

AP ASSIGNMENT - 4

Name : Naman Kumar

UID : 22ICS10001

Class : FL_IOT_604 (A)

1763. Longest Nice Substring

```
from collections import defaultdict

class Solution:
    def longestNiceSubstring(self, s):
        len_s = len(s)
        if len_s <= 1:
            return ''

        char_to_freq_map = defaultdict(int)
        for c in s:
            char_to_freq_map[c] += 1

        is_broken = False
        i = 0
        while (i < len(s)):
            if s[i].islower() and s[i].upper() in char_to_freq_map.keys():
                pass
            elif s[i].isupper() and s[i].lower() in char_to_freq_map.keys():
                pass
            else:
                is_broken = True
                break
            i += 1

        if not is_broken:
            return s

        longest_nice_substr_1 = self.longestNiceSubstring(s[:i])
```

```

longest_nice_substr_2 = self.longestNiceSubstring(s[i+1:])

if len(longest_nice_substr_1) >= len(longest_nice_substr_2):
    return longest_nice_substr_1
else:
    return longest_nice_substr_2

```

The screenshot shows the LeetCode interface for the problem "Longest Nice Substring". The solution is written in Python3 and is marked as "Accepted". The runtime is 7 ms, beating 52.10% of other solutions. The memory usage is 17.76 MB, beating 61.95% of other solutions. A line graph shows the runtime distribution across different time intervals. The code is as follows:

```

1 from collections import defaultdict
2
3 class Solution:
4     def longestNiceSubstring(self, s):
5         len_s = len(s)
6         if len_s <= 1:
7             return s
8
9         char_to_freq_map = defaultdict(int)
10        for c in s:
11            char_to_freq_map[c] += 1
12
13        is_broken = False
14        i = 0
15        while (i < len(s)):
16            if s[i].islower() and s[i].upper() in char_to_freq_map.keys():
17                pass
18            elif s[i].isupper() and s[i].lower() in char_to_freq_map.keys():
19                pass

```

The test result shows that the solution is "Accepted" with a runtime of 0 ms. The input string is "YazaAay".

190. Reverse Bits

```

class Solution:
    def reverseBits(self, n):
        out = 0
        for i in range(32):
            out = (out << 1) ^ (n & 1)
            n >>= 1
        return out

```

Screenshot of the LeetCode interface for the "Reverse Bits" problem. The problem is marked as "Accepted" with 600/600 testcases passed. The submission was made by user 'n.' on Feb 20, 2025 at 22:34.

Runtime: 34 ms, Beats 86.21%
Memory: 17.70 MB, Beats 54.89%

The code editor shows the following Python3 solution:

```
class Solution:
    def reverseBits(self, n):
        out = 0
        for i in range(32):
            out = (out << 1)^(n & 1)
            n >>= 1
        return out
```

The Testcase section shows the input: `n = 00000010100101000001111010011100`.

191. Number of 1 Bits

```
class Solution:
    def hammingWeight(self, n: int) -> int:
        bin_n = bin(n)[2:]
        count = 0
        for i in bin_n:
            if i == '1':
                count += 1
        return count
```

Screenshot of a LeetCode submission for the problem "Number of 1 Bits" (LeetCode 191). The submission is accepted, showing a runtime of 0 ms and memory usage of 17.66 MB. The code is in Python3 and uses a built-in function to calculate the number of 1 bits.

Runtime: 0 ms | Beats 100.00%
Memory: 17.66 MB | Beats 80.88%

Code:

```
class Solution:
    def hammingWeight(self, n: int) -> int:
        bin_n = bin(n)[2:]
        count = 0
        for i in bin_n:
            if i == '1':
                count += 1
        return count
```

Testcase: Accepted Runtime: 0 ms
Case 1: Input: n = 11

53. Maximum Subarray

```
class Solution:
    def maxSubArray(self, nums: List[int]) -> int:
        res = nums[0]
        total = 0

        for n in nums:
            if total < 0:
                total = 0

            total += n
            res = max(res, total)

        return res
```

leetcod.com/problems/maximum-subarray/

Problem List < > > >

Description Accepted x Editorial Solutions

All Submissions

Accepted 210 / 210 testcases passed
n. submitted at Feb 20, 2025 22:36

Runtime
52 ms Beats 71.45%
Analyze Complexity

Memory
32.03 MB Beats 65.92%

Code Python3

```
class Solution:
    def maxSubArray(self, nums: List[int]) -> int:
        res = nums[0]
        total = 0
        for n in nums:
            if total < 0:
                total = 0
            total += n
            res = max(res, total)
        return res
```

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums = [-2,1,-3,4,-1,2,1,-5,4]

240. Search a 2D Matrix II

```
class Solution:
    def searchMatrix(self, mat: List[List[int]], target: int) -> bool:

        m=len(mat)
        n=len(mat[0])

        for i in range(m):
            if mat[i][0]<=target and mat[i][-1]>=target:
                lo=0
                hi=n
                while (lo<hi):
                    mid=(lo+hi)//2

                    if mat[i][mid]==target:
                        return True
                    elif mat[i][mid]<target:
```

```

        lo = mid + 1

    else:
        hi = mid

return False

```

leetcod.com/problems/search-a-2d-matrix-ii/

Problem List < > < >

Description Accepted Editorial Solutions

All Submissions

Accepted 130 / 130 testcases passed
n. submitted at Feb 20, 2025 22:37

Runtime
150 ms / Beats 30.74%
Analyze Complexity

Memory
24.06 MB / Beats 50.88%

Code Python3

```

class Solution:
    def searchMatrix(self, mat: List[List[int]]) -> bool:
        m=len(mat)
        n=len(mat[0])

        for i in range(m):
            if mat[i][0]<target and mat[i][-1]>=target:
                lo=0
                hi=n
                while (lo<hi):
                    mid=(lo+hi)//2
                    if mat[i][mid]==target:
                        return True
                    elif mat[i][mid]<target:
                        lo = mid + 1
                    else:
                        hi = mid

```

Testcase Test Result

Accepted Runtime: 53 ms

Case 1 Case 2

Input

matrix =

[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]

372. Super Pow

```

class Solution:
    def superPow(self, a: int, b: List[int]) -> int:
        mod = 1337
        p = 1
        for i in b:
            p+=str(i)
        p=int(p)
        return pow(a,p,mod)

```

← → ↻ leetcode.com/problems/super-pow/

🏠 Gmail 📺 YouTube 🎓 CUIIMS 📧 Outlook 🖱️ Hand Gesture Con... ⚡ yes 📺 movies 🧑‍🎓 DSA

🔖 All Bookmarks

🏠 Problem List < > 🔍

🔧 Run 📄 Submit 🗑️ 📄

🔧 ⚙️ 🔥 0 👤 Premium

Description Accepted × Editorial Solutions

← All Submissions 🔗

Accepted 57 / 57 testcases passed

📄 Solution

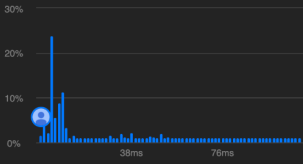
🕒 Runtime

1 ms Beats 99.22%

🔍 Analyze Complexity

💾 Memory

17.97 MB Beats 39.69%



Code Python3

```
class Solution:
    def superPow(self, a: int, b: List[int]) -> int:
        mod = 1337
```

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

a =
2