Word by Word (`wordle2`)

> ✍ This is an interactive problem. Your program communicates with the evaluator: it should alternately write messages to the evaluator on the standard output and read the next input from the standard input.

Wordle is a web-based word game created and developed by Welsh software engineer Josh Wardle. Players have six attempts to guess a five-letter word, with feedback given for each guess in the form of coloured tiles indicating when letters match or occupy the correct position. — Wikipedia

An example of a Wordle game

In this problem, you will have to find a hidden string $s = \overline{s_1 s_2 s_3 s_4 s_5}$ consisting of 5 lowercase letters by using at most 9 guesses.

To make a guess, you need to pick a string g consisting of 5 lowercase letters and print a line with the following format:

- `"?  g"`

After that, you will receive a string r consisting of 5 characters, which can be 'G' (green), 'Y' (yellow) or 'W' (white). These colors largely have the following meanings:

- If r_i is 'G' (green), then the i-th letter from your guess is correct (that is, $g_i = s_i$).

- If r_i is 'Y' (yellow), then the i-th letter from your guess is incorrect, however it occurs somewhere else in the hidden string (that is, $g_i \neq s_i$ and there exists some $j \neq i$ for which $g_i = s_j$).

- If r_i is 'W' (white), then the i-th letter from your guess is incorrect.

However, if you repeat a letter in your guess more times than it appears in the hidden string, then the excess will be colored in white.

For example, if $s =$ `"jazzy"` and $g =$ `"zyzyz"`, then r will be equal to `"YYGWW"`:

- Since $g_3 = s_3$, $r_3 =$ 'G'.

- Since $g_1 \neq s_1$, but $g_1 = s_4$, $r_1 =$ 'Y'.

- Since $g_2 \neq s_2$, but $g_2 = s_5$, $r_2 =$ 'Y'.

- Since there are no more occurrences of 'y' and 'z' in s to match with g_4 and g_5, both r_4 and r_5 are equal to 'W'.

After you have found the hidden string, print a line with the following format:

- "! answer"

Please note that printing the answer **does not count** towards the total number of guesses.

> ☞ Among the attachments of this task you may find a template file `wordle2.*` with a sample incomplete implementation.

Input

Each test contains multiple test cases. The first line contains the number of test cases T.

In each test case, the jury will choose a hidden string consisting of 5 lowercase letters, which you can start guessing immediately.

After making a guess, the response r from the jury will appear in the input.

Output

To make a guess, you need to pick a string g consisting of 5 lowercase letters and print a line with the following format:

- "? g"

After that, you will receive a string r consisting of 5 characters, which can be 'G' (green), 'Y' (yellow) or 'W' (white).

After you have found the hidden string, print a line with the following format:

- "! answer"

After solving a test case, your program should move to the next one immediately. After solving all test cases, your program should terminate immediately.

If your solution uses more than 30 guesses for a particular test case, you will receive the `Wrong answer` verdict.

Please note that printing the answer **does not count** towards the total number of guesses.

After printing a guess or the answer for a test case, **do not forget to print the end of line character ('\n') and to flush the output**. Otherwise, you may receive either the `Time Limit Exceeded` or the `Memory Limit Exceeded` verdict. To flush the output, use:

- `fflush(stdout)` or `cout.flush()` in C++;

- `System.out.flush()` in Java;

- `flush(output)` in Pascal;

- `stdout.flush()` in Python;

- see the documentation for other languages.

Constraints

- $1 \le T \le 5000$.

- The interactor in this task is **not adaptive**. In other words, the hidden string is fixed in every test case and does not change during the interaction.

Scoring

If your program did not solve all test cases correctly for a given test, then you will receive no points for this test.

Otherwise, let Q be the maximum number of guesses your program needed to guess the hidden string across all test cases for a given test.

- If $Q > 30$, then you will receive no points for this test.

- If $26 \le Q \le 30$, then you will receive 10% of the total number of points for this test.

- If $12 \le Q < 26$, then you will receive 25% of the total number of points for this test.

- If $Q = 11$, then you will receive 40% of the total number of points for this test.

- If $Q = 10$, then you will receive 60% of the total number of points for this test.

- If $Q \le 9$, then you will receive the full amount of points for this test.

Your program will be tested against several tests grouped in subtasks. The score for each subtask will be equal to the smallest point percentage across all tests from that subtask multiplied by the number of points assigned to the subtask.

– **Subtask 1** (0 points)   Examples.

– **Subtask 2** (100 points)   No additional limitations.

Examples

input	output
3 GGGGG WWGYW WWGGY WGGGG GGGGG YYGWW YWGGG WGGGG	? aaaaa ! aaaaa ? green ? piece ? beech ? leech ! leech ? zyzyz ? azzzy ? yazzy ! jazzy

Explanation

The interaction starts by reading the number of test cases T. In this example, $T = 3$.

- In the first test case, the hidden string is `"aaaaa"`. After guessing `"aaaaa"`, the jury responds with `"GGGGG"`, meaning that every character from the guess is correct. As such, we can submit our answer with `"!  aaaaa"` and move on to the next test case.

- In the second test case, the hidden string is `"leech"`. The interaction for this test case is depicted in the image from the statement.

- In the third test case, the hidden string is `"jazzy"`.

  - After guessing `"zyzyz"`, the jury responds with `"YYGWW"`.
  - After guessing `"azzzy"`, the jury responds with `"YWGGG"`.
  - After guessing `"yazzy"`, the jury responds with `"WGGGG"`.
  - Then, the answer is submitted with `"!jazzy"`. Note that there are other strings that are also consistent with the responses from the jury, such as: `"bazzy"`, `"cazzy"`, `"xazzy"`, . . .