Pancakes!

A *filled pancake* consists of a *pancake* and its *filling*. In this problem, you will be given a list of ingredients you have and your task is to prepare as many filled pancakes as possible.

Given 8 cups of milk, 8 egg yolks, 4 tablespoons of sugar, 1 teaspoon of salt, and 9 cups of flour, we can prepare 16 pancakes. For the purpose of this problem, we assume that these quantities scale arbitrarily: for any $x \geq 0$, if you have x times as much of every ingredient, you can prepare $\lfloor 16x \rfloor$ pancakes.

Given the necessary ingredients, you can prepare the following fillings:

- banana pancake: 1 banana
- strawberry pancake: 30 g of strawberry jam
- chocolate pancake: 25 g of chocolate spread
- walnut pancake: 10 walnuts
- $\star$ banana pancake with chocolate: 3/4 of a banana, 10 g of chocolate spread
- $\star$ walnut pancake with chocolate: 7 walnuts, 10 g of chocolate spread
- $\star$ grand mix pancake: 1/3 of a banana, 3 walnuts, 5 g of strawberry jam, and 5 g of chocolate spread

Banana pieces can be combined. For example, you can use three pieces of 1/3 banana each to create a banana pancake. **For the easy subproblem, ignore the pancakes marked by a star**.

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 of a test case contains five integers c_m, y, s_{su}, s_{sa}, and f, meaning that you have c_m cups of milk, y egg yolks, s_{su} tablespoons of sugar, s_{sa} teaspoons of salt, and f cups of flour.

The second line contains four integers b, g_s, g_c, and w, meaning that you have b bananas, g_s grams of strawberry jam, g_c grams of chocolate spread, and w walnuts.

All quantities are between 0 and 10^6, inclusive.

Output specification

For each test case, output a single line containing a single integer – the maximum number of filled pancakes you can prepare.

Example for the easy subtask

input

```
2

16 16 8 2 17
10 47 100 19

16 16 8 2 17
10 470 100 19
```

output

```
16
30
```

In the first test case, we are limited by the number of fillings: we can make 10 banana pancakes, 1 strawberry pancake, 4 chocolate pancakes, and 1 walnut pancake. In the second test case, we only have enough flour for 30 pancakes.

If we solved the same input in the hard subtask, the first answer would be 17: First, make 3 grand mix pancakes. You are left with 9 bananas, 32 g of strawberry jam, 85 g of chocolate spread, and 10 walnuts. That is enough for another 9+1+3+1=14 easy pancakes.

Quality of Service

For technical reasons, this problem statement is only available online. Sorry for the inconvenience. The example input and output is available, though:

Example

input

```
3

1

2

47
```

output

```
2
47
42
```

Rotate to divide

We have found an interesting paragraph in an article about physicist Freeman Dyson[1].

> A group of scientists will be sitting around the cafeteria, and one will idly wonder if there is an integer where, if you take its last digit and move it to the front, turning, say, 112 to 211, it's possible to exactly double the value. Dyson will immediately say, "Oh, that's not difficult," allow two short beats to pass and then add, "but of course the smallest such number is 18 digits long." When this happened one day at lunch, William Press remembers, "the table fell silent; nobody had the slightest idea how Freeman could have known such a fact or, even more terrifying, could have derived it in his head in about two seconds."

Problem specification

You will solve a slightly different problem. Instead of doubling the number, your goal is to make it k times **smaller**. We will also consider numbers in different bases. Your task is, given a base b and number k, to find the smallest number in base b which becomes k times smaller after a single rotation. Since the number can be very large, you only need to output its length and its first digit.

The rotation is done by taking the last digit and moving it to the front of the number. Rotation can only be applied to a number that has at least two digits. The last digit can also be a zero. For example, 110 rotates to 011, which is the same as 11.

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 is a single line with two integers: the base $b \geq 2$ and the ratio $k \geq 1$.

In the easy data set we have $b, k \leq 8$, in the hard data set $b, k \leq 2,000,000$.

Output specification

Print one line for each test case. If there is no number in base b such that by rotating it we get a k times smaller number, output "`NO`". Otherwise, find the smallest such number, output its length and its first digit. The first digit should be output in base 10, as shown in the last example below.

Kindly note that the length will always be at least 2, as single-digit numbers cannot be rotated.

Example

input

```
3

5 2

3 2

16 7
```

output

```
2 3
NO
3 12
```

In the first test case, $(31)_5 = (16)_{10}$ becomes $(13)_5 = (8)_{10}$. This is the smallest solution, so we output length 2 and first digit 3. In the last test case, $(c71)_{16}$ is rotated to $(1c7)_{16}$, so the length is 3 and the first digit of the original number has value 12.

[1] `http://www.nytimes.com/2009/03/29/magazine/29Dyson-t.html?pagewanted=8`