A numeric string, s, is beautiful if it can be split into a sequence of two or more positive integers, $a[1], a[2], \ldots, a[n]$, satisfying the following conditions:

1. $a[i] - a[i-1] = 1$ for any $1 < i \leq n$ (i.e., each element in the sequence is 1 more than the previous element).
2. No $a[i]$ contains a leading zero. For example, we can split $s = 10203$ into the sequence $\{1, 02, 03\}$, but it is not beautiful because 02 and 03 have leading zeroes.
3. The contents of the sequence cannot be rearranged. For example, we can split $s = 312$ into the sequence $\{3, 1, 2\}$, but it is not beautiful because it breaks our first constraint (i.e., $1 - 3 \neq 1$).

The diagram below depicts some beautiful strings:

Perform q queries where each query consists of some integer string s. For each query, print whether or not the string is beautiful on a new line. If it is beautiful, print `YES x`, where x is the first number of the increasing sequence. If there are multiple such values of x, choose the smallest. Otherwise, print `NO`.

Function Description

Complete the separateNumbers function in the editor below.

separateNumbers has the following parameter:

- s: an integer value represented as a string

Prints

- string: Print a string as described above. Return nothing.

Input Format

The first line contains an integer q, the number of strings to evaluate.

Each of the next q lines contains an integer string s to query.

Constraints

- $1 \leq q \leq 10$
- $1 \leq |s| \leq 32$
- $s[i] \in [0 - 9]$

```
#
# Complete the 'separateNumbers' function below.
#
# The function accepts STRING s as parameter.
#
#
def separateNumbers(s):
    result = 'NO'

    for idx in range(len(s)):
        n_digits_number = int(s[:idx+1])
        current = str(n_digits_number)

        if len(current) != len(s):
            next_digit = n_digits_number + 1

            if len(current) < len(s):
                current += str(next_digit)

            while len(current) < len(s):
                next_digit += 1
                current += str(next_digit)

            if current == s:
                result = 'YES ' + str(n_digits_number)
                break

    print(result)
    # return result
```

Line: 37 Col: 20

Upload Code as File Test against custom input Run Code Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

Test case 15
Test case 16
Test case 17
Test case 18

Compiler Message

```
Success
```

Input (stdin) Download

```
7
1234
91011
99100
```