Data Structures & Algorithms (DSLGO)

Topics:

- *Space Complexity:*
- *Online Judge*
- ***3 Problems → Array (Interview)***
- *Map*

Space Complexity (Memory Complexity):

If I have two programs, by looking at the program I can tell which program is faster as per the Time Complexity method. Now, O (n^2) and O (log n), O (log n) is faster.

RAM – whenever we create a program, all the variables mentioned in the program (a = 4, b = 5, c = 6) are stored in the memory. If you have an array of

a = [o] * 10000

let's suppose the memory consumed by one integer is *x*, then the memory consumed would be 10000 * *x*.

The memory is constant so for d = (a + b) * (2 + c), the total cost would be O(4) i.e., a, b, c and d.

The space complexity of a list is O(n). A list of size 'n' will always take 'n' spaces. If one variable takes one place, 'n' variables will take 'n' places.

We take an estimate of extra space we take in executing a program.

NOTE: Both time and space complexity are denoted by Big-O

Online Judge:

In a program, let's say, given 'n' value, to check if it even or odd. For a program:

```
if n == 1:
    print('odd')
if n == 2:
    print('even')
if n == 3:
    print('odd')
```

If an **INPUT n = 5**, the program will not give us any output.

So, there is an online judge, which has few defined test cases. We will write our code and we will give to the online judge and these test cases are given to the online judge and it already knows the corresponding output as well.

It will check the answer it has and the answer the code we submitted is giving.

So, we will submit our program to online judge who already knows the answer and it will check, if the answer given is right or wrong.

Judge is computer program to which we will give our program and it will check if our program is working correctly or not. We have to submit our code as input and our output will be compared to the answer.

The popular judge available in the market is

- Leetcode
- Practice.geeksforgeeks
- Hackerrank
- Hackerearth
- Codechef
- Codeforces
- Atcoder

Leetcode is the best from the interview perspective. Finish 400 problems from it and you can easily land a job (LPA – 11).

Top 100 Liked Questions

Top Interview Questions

Finish them without fail. First finish easy & medium, then go for hard.

First Problem: (LeetCode)

Given a non-empty array of decimal digits representing a non-negative integer, increment one to the integer.

The digits are stored such that the most significant digit is at the head of the list, and each element in the array contains a single digit.

You may assume the integer does not contain any leading zero, except the number 0 itself.

Example 1:

INPUT: digits = [1, 2, 3]
OUTPUT: [1, 2, 4]
Explanation: The array represents the integer 123.

Example 2:

INPUT: digits = [4, 3, 2, 1]
OUTPUT: [4,3,2,2]
Explanation: The array represents the integer 4321.

Example 3:

INPUT: digits = [0]
OUTPUT = [1]

NOTE: if you are not able to understand the question, look at the examples.

In an array A = [1, 2, 3, 4] and if add 1 to it, you will get an answer, [1, 2, 3, 5].

Solution:

A = [1, 2, 5]

class Solution:

def plusOne(self, digits: List[int] → List[int]

- Digits is a list of integers
- Do not delete this line, but code after it.
- "**:**" **means,** type of data
  "→" tells the answer it is supposed to return.

How can we convert an array into a string? Join command joins all the elements in an array and makes it a string.

Ex: a = [“1”, “2”, “5”]

“ “ a.join(a) >>> 1 2 3 4

B = [str(x) for x in a] → to type cast it into a string. This will convert a the list into a string. **The “.join” can only joins an array of string and not integers.**

So, first convert the list of int into list of str and then use join command to get “125”. Do not add 1 to “125”, because it is a string. So, typecast it into int and then add one. Now, we have an int 126, we need to convert it back into list?

We can use % and //. Ex:

126 % 10 will give us the last digit ‘6’ and 126 // 10 will give us the answer 12.

