Back From Summer '17 P4: Pair Programming

It's time for ICS (Information & Computer Science) at your school! After exploring the expensive new computer lab, your teacher tells you the task of the day is to learn problem-solving by pair programming. Thankfully, your assigned partner is your old friend Alex, and you know how much he likes money. Being a good entrepreneur but not such a good programmer, you pay Alex $100\,000$﷼ to write code while you go play Factorio.

After the bell rings, Alex hands the code to the teacher and sends you a copy over slack. Despite your mediocre programming abilities, you immediately realize that his code is actually garbage and much of it is wrong, straight up unreadable or maybe even both. Worried that your ICS mark might fall below 99%, you write your own program to check how many errors the code will have in the best and worst case scenarios.

The structure of the code is very simple. In each line, Alex can declare a new variable in one of two ways:

- `C s`: set the new variable to a constant value.
- `E v s` set the new variable to a value based on the variable declared at line `v`.

Once a variable is declared, the program automatically prints out its value for the teacher to check. You know that the structure of the code is correct, but the values or expressions in the code might be wrong. The `s` value in each line is a string consisting of either `AC`, `WA` or `?`.

- `AC` means that you are certain the expression is correct.
- `WA` means that you are certain the expression is wrong.
- `?` means that the code is so stupid that you cannot determine if it is correct.

An example of the code might be:

```
radius = 1                                //correct constant
pi = 3                                    //wrong constant
sum_of_all_naturals = -1 / 12             //stupid constant that may or may not be
right

tau = 2 * pi                              //correct expression (despite the
variable above being wrong)
diameter = 3 * radius                     //wrong expression
solution_to_P_vs_NP = 12345 * sum_of_naturals   //stupid expression that may or may not
be right

area = pi * radius * radius               //illegal expression; can only be based
on one other value
```

Alex worries that the code might all be bad, but because you have recently been blessed with good luck by Steven Halim, an error at the beginning of your code will not completely destroy the rest of your program. If a variable is set based on a wrong variable **and** a wrong expression, it will output a correct answer. However, a variable set based on a wrong variable and a correct expression, or vice versa, will still output an incorrect answer.

Input Specifications

The first line will contain an integer N, the number of lines in the code.

At line i $(1 \le i \le N)$, Alex will declare a variable with either `C` or `E`.

- If the line starts with `C`, then it will be followed by a single string `s`.
- If the line starts with `E`, then it will be followed by an integer `v` and string `s`. It is guaranteed that $v < i$.

See above for an explanation of the lines.

Output specifications

The output should contain `2` integers: the least and greatest number of errors possible respectively.

Subtasks and Score

Subtask	Constraint	Score
1	$N \le 20$	25
2	Number of `?` $\le 10, N \le 1\,000$	25
3	No additional constraints	50

For all testcases

$1 \le N \le 100\,000$

Sample Input 1

```
4
C AC
E 1 ?
E 2 WA
E 2 WA
```

Sample Output 1

```
1 2
```

Sample Explanation 1

If line `2` is correct, then both line `3` and line `4` will be wrong, giving 2 incorrect answers.

If line `2` is incorrect, then by Steven Halim's blessing, line `3` and line `4` will self-correct, making line `2` the only incorrect answer in the whole program.

Sample Input 2

```
4
C WA
E 1 WA
E 2 WA
E 3 WA
```

Sample Output 2

```
2 2
```

Sample Explanation 2

The first and third lines are wrong, while the second and fourth lines are correct.