1. Various Order of Operation Evaluation

A binary operation is defined below: where operand x ∈ {a, b, c}

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

For example, the results of ab, ba, and cc are b, c, and c respectively.
Different combinations of parentheses into such operations may lead it to different outcomes.

Given a string $x_1x_2\cdots x_n$ where $x_i \in \{a,b,c\}$, **write** an algorithm that counts the number of each result $\{a,b,c\}$ from all possible combinations of parentheses.

[Case 1]
String : *abaa*

The only combination that leads to a is $((ab)a)a$.
The combinations that lead to b are $((ab)(aa))$
$(a(ba))a$
$a((ba)a)$
$a(b(aa))$,
And there is no case of c.

Note : 1) You should make your algorithm as efficient as possible.
2) Total runtime of processing 10 cases with your algorithm **MUST NOT** exceed 1 second.
3) You are **NOT** allowed to use any optimization options at compilation,

[Input]
An input file “input.txt” will be given, which has 10 test cases.
Each case consists of two lines; first is length of string ($1 \le N \le 30$), second is string $x_1x_2\cdots x_n$ where $x_i \in \{a,b,c\}$.

[Output]
For each case, you should print the case number as #x where x is the index of the case. Then, print the number of all possible cases for a, b, c followed by space.
You must produce your results as “output.txt”

[Example]

Input (input.txt)

```
2                                                        ← First Case
ac
3                                                        ← Second Case
bbc
…
```

Ouput (output.txt)

```
#1 1 0 0
#2 1 0 1
…
```