Documentation for the isNumber Function

Overview

A valid number can be categorized as either an integer number or a decimal number, with an optional exponent part.

- Integer Number: An optional sign (+ or -) followed by digits.
- Decimal Number: An optional sign (+ or -) followed by one of the following:
    - Digits followed by a dot (.).
    - Digits followed by a dot (.) and more digits.
    - A dot (.) followed by digits.
- Exponent Part: An exponent notation (e or E) followed by an optional sign (+ or -) and an integer number.

Regular Expression Breakdown

The function uses a regular expression (regex) to match the input string against the valid number formats. Here’s a detailed breakdown of the regex:

- ^ : Start of the string.
- [+-]? : An optional sign (+ or -).
- (: Start of the main number part:
- (d+.d) : Digits followed by a dot (with optional digits).
- | : OR
- (.d+) : Dot followed by digits.
- | : OR
- (d+(.d)?) : Digits (with optional dot and more digits).
-) : End of the main number part.
- ([eE][+-]?d+)? : Optional exponent part:
- [eE] : Exponent notation (e or E).

- [+-]? : Optional sign (+ or -).
- d+ : One or more digits.
- $: End of the string.

Examples

Here are some examples to illustrate how the function works:

- isNumber("0") returns True : "0" is a valid integer number.
- isNumber("e") returns False : "e" is not a valid number.
- isNumber(".") returns False : A single dot is not a valid number.
- isNumber("2e10") returns True : "2e10" is a valid number with an exponent.
- isNumber("-90E3") returns True : "-90E3" is a valid number with an exponent.
- isNumber("1e") returns False : "1e" is not a valid number because the exponent part is incomplete.
- isNumber("99e2.5") returns False : "99e2.5" is not a valid number because the exponent part must be an integer.
- isNumber("--6") returns False : "--6" is not a valid number due to multiple signs.
- isNumber("-+3") returns False : "-+3" is not a valid number due to multiple signs.
- isNumber("95a54e53") returns False : "95a54e53" is not a valid number due to the presence of non-numeric characters.

Problem Statement

The task is to determine whether a given string s is a valid number. The definition of a valid number includes integers and decimal numbers, which may optionally be followed by an exponent.

Valid Numbers

Examples of valid numbers include:

- Integers: "2" , "0089"
- Signed integers: "-0.1" , "+3.14"
- Decimals: "4." , "-.9"
- Scientific notation: "2e10" , "-90E3" , "3e+7" , "+6e-1" , "53.5e93" , "-123.456e789"

Invalid Numbers

Examples of invalid numbers include:

- Non-numeric strings: "abc" , "1a"
- Improper exponent notation: "1e" , "e3" , "99e2.5"
- Multiple signs: "--6" , "-+3"
- Embedded letters: "95a54e53"

Formal Definitions

- **Integer Number :** An optional sign ('-' or '+') followed by digits (0-9).
- **Decimal Number :** An optional sign followed by one of the following:
    - Digits followed by a dot ('.')
    - Digits followed by a dot and more digits
    - A dot followed by digits
- **Exponent :** An exponent notation ('e' or 'E') followed by an integer number (optional sign followed by digits).

Example Cases

Example 1

- *Input :* "0"
- *Output :* true

Example 2

- *Input :* "e"
- *Output :* false

Example 3

- *Input :* "."
- *Output :* false

Constraints

- 1 <= s.length <= 20
- s consists of only English letters (both uppercase and lowercase), digits (0-9), plus ('+'), minus ('-'), or dot ('.').

Solution

The solution uses a regular expression to validate the format of the number. The regular expression accounts for the various forms of valid numbers as per the definitions above.

Explanation

1. **Regular Expression Definition :** The number_regex is defined using the re.VERBOSE flag for readability. It captures the following:

- Optional sign ([+-]?)
- Main number part which can be:
  - Digits followed by a dot with optional digits (d+.d)
  - Dot followed by digits (.d+)
  - Digits with optional dot and more digits (d+(.d)?)
  - Optional exponent part ([eE][+-]?d+)

2. **Matching the Input String :** The match method is used to determine if the entire input string matches the defined pattern.