Challenge 2: Implement Two Stacks Using One List

Can you implement two stacks using a single array? A solution is placed in the "solution" section for your help, but we would suggest you to solve it on your own first.

We'll cover the following

Problem Statement#

Implement the following functions to implement two stacks using a single array such that for storing elements both stacks should use the same array. An illustration is also provided for your understanding. Also, for this problem, initialize a Python list with the provided fixed **size** and perform all the operations **in-place** without growing or shrinking the list!

Function Prototypes#

```
def push1(value): # pushes value in stack 1
def push2(value): # pushes value in stack 2
def pop1(): # pops an element from stack 1
def pop2():# pops an element from stack 2
```

Input/Output#

`push1(value)`

Input: an integer

Output: inserts the given value in the first stack, i.e., *stack1*

`push2(value)`

Input: an integer

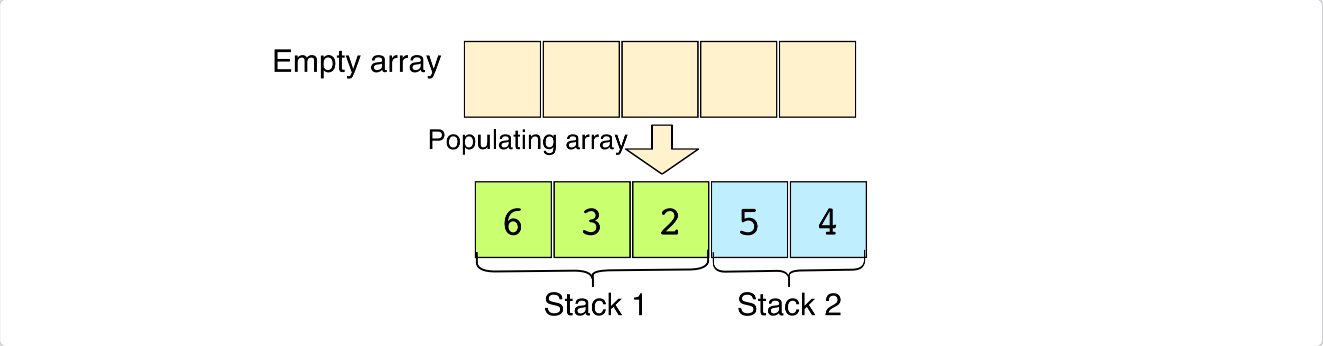
Output: inserts the given value in the second stack i.e *stack2*

`pop1()`

Output: returns and removes the top value of *stack1*

`pop2()`

Output: returns and removes the top value of *stack2*



Coding Exercise

Take a close look and design a step-by-step algorithm first before jumping on to the implementation. This problem is designed for your practice, so try to solve it on your own first. If you get stuck, you can always refer to the solution provided in the solution section. Good Luck!

```
class TwoStacks:
    # Initialize the two stacks here
    def __init__(self, size):
        self.master_stack = [None]* size
        self.size = size
        self.first_end_idx = -1
        self.second_end_idx = self.size

    # Insert Value in First Stack
    def push1(self, value):
        prev = value
        i = 0
        while i < self.second_end_idx:
            curr = self.master_stack[i]
            self.master_stack[i] = prev
            prev = curr
            i += 1
        if i > self.first_end_idx:
            self.first_end_idx += 1


    # Insert Value in Second Stack
    def push2(self, value):
        prev = value
        i = self.size - 1
        while i > self.first_end_idx:
            curr = self.master_stack[i]
            self.master_stack[i] = prev
            prev = curr
            i -= 1
        if i < self.second_end_idx:
            self.second_end_idx -= 1


    # Return and remove top Value from First Stack
    def pop1(self):
        if self.first_end_idx < 0:
            return None
        i = self.first_end_idx
        prev = None
        while i >=0:
            curr = self.master_stack[i]
            self.master_stack[i] = prev
            prev = curr
            i -= 1
        self.first_end_idx -= 1
        return prev
    def pp(self):
        print(self.master_stack, self.first_end_idx, self.second_end_idx)


    # Return and remove top Value from Second Stack
    def pop2(self):
        if self.second_end_idx >= self.size:
            return None
        i = self.second_end_idx
        prev = None
        while i < self.size:
            curr = self.master_stack[i]
            self.master_stack[i] = prev
            prev = curr
            i += 1
        self.second_end_idx += 1
        return prev
```

← Back

Solution Review: Generate Binary Nu...

Next →

Solution Review: Implementing Two S...

☑ Completed

Report an Issue