1. **Primes Again**

Matt loves primes and created **q** queries where each query takes an integer **n**.

For each **n**, count the *maximum number of distinct prime factors* of any number in the inclusive range **[1, n]**.

Example:
n = 100

For the numbers in the inclusive range **[1, 100]**, there are few numbers with maximum number of *distinct prime factors*. Couple of them are 30 and 42.

- 30 can be written as 2 x 3 x 5 (3 distinct prime factors)
- 42 can be written as 2 x 3 x 7 (3 distinct prime factors).

So, the answer for 100 is 3.

Note that, other numbers like 60, 84 and 90 can be written as the product more prime factors but, the *maximum number of distinct prime factors* we can get is 3 in the above example.

Input Format:
First line of input contains **q** denoting number of queries.
Each line of next **q** lines contains an integer **n**.

Output Format:
Print the output according to the description.

Constraints:
$1 \quad \leq q \leq 10^5 1 \leq N \leq 10^{18}$

Sample I/O:

Input 1:
2
100
500
Output 1:
3
4

Input 2:
3
9
29
5
Output 2:
2
2
1

2. **Duck Number**

A number **N** is called a **Duck Number** it contains at least one 0 (zero) in it.

You're given a number **N**, determine if it's a duck number or not.

Input Format:
Only line of input contains a number **N**.

Output Format:
Print **Duck** if the given number is duck number, else print **Not Duck**.

Constraints:
$-10^9 \leq N \leq 10^9$

Sample I/O:

Input 1:
101
Output 1:
Duck

Input 2:
137
Output 2:
Not Duck

Input 3:
204407
Output 3:
Duck

Input 4:
654321
Output 4:
Not Duck

3. Magical Ten

In the game show "Magical Ten", there are **N** participants numbered from **1** to **N**, each of whom submits one response. The i^{th} response is A_i words long and has quality B_i. No two responses have the same quality, and at least one response has word length at most 10.

The winner of the show is the response which has the **highest quality** out of all responses that are **not longer than 10** words. Which response is the winner?

Input Format:
First line of input contains an integer **N** denoting number of participants.
Each line of Next **N** lines contain 2 space separated integers denoting length of resonse and quality respsectively.

Output Format:
Print the output according to the problem statement.

Constraints:
$1 \quad \leq N \leq 501 \leq A[i], \quad B[i] \quad \leq 50$

Sample I/O:

Input 1:
5
7 2
12 5
9 3
9 4
10 1
Output 1:
4

Input 2:
3
1 2
3 4
5 6
Output 2:
3

Input 3:
1
1 43
Output 3:
1

Explanation:
For input1,
Response 1: 7 words, quality 2
Response 2: 12 words, quality 5
Response 3: 9 words, quality 3
Response 4: 9 words, quality 4
Response 5: 10 words, quality 1

We can see that the responses with indices 1, 3, 4, and 5 have lengths not exceeding 10 words. Out of these responses, the winner is the one with the highest quality.

4. Good alphabet

An alphabet **alpha** can be called as a good alphabet it appears *exactly* **P** times in the given string **S**, where **P** is the position of **alpha** in English Alphabets.

Given a string **S** and an alphabet **alpha** find out if it's good or not.

Note:

- Both **alpha** and **S** should be considered **case-insensitive**.

Input Format:
First line of input contains a string **S**.
Second line of input contains an alphabet **alpha**.

Output Format:
Print **YES** if **alpha** is a good alphabet, else print **NO**.

Constraints:
$1 \leq len(S) \leq 10^3$

Sample I/O:

Input 1:
a mystery
a
Output 1:
YES

Input 2:
What AN Elegant eVENINg we HAd
e
Output 2:
YES

Input 3:
What AN Elegant eVENINg we HAd
E
Output 3:
YES

Input 4:
What AN Elegant eVENINg we HAd
d
Output 4:
NO

Explanation:

For Input 1,
Given alphabet **a** is present in the string for once. One is the position of **a**/**A** in English Alphabets.

For Input 4,
Given alphabet **d** is present only once in the given string. It has to appear for a total of 4 times to become a good alphabet as the position of **d**/**D** is 4 in English Alphabets.