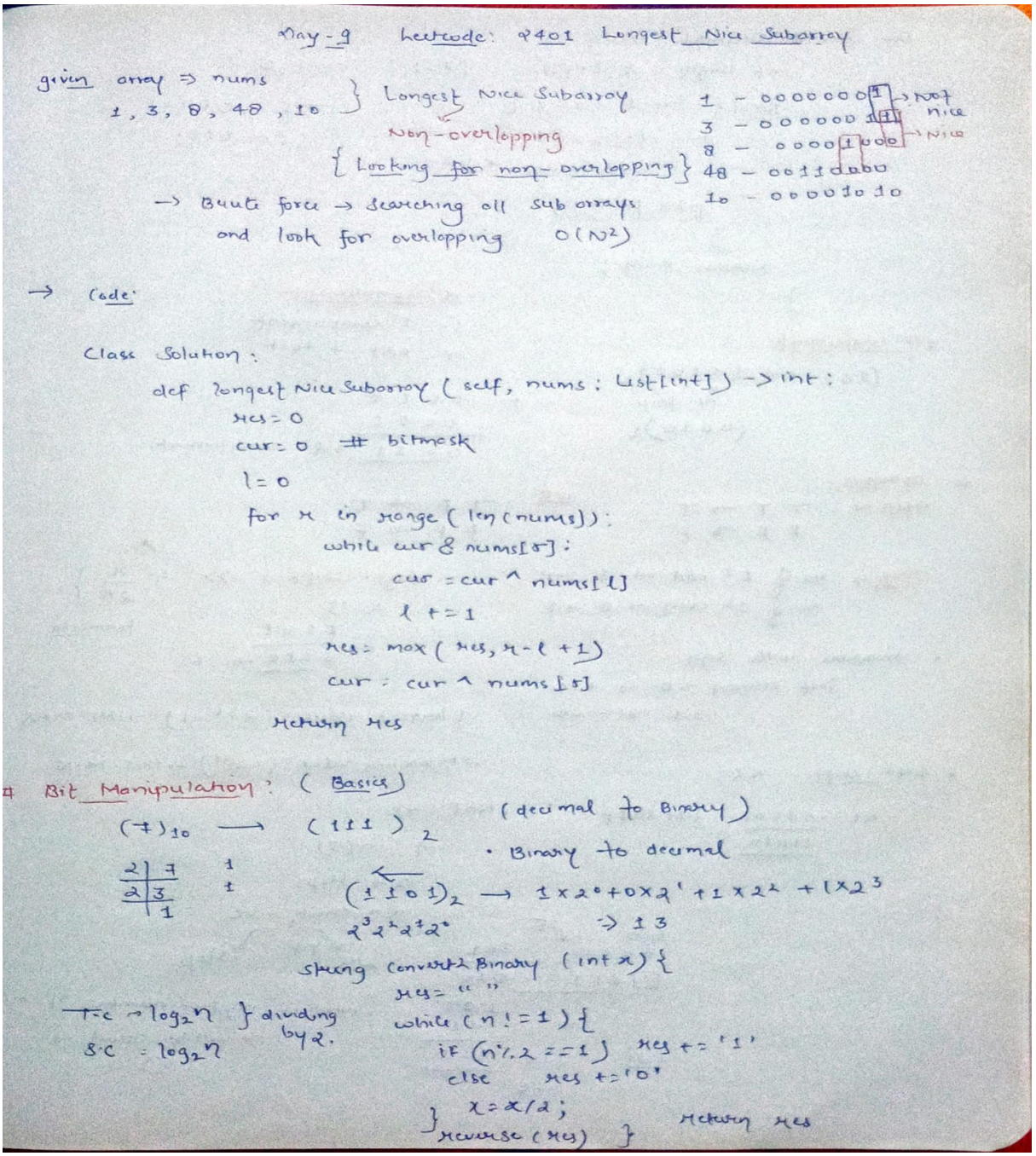
Day_9: DSA



Day - 9 leetcode: 2401 Longest Nice Subarray

given array ⇒ nums
1, 3, 8, 48, 10 } Longest Nice Subarray
Non-overlapping
{ Looking for non-overlapping }

1 - 00000001 → Not Nice
3 - 00000011
8 - 00001000 → Nice
48 - 00110000
10 - 00001010

→ Brute force → Searching all sub arrays and look for overlapping O(N²)

→ Code:

```
Class Solution:
    def longestNiceSubarray(self, nums: List[int]) -> int:
        res = 0
        cur = 0  # bitmask
        l = 0
        for r in range(len(nums)):
            while cur & nums[r]:
                cur = cur ^ nums[l]
                l += 1
            res = max(res, r-l+1)
            cur = cur ^ nums[r]
        return res
```

Bit Manipulation: (Basics)

(decimal to Binary)
(7)₁₀ → (111)₂

2 | 7	1
2 | 3	1
1	

• Binary to decimal
(1101)₂ → 1x2⁰ + 0x2¹ + 1x2² + 1x2³
2³ 2² 2¹ 2⁰ → 13

```
string convert2Binary(int n){
    res = ""
    while(n != 1){
        if(n%2 == 1) res += '1'
        else res += '0'
        n = n/2;
    }
    reverse(res)
}
return res
```

T.C → $\log_2 n$ } dividing by 2.
S.C = $\log_2 n$

```
int convert2Decimal (string x) {
    int length = x.length ,  p2 = 1     num = 0
    for (i = len-1 → 0) {                        T.C → O(len)
        if (x[i] == '1')                         S.C → O(1)
            num = num + p2;
        p2 = p2 × 2;
    }
    return num;
}
```

1st compliment:

$(13) \rightarrow (1101)_2$

↓ flip

$(0010)_2$

2's compliment:-

1. 1s complement
2. add 1 to it

```
  0 0 1 0
+ 0 0 0 1
---------
  0 0 1 1  → 2's compliment
```

Operators:

AND → T T → T ; F F → F

OR → T F → T ; F F → F

XOR → no. of 1s odd → 1 → T ; no. of 1s even → 0 → F

- Right shift → >> = $\frac{x}{2^k}$ } Formula ★

x = 13

```
  1 1 0 1
→ 0 1 1 0 → 6
```

- Numbers with sign

last column → 0 → +ve ; → 1 → -ve

} largest value $(2^{31} - 1)$ → INT-MAX

→ Minimum value (-2^{31}) → INT-MIN

- Left shift <<

```
00--01101  left shift
11010
```

- NOT (~)

x = ~(5)

1 → flip

2 → check -ve → yes → 2's ; → stop

```
sign     ↙5
0 0 0 0 -- 1 0 1
[1] 1 1 1 ---0 1 0
→          1 0 1
            + 1
[1]   --  --. 1 1 0
       -6
```

- what if negative number x = ~(-6) it will be stored as (5)