* that is maximized.
2. The second line consists of the length *l* of the seed string that is repeated *k* times.
3. The third and final line consists of the position p ($1 \leq p \leq n$) at which the *(k, l)*-repeat starts.

If for given test data there are different solutions with the same *k*, your program must report any one of them.

Example *(corresponds to string u given in task description)*

repeats.in	repeats.out
17 b a b b a b a a b a a b a a b a b	4 3 5 since a (4, 3)-repeat is found starting at the 5th character of the input string (which is line 6 of the input file).