12 % 10 will give us the last digit ‘2’ and 12 // 10 will give us the answer 1

We can add the last digit to an empty array use reverse to get [1, 2, 6]

Step 1	**We will be converting it into list of str.**	digit_string_list = [str(x) for x in digits]
Step 2	**We will join them and get a string**	digit_string = “”.join (digit_string_list)
Step 3	**Typecast to int and add 1 to it.**	ans = int(digit_string) + 1
Step 4	**Covert int – str.**	

Step 4.1	**We can add another function to do this at the beginning of the program.**	def convert_integer_to_list (self, no):
		  res = list ()
		  while no > 0:
		  digit = no % 10
		  no //= 10
		  res.insert(0, digit)
		  return res

Step 4.2	**Once we add this at the top, then we can give the following command**	res = self.convert_integer_to_list(ans) return res

NOTE: int does not contain leading zero ex: [0,0,1] or [0,20,0,1,0] is not acceptable. It should be completely '0' and only then we can add one at the end or last index. We will return all digits except the last digit i.e.,

```
If int(digit_string) == 0:
    Return digits [0:-1] + [1]
```

Suppose, A = [1, 2, 3, 4]

If A[:-1] + [1] ➔ A = [1, 2, 3, 1]

2nd Approach:

```
def convert_int_to_list_2 (self, num):
    res = []
    for i in str (num):
        res.append(int(i))
    return res
```

Understanding:

the string we have is "125" so,

```
res = []                → we create an empty list "res" and add to it
for i in "125":         → iteration of the str – "125"
    res.append(int(i))  → add each I i.e., I = 1,2,5 to the empty list
return res
```

3rd Approach:

Simple addition.

```
  9 9 9
+   1
1 0 0 0
```

A = [1,2,3,4,5], we travel from right to left.

For 2 1 9 9 9, index is at 4th position because,

0 1 2 3 4

2 1 9 9 9

Initially, carry will be '0' and the index val will be '4' (in, 2 1 9 9 9).

Can we say (digit + 1) // 10 = carry and
(digit + 1) % 10 = val.

So, if digit = 9, then (9 + 1) // 10 → 10 // 10 = 1 and
(9 + 1) % 10 → 10 % 10 = 0

For a number 15, carry will be 1 and val will be 5.

Understanding:

idx = 4
carry = 0
carry = (9 + 1) // 10 = 1
val = (9+1) % 10 = 0
digit[idx] = val

To propogate the carry among the list, we need to decrement index. The code for the program can be written as:

```
idx = 4
carry = 0
while idx >= 0:
    carry(digit[idx] + 1) // 10
    val(digit[idx] + 1) % 10
    digit[idx] = val
```

After idx = 4 value is done, we will decrement the value of idx -= 1 and the value of idx will be 3. So using a while loop, we can write it down as,

```
    carry (digit[idx] + carry) // 10
    val (digit[idx] + carry) % 10
    digit[idx] = val
```

NOTE: check for edge case (9 9 9)

Time Complexity:

For the above code,

digit_string_list = [str(x) for x in digits]	O (n)
digit_string = “”.join(digit_string_list)	O (n)
ans = int(digit_string) + 1	O (1)

def convert_integer_to_list (self, no):
  res = list ()
  while no > 0:
  digit = no % 10
  no //= 10
  res.insert(0, digit)
  return res
	O (log n)

Time Complexity =	O (n), because it is the highest of all.

$$\mathbf{O(n^5) > O(n^4) > O(n^3) > O(n^2) > O(n) > O(\log n) > O(1)}$$

Space Complexity:

Are we using extra array other than “*digits*”?

Yes, we are creating a new list (below line), so the space complexity is O (n).

digit_string_list = [str(x) for x in digits]	O (n)

For the above line, we are creating string so it will have only one space

digit_string = “”.join(digit_string_list)	O (n)

For the next line, we are only adding so there won’t be space

Then the next line, we will saving the value in *ans* so,

ans = int(digit_string) + 1	O (1)

We are converting in the next step using the self.covert_int_to_list_2

def convert_integer_to_list (self, no):
    res = list ()
    while no > 0:
        digit = no % 10
        no //= 10
        res.insert(0, digit)
    return res
	O (log n)

So, the space complexity will for O (n) + O (log n) + O (1) + (1) will be **O (n)**