Implement two stacks in an array

This method efficiently utilizes the available space. It doesn't cause an overflow if there is space available in arr[]. The idea is to start two stacks from two extreme corners of arr[]. stack1 starts from the leftmost element, the first element in stack1 is pushed at index 0. The stack2 starts from the rightmost corner, the first element in stack2 is pushed at index (n-1). Both stacks grow (or shrink) in opposite direction. To check for overflow, all we need to check is for space between top elements of both stacks. This check is highlighted in the below code.

```
# Python Script to Implement two stacks in a list
class twoStacks:

    def __init__(self, n):      # constructor
        self.size = n
        self.arr = [None] * n
        self.top1 = -1
        self.top2 = self.size

    # Method to push an element x to stack1
    def push1(self, x):

        # There is at least one empty space for new element
        if self.top1 < self.top2 - 1 :
            self.top1 = self.top1 + 1
            self.arr[self.top1] = x

        else:
            print("Stack Overflow ")
            exit(1)

    # Method to push an element x to stack2
    def push2(self, x):

        # There is at least one empty space for new element
        if self.top1 < self.top2 - 1:
            self.top2 = self.top2 - 1
            self.arr[self.top2] = x

        else :
           print("Stack Overflow ")
           exit(1)
```

```
    # Method to pop an element from first stack
    def pop1(self):
        if self.top1 >= 0:
            x = self.arr[self.top1]
            self.top1 = self.top1 -1
            return x
        else:
            print("Stack Underflow ")
            exit(1)

    # Method to pop an element from second stack
    def pop2(self):
        if self.top2 < self.size:
            x = self.arr[self.top2]
            self.top2 = self.top2 + 1
            return x
        else:
            print("Stack Underflow ")
            exit()

# Driver program to test twoStacks class
ts = twoStacks(5)
ts.push1(5)
ts.push2(10)
ts.push2(15)
ts.push1(11)
ts.push2(7)

print("Popped element from stack1 is " + str(ts.pop1()))
ts.push2(40)
print("Popped element from stack2 is " + str(ts.pop2()))
```