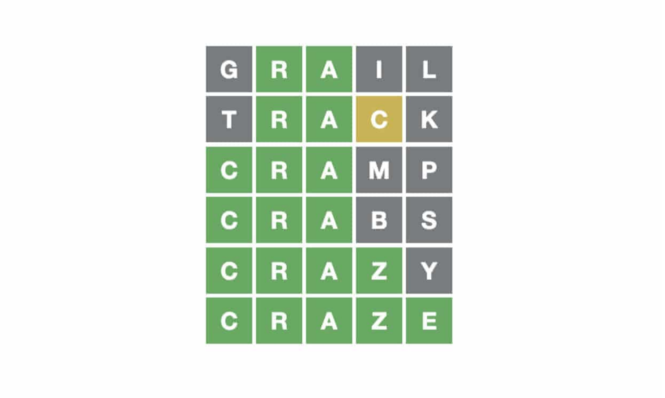
Solving Wordle (wordle)

The new, popular word game Wordle spreads so fast that its contagiousness has reached Luca. For those of you who have not yet heard about it, in Wordle you have to guess a 5-letter word in a limited number of attempts. After each attempt you get to know, for each position, whether your guess had the right letter in the right position, a letter present in the word but not in the right position, or a completely wrong letter.



G	R	A	I	L
T	R	A	C	K
C	R	A	M	P
C	R	A	B	S
C	R	A	Z	Y
C	R	A	Z	E

Figure 1: A screenshot of the game (by Josh Wardle).

Intrigued by the characteristics of the word game, Luca has decided to create his own more generic version. In this version, one needs to guess an N-letter word. *Every* possible sequence of letters of the English alphabet constitutes a potentially valid guess, but it is guaranteed that the word to guess does not contain the *same letter twice*.

You are in the middle of a game: you already guessed some letters but you still have to figure out some of them, which are indicated in the input with an underscore "_". You wonder: how many words could be a valid solution for the game, given the letters you know?

☞ Among the attachments of this task you may find a template file `wordle.*` with a sample incomplete implementation.

Input

The first line contains the only integer N. The second line contains N letters L_i: either an uppercase letter of the English alphabet or an underscore.

Output

You need to write a single line with an integer: the number of possible solutions.

Constraints

- $3 \leq N \leq 6$.
- The number of missing letters (i.e., underscores) ranges from 1 to $N - 1$.
- The input describes a valid game status, and thus does not contain duplicate letters.
- You only know some of the letters in their correct position: you can assume that Luca forgot which letters he previously entered wrongly (or in the wrong position) .

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points)	Examples.
– **Subtask 2** (10 points)	$N = 6$ and the number of missing letters is always 1.
– **Subtask 3** (20 points)	The number of missing letters is always 1.
– **Subtask 4** (30 points)	The number of missing letters is always 1 or 2.
– **Subtask 5** (40 points)	No additional limitations.

Examples

input	output

`5`
`H O U S _`
	`22`

`3`
`A _ _`
	`600`

Explanation

In the **first sample case** there are 22 valid possibilities. One of the obvious ones is an `E` for the fifth position. Note that `B` is a valid possibility too (even though it does not form a valid English word), but `S` is not, given that the letter is already present in the word in the fourth position.

In the **second sample case** there are 600 valid possibilities. Among them:

- `A C E`
- `A I R`
- `A N T`