

```
File Edit Selection View Go Run Terminal Help
L4list.ipynb
C:\Users\Darshil Shah > OneDrive - Northeastern University > Desktop > PSA > HW3 > L4list.ipynb
+ Code + Markdown + Run All + Restart + Clear All Outputs + Variables + Outline ... Python 3.9.5
[8] ✓ 8.3s
...
3.9.5 (tags/v3.9.5:0a7dcbf, May 3 2021, 17:27:52) [MSC v.1928 64 bit (AMD64)]
Testing stack using python_list_of_unknown_size
Testing on 5 elements started using python_list_of_unknown_size
4 3 2 1
Testing on 5 elements Passed
Testing on 5 elements started using python_list_of_unknown_size
4 3 2 1 0
Testing on 5 elements Passed
Testing on 100 elements started using python_list_of_unknown_size
99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57
Testing on 100 elements Passed
Testing on 500000 elements started using python_list_of_unknown_size
Testing on 500000 elements Passed
Testing on 500001 elements started using python_list_of_unknown_size
Testing on 500001 elements Passed
All the above function must have O(1) time and O(1) space for A grade
s =
after adding 6 = 6
after removing front =
after adding 5 = 5
after removing front =
after removing front =
s =
after adding 0 : 0
after adding 1 : 0 1
...
622. Design Circular Queue: https://leetcode.com/problems/design-circular-queue/
225. Implement Stack using Queues: https://leetcode.com/problems/implement-stack-using-queues/
232. Implement Queue using Stacks: https://leetcode.com/problems/implement-queue-using-stacks/
155. Min Stack: https://leetcode.com/problems/min-stack/
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Q1 Circular queue using Python list

leetcode.com/problems/design-circular-queue/submissions/1166049881/

Problem List > > > Run Submit < 00:00:00 Premium

Description Editorial Solutions Submissions

All Submissions

Accepted

shahdarshil622 submitted at Feb 04, 2024 12:38

Runtime: 59 ms Beats 11.39% of users with Python3 Memory: 17.20 MB Beats 75.68% of users with Python3

Code: Python3

```
class Queue():
    def __init__(self, k: 'int'):
        # You must use Python List
        # Must use all spaces
        self.a = [None]*k #CANNOT CHANGE THIS. k space is already allocated
        self._MAX = k
        ## YOU CAN HAVE YOUR PRIVATE DATA MEMBER HERE
        self._front = -1
        self._back = -1

    def enqueue(self, T)->'bool':
        ## YOU CANNOT CALL append in this routine as U already have enough space
        ## YOU CAN DO: self.a[pos] = T #pos is some position between 0 to self._MAX-1
        #print("WRITE CODE")
```

Testcase: Test Result

Case 1 +

```
["MyCircularQueue","enqueue","enqueue","enqueue","enqueue","Rear","isFull","deQueue","enqueue","Rear"]
```

[[3],[1],[2],[3],[4],[1],[1],[1],[4],[1]]

Source

Q2 # Stack Using Circular Queue

This screenshot shows a LeetCode submission for the problem "Implement Stack using Queues". The submission is accepted and was made by user 'shahdarshil622' on February 04, 2024, at 13:44. The performance metrics are: Runtime 39 ms (Beats 37.15% of users with Python3) and Memory 16.94 MB (Beats 49.42% of users with Python3). A bar chart shows the runtime distribution. The code is written in Python3 and implements a Queue class with methods enqueue and dequeue. The test case shows a sequence of operations: ["MyStack", "push", "push", "top", "pop", "empty"], which results in the output: [[], [1], [2], [], [], []].

```
class Queue():
    def __init__(self, k: 'int'):
        # You must use Python List
        # Must use all spaces
        self._a = [None] * k # CANNOT CHANGE THIS. k space is already allocated
        self._MAX = k
        # YOU CAN HAVE YOUR PRIVATE DATA MEMBER HERE
        self._front = -1
        self._back = -1

    def enqueue(self, T) -> 'bool':
        # YOU CANNOT CALL append in this routine as U already have enough space
        # YOU CAN DO: self._a[pos] = T # pos is some position between 0 to self._MAX-1
        # print("WRITE CODE")
```

Q3 Implement Queue using Stacks

This screenshot shows a LeetCode submission for the problem "Implement Queue using Stacks". The submission is accepted and was made by user 'shahdarshil622' on February 04, 2024, at 08:47. The performance metrics are: Runtime 32 ms (Beats 76.36% of users with Python3) and Memory 16.70 MB (Beats 30.70% of users with Python3). A bar chart shows the runtime distribution. The code is written in Python3 and implements a Stack class with methods push, len, and peek. The test case shows a sequence of operations: ["MyQueue", "push", "push", "peek", "pop", "empty"], which results in the output: [[], [1], [2], [], [], []].

```
class Stack():
    def __init__(self):
        # MUST USE ONLY PYTHON LIST
        self._a = []
        # print("WRITE CODE")
        self._maxspace = 0
        self._currentspace = 0

    def __len__(self):
        return len(self._a)

    def push(self, T) -> 'None':
        self._currentspace += 1
        if (self._maxspace < self._currentspace):
            self._maxspace = self._currentspace
```

Q4 Min Stack

The screenshot displays a LeetCode submission for the 'Min Stack' problem. The submission is accepted, showing a runtime of 50ms and memory usage of 20.64 MB. The code is a Python class Stack with methods push, pop, top, and getMin. The test case shows a sequence of operations: push, push, push, getMin, pop, top, getMin, resulting in a final state of [1, [-2], [0], [-3], [], [], [], []].

Accepted shahdarshi622 submitted at Feb 03, 2024 23:56

Runtime: 50 ms (Beats 98.16% of users with Python3)

Memory: 20.64 MB (Beats 93.66% of users with Python3)

Code: Python3

```
class Stack():
    def __init__(self):
        #MUST USE ONLY PYTHON LIST
        self.a = []
        #print("WRITE CODE")
        self._maxspace = 0
        self._currentspace = 0

    def __len__(self):
        return(len(self.a))

    def push(self, t) -> "None":
        self._currentspace += 1
        if(self._maxspace < self._currentspace):
            self._maxspace = self._currentspace
```

Testcase: Case 1

["MinStack", "push", "push", "push", "getMin", "pop", "top", "getMin"]

[[], [-2], [0], [-3], [], [], [], []]

