# Problem P: Partitioning a square

In this problem you are given a number $n$ and your task is to produce a square of letters.
The square must have the following properties:

- Each letter is one of the first $n$ lowercase English letters (`a-z`).
- Each of those $n$ letters occurs the same number of times.
- All occurrences of each letter form one 4-connected region (explained below).
- The square is **as small as possible**.

We say that the occurrences of some letter $X$ form a 4-connected region if it is possible to travel from any $X$ to any other $X$ by only moving one step up/down/left/right at a time without stepping onto a letter other than $X$.

Below is an example of a $6 \times 6$ square divided into $n = 4$ equally large 4-connected regions. (Note that this is not the *smallest possible* square that can be divided into 4 regions.)

```
aaaaab
aaaabb
bbbbbd
bddddd
cdddcc
cccccc
```

## Input specification

The first line of the input file contains an integer $t$ specifying the number of test cases. Each test case is preceded by a blank line.

Each test case consists of a single line with a single positive integer $n$.

In the **easy subproblem P1** we have $t = 1$ and the only test case has $n = 4$.

In the **hard subproblem P2** we have $t \leq 50$ and each $n$ is between 1 and 26, inclusive.

## Output specification

For each test case, output several lines. The first of those lines should contain an integer $s$: the side length of your square. The rest of the output for that test case should consist of $s$ lines, each containing $s$ lowercase letters.

## What to submit

**Do not submit any programs.** Your task is to produce and submit the correct **output files** `p1.out` and `p2.out` for the provided input files `p1.in` and `p2.in`.

## Example

| input | output |
|---|---|
| 1<br><br>2 | 2<br>ab<br>ab |

 

# Problem Q: Qizz Quzz

After a successful presentation of your coding skills on the Fizz Buzz problem you have advanced to the second round of interviews for your dream job of code ninja at FizzBuzz Corp. In this interview you will tackle the Generalized Fizz Buzz problem.

Generalized Fizz Buzz is similar to the original Fizz Buzz. The program must print positive integers starting from 1, but some special numbers (and their multiples) should be replaced by strings. If something is divisible by multiple special numbers, it must be replaced by a concatenation of all their strings.

In this problem, your task **isn't** to implement Generalized Fizz Buzz. Instead, you will be given a sequence and you need to decide whether it can be the output of a Generalized Fizz Buzz program.

## Generalized Fizz Buzz

A particular instance of Generalized Fizz Buzz is defined by the following parameters:

- A positive integer $n$.
- A nonnegative integer $k$.
- Distinct positive integers $d_1, \ldots, d_k$ such that $d_1 < d_2 < \cdots < d_k$.
- Strings $s_1, \ldots, s_k$ of lowercase English letters.

Their meaning is as follows:

- The number $n$ is the length of the sequence the program should output.
- The number $k$ is the number of replacement rules.
- Each of the replacement rules has the form "Instead of multiples of $d_i$ print the string $s_i$."
- If multiple replacement rules apply, the label that should be printed is obtained by concatenating all the corresponding $s_i$, ordered **from the smallest to the largest divisor**.

For example, the original Fizz Buzz corresponds to $k = 2$, $d = (3, 5)$, and $s = (\texttt{fizz}, \texttt{buzz})$. The correct output of that program for $n = 17$ would be:

```
1 2 fizz 4 buzz fizz 7 8 fizz buzz 11 fizz 13 14 fizzbuzz 16 17
```

Note that instead of 15 we printed the concatenation of $s_1$ (`fizz`) and $s_2$ (`buzz`) because 15 is divisible both by $d_1$ (3) and by $d_2$ (5). Also note that you cannot print `buzzfizz` for 15 because you cannot change the order of the $s_i$ when concatenating them.

## Subproblem Q1

Samko has written his own implementation of a Generalized Fizz Buzz program. However, Samko's implementation has some additional constraints: It can only handle inputs where $k \leq 2$ and each of the strings $s_i$ has length exactly 4.

Samko has shown you a sequence of labels $\ell_1, \ldots, \ell_n$. Find the largest $q$ such that the prefix $\ell_1, \ldots, \ell_q$ can be the output of Samko's program.

## Subproblem Q2

Monika has also written her own implementation of a Generalized Fizz Buzz program. Monika's program can only handle inputs where each of the strings $s_i$ has between 1 and 25 characters, inclusive. (Her program can handle arbitrarily large values of $k$ and $d_i$.)

Monika has shown you a sequence of labels $\ell_1, \ldots, \ell_n$. Find the largest $q$ such that the prefix $\ell_1, \ldots, \ell_q$ can be the output of Monika's program.

 

## Input specification

The first line of the input file contains an integer $t$ specifying the number of test cases. Each test case is preceded by a blank line.

Each test case consists of two lines. The first line contains a positive integer $n \leq 1000$: the number of labels in the sequence you were given. The second line contains the sequence: $n$ non-empty tokens $\ell_1, \ldots, \ell_n$ separated by single spaces. Each token will only contain digits and lowercase English letters.

## Output specification

For each test case output one line with the corresponding integer $q$.

Make sure that in **subproblem Q1** the $q$ you output corresponds to Samko's program, while in **subproblem Q2** your output should correspond to Monika's program.

## Example

input

```
5

7
mlem mlem mlem mlem mlem mlem fail

7
1 qizz quzz qizz 5 quzzqizz 007

8
1 2 3 4 hunter2 is my password

2
haha hahahaha

9
1 bat 3 batman robin batbat 7 batmanman 9
```

output for Q1

```
6
5
4
2
1
```

output for Q2

```
6
5
4
2
9
```

- In the first example, the first six tokens are the output of a Generalized Fizz Buzz program for $k = 1$, $d_1 = 1$, and $s_1 =$ `mlem`. Both Samko's and Monika's program can handle this input, so for each of them the answer is 6.
- In the second example, it's clear that 2 must be replaced with `qizz` and 3 with `quzz`. The first five tokens match that sequence, but the next one should be `qizzquzz` instead of `quzzqizz`, and the last one should be `7` instead of `007`.
- In the third example, `hunter2` is wrong because it contains both letters and numbers.
- In the fourth example, $d = (1, 2)$ and $s =$ (`haha`, `haha`). Note that the strings $s_i$ do not have to be distinct.
- In the fifth example, `bat` is already wrong in Q1 because Samko's program cannot handle a three-character string. In Q2 the whole sequence is correct because it's possible that 2 was replaced by `bat`, 4 was replaced by `man`, 5 by `robin`, 6 also by `bat`, and 8 also by `man`.

 

# Problem R: Raw data

We received an anonymous tip that a famous hacker group, the Inquisitors of Poorly Secured Code, will try to remotely disable your web browser during tomorrow's contest. We want to make sure you'll still be able to reach us even if your web browser stops working.

## Problem specification

To solve the **easy subproblem R1**, send three HTTP POST requests to the address `https://ipsc.ksp.sk/2018/practice/problems/r1`. You must be logged in (the request must have the correct cookies). The POST request body can contain anything.

To solve the **hard subproblem R2**, send three HTTP POST requests to the address `https://ipsc.ksp.sk/2018/practice/problems/r2`. You must be logged in *and* you can't use a web browser. The POST request body can contain anything.

## Input specification

There is no input.

## Output specification

There is no output. You don't have to submit any files, you will automatically get an **OK** verdict when you send the correct requests.

 