

# LEETCODE BOOTCAMP

## ASSIGNMENT 4

```
class MyQueue:

    def __init__(self):
        self.input = []
        self.output = []

    def push(self, x: int) -> None:
        self.input.append(x)

    def pop(self) -> int:
        if len(self.output) > 0:
            return self.output.pop()
        while(len(self.input) > 0):
            self.output.append(self.input.pop())
        return self.output.pop()

    def peek(self) -> int:
        if len(self.output) > 0:
            return self.output[-1]
        while(len(self.input) > 0):
            self.output.append(self.input.pop())
        return self.output[-1]

    def empty(self) -> bool:
        return len(self.input) == 0 and len(self.output) == 0
```

Problem List

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted 22 / 22 testcases passed

XoITACHiXo submitted at Mar 26, 2025 16:11


Runtime

0 ms Beats 100.00%

Analyze Complexity

Memory

18.04 MB Beats 20.84%



Code

Python3

```
class MyQueue:
    def __init__(self):
        self.input = []
        self.output = []

    def push(self, x: int) -> None:
        self.input.append(x)

    def pop(self) -> int:
        if len(self.output) > 0:
            return self.output.pop()
        while len(self.input) > 0:
            self.output.append(self.input.pop())
        return self.output.pop()
```

Saved

Testcase

Test Result

Accepted Runtime: 0 ms

Case 1

Input

["MyQueue", "push", "push", "peek", "pop", "empty"]

[[], [1], [2], [], [], []]

Output

[null, null, null, 1, 1, false]

Expected

[null, null, null, 1, 1, false]

```
class Solution:
    def decodeString(self, s: str) -> str:
        stack = []; curNum = 0; curString = ''
        for c in s:
            if c == '[':
                stack.append(curString)
                stack.append(curNum)
                curString = ''
                curNum = 0
            elif c == ']':
                num = stack.pop()
                prevString = stack.pop()
                curString = prevString + num*curString
            elif c.isdigit():
                curNum = curNum*10 + int(c)
            else:
                curString += c
        return curString
```

Description
Editorial
Solutions
Accepted
Submissions

All Submissions

Accepted
XoITACHIxo submitted at Mar 26, 2023 16:12

Editorial
Solution

Runtime
38 ms | Beat: 0.10%
Analyze Complexity

Memory
17.83 MB | Beat: 35.06%

2.32% of solutions used 3 ms of runtime

Python3

```

class Solution:
    def decodeString(self, s: str) -> str:
        stack = []
        curNum = 0
        curString = ''
        for c in s:
            if c == '[':
                stack.append(curString)
                stack.append(curNum)
                curString = ''
            elif c == ']':
                k = stack.pop()
                current_string = stack.pop()
                # Just finished parsing this k, save current string and k for when we pop
                stack.append((current_string, k))
                # Reset current_string and k for this new frame
                current_string = ''
                k = 0
            elif c == '0':
                k = 0
            elif c == '1':
                k = 1
            elif c == '2':
                k = 2
            elif c == '3':
                k = 3
            elif c == '4':
                k = 4
            elif c == '5':
                k = 5
            elif c == '6':
                k = 6
            elif c == '7':
                k = 7
            elif c == '8':
                k = 8
            elif c == '9':
                k = 9
            else:
                curString += c
        return current_string

```

Testcase
Test Result

Accepted
Runtime: 0 ms

Case 1
Case 2
Case 3

Input
s = "3[a]2[bc]"

Output
"aaabcbc"

Expected
"aaabcbc"

Contribute a testcase

```

class Solution:
    def peopleAwareOfSecret(self, n: int, delay: int, forget: int) -> int:
        dp = [1] + [0] * (n - 1)
        mod = 10 ** 9 + 7
        share = 0
        for i in range(1, n):
            dp[i] = share = (share + dp[i - delay] - dp[i - forget]) % mod
        return sum(dp[-forget:]) % mod

```

All Submissions

Accepted 82 / 82 testcases passed

XoTACHiXo submitted at Mar 26, 2025 16:14

Solution

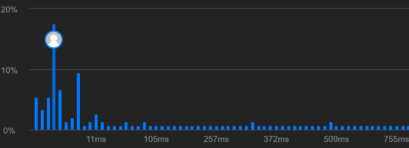
Runtime

3 ms Beats 85.81%

Analyze Complexity

Memory

17.81 MB Beats 77.70%



11ms 105ms 257ms 372ms 509ms 755ms

Code | Python3

```
class Solution:
    def peopleAwareOfSecret(self, n: int, delay: int, forget: int) -> int:
        dp = [1] + [0] * (n - 1)
        mod = 10 ** 9 + 7
        share = 0
        for i in range(1, n):
            dp[i] = share + (share + dp[i - delay] - dp[i - forget]) % mod
        return sum(dp[-forget:]) % mod
```

Python3 Auto

```
1 class Solution:
2     def peopleAwareOfSecret(self, n: int, delay: int, forget: int) -> int:
3         dp = [1] + [0] * (n - 1)
4         mod = 10 ** 9 + 7
5         share = 0
6         for i in range(1, n):
7             dp[i] = share + (share + dp[i - delay] - dp[i - forget]) % mod
8         return sum(dp[-forget:]) % mod
```

Saved Ln 8, Col 41

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

n = 6

delay = 2

forget = 4

Output

5