VMES Grinds

VMES Grinds

Pilot Program - Week 1 - Day 2

Arrays, Numbers, Strings

Leetcode: 8
String to Integer (atoi)

ZWE

8. String to Integer (atoi)

Solved

Medium | Topics | Companies

Implement the `myAtoi(string s)` function, which converts a string to a 32-bit signed integer.

The algorithm for `myAtoi(string s)` is as follows:

1. **Whitespace**: Ignore any leading whitespace (`" "`).
2. **Signedness**: Determine the sign by checking if the next character is `'-'` or `'+'`, assuming positivity if neither present.
3. **Conversion**: Read the integer by skipping leading zeros until a non-digit character is encountered or the end of the string is reached. If no digits were read, then the result is 0.
4. **Rounding**: If the integer is out of the 32-bit signed integer range $[-2^{31}, 2^{31} - 1]$, then round the integer to remain in the range. Specifically, integers less than -2^{31} should be rounded to -2^{31}, and integers greater than $2^{31} - 1$ should be rounded to $2^{31} - 1$.

Return the integer as the final result.

Example 1:

Input: s = "42"

Output: 42

Explanation:

```
The underlined characters are what is read in and the caret is the current reader position.
Step 1: "42" (no characters read because there is no leading whitespace)
         ^
Step 2: "42" (no characters read because there is neither a '-' nor '+')
         ^
Step 3: "42" ("42" is read in)
           ^
```

Example 2:

Input: s = " -042"

Output: -42

Explanation:

```
Step 1: "   -042" (leading whitespace is read and ignored)
            ^
Step 2: "   -042" ('-' is read, so the result should be negative)
             ^
Step 3: "   -042" ("042" is read in, leading zeros ignored in the result)
```

Example 3:

> **Input:** s = "1337c0d3"
>
> **Output:** 1337
>
> **Explanation:**
>
> Step 1: "1337c0d3" (no characters read because there is no leading whitespace)
>
> ^
>
> Step 2: "1337c0d3" (no characters read because there is neither a '-' nor '+')
>
> ^
>
> Step 3: "1337c0d3" ("1337" is read in; reading stops because the next character is a non-digit)
>
> ^

Example 4:

> **Input:** s = "0-1"
>
> **Output:** 0
>
> **Explanation:**
>
> Step 1: "0-1" (no characters read because there is no leading whitespace)
>
> ^
>
> Step 2: "0-1" (no characters read because there is neither a '-' nor '+')
>
> ^
>
> Step 3: "0-1" ("0" is read in; reading stops because the next character is a non-digit)
>
> ^

Example 5:

> **Input:** `s = "words and 987"`
>
> **Output:** `0`
>
> **Explanation:**
>
> Reading stops at the first non-digit character 'w'.

Constraints:

- `0 <= s.length <= 200`
- `s` consists of English letters (lower-case and upper-case), digits (`0-9`), `' '`, `'+'`, `'-'`, and `'.'`.

Explanation

Thought Process

- Linear iteration through the string
- Check constraints

Code

VMES Grinds

8_String_to_Integer_(atoi).py - C:/Workspace/LeetCode/Algorithms/8_String_to_Integer_(atoi).py (3.12.3)

File Edit Format Run Options Window Help

```python
def myAtoi(s: str) -> int:
    num = '0'
    neg = False
    check = {' ':True,'-':False,'+':False,'digit':False}

    for i in range(len(s)):
        if s[i].isdigit():
            num += s[i]
            if check[' ']:
                check[' '] = False
            if not check['digit']:
                check['digit'] = True
            if not check['-']:
                check['+'] = True
        elif check['digit']:
            break
        else:
            if (s[i] == '+' or s[i] == '-') and not check['+'] and not check['-']:
                check[s[i]] = True
                check[' '] = False
                if s[i] == '-':
                    neg = True
            elif (s[i].isspace() and not check[' ']) or (not s[i] in check) or ((s[i] == '+' or s[i] == '-') and (check['+'] or check['-'])):
                break

    num = int(num)
    if neg:
        num = -num
    if num < -2**31:
        return -2**31
    elif num > (2**31)-1:
        return (2**31)-1
    return num
```

8_String_to_Integer_(atoi)_v2.py - C:\AU\LeetCode\Algorithms\8_String_to_Integer_(atoi)_v2.py (3.12.3)

File Edit Format Run Options Window Help

```python
def myAtoi(self, s: str) -> int:
    num = '0'
    sign = ""
    space = True
    digit = False

    for i in s:
        if i.isdigit():
            num += i
            if not digit:
                space = False
                digit = True
        elif digit:
            break
        else:
            if sign == "" and (i == '-' or i == '+'):
                sign = i
                space = False
            elif (i.isspace() and not space) or (sign != "" and (i == '-' or i == '+')) or (not i.isspace() and i != '-' and i != '+'):
                break

    num = int(num)
    if sign == "-":
        num = -num
    if num < -2**31:
        return -2**31
    elif num > (2**31)-1:
        return (2**31)-1
    return num
```

VMES Grinds

Time and Space Complexity Analysis

8_String_to_Integer_(atoi)_v2.py - C:\AU\LeetCode\Algorithms\8_String_to_Integer_(atoi)_v2.py (3.12.3)

File Edit Format Run Options Window Help

```python
def myAtoi(self, s: str) -> int:
    num = '0'
    sign = ""
    space = True
    digit = False

    for i in s:
        if i.isdigit():
            num += i
            if not digit:
                space = False
                digit = True
        elif digit:
            break
        else:
            if sign == "" and (i == '-' or i == '+'):
                sign = i
                space = False
            elif (i.isspace() and not space) or (sign != "" and (i == '-' or i == '+')) or (not i.isspace() and i != '-' and i != '+'):
                break

    num = int(num)
    if sign == "-":
        num = -num
    if num < -2**31:
        return -2**31
    elif num > (2**31)-1:
        return (2**31)-1
    return num
```

← reason why O(n²) with just 1 for loop

Space - O(n)

Time - O(n^2)

Key Takeaway

- Strings in Python → immutable
- In string concatenation, Python actually creates a new string by copying the entire previous string
- Each concatenation grows costlier as string gets longer → $O(n^2)$

VMES Grinds

Thank You !