class DequeUsingTwoStacks:  
 def \_\_init\_\_(self):  
 self.stack1 = [] # For operations on the "right" end  
 self.stack2 = [] # For operations on the "left" end  
  
 def push\_to\_left(self, value):  
 while len(self.stack1) > 0:  
 self.stack2.append(self.stack1.pop()) # Transfer all elements to stack2  
 self.stack2.append(value) # Add the new value to the left  
 while len(self.stack2) > 0:  
 self.stack1.append(self.stack2.pop()) # Transfer all elements back to stack1  
  
 def push\_to\_right(self, value):  
 self.stack1.append(value) # Push directly to stack1  
  
 def pop\_from\_left(self):  
 if len(self.stack1) == 0:  
 raise IndexError("Pop from empty deque")  
 while len(self.stack1) > 0:  
 self.stack2.append(self.stack1.pop()) # Transfer all elements to stack2  
 value = self.stack2.pop() # Pop the leftmost element  
 while len(self.stack2) > 0:  
 self.stack1.append(self.stack2.pop()) # Transfer all elements back to stack1  
 return value  
  
 def pop\_from\_right(self):  
 if len(self.stack1) == 0:  
 raise IndexError("Pop from empty deque")  
 return self.stack1.pop() # Pop directly from stack1  
  
  
class DequeUsingStackQueue:  
 def \_\_init\_\_(self):  
 self.stack = [] # Stack for right-side operations  
 self.queue = [] # List as a queue for left-side operations  
  
 def push\_to\_left(self, value):  
 self.queue.insert(0, value) # Insert at the front of the queue  
  
 def push\_to\_right(self, value):  
 self.stack.append(value) # Push to the stack  
  
 def pop\_from\_left(self):  
 if len(self.queue) > 0:  
 return self.queue.pop(0) # Remove from the front of the queue  
 elif len(self.stack) > 0:  
 while len(self.stack) > 0:  
 self.queue.insert(0, self.stack.pop()) # Move all elements to the queue  
 return self.queue.pop(0)  
 else:  
 raise IndexError("Pop from empty deque")  
  
 def pop\_from\_right(self):  
 if len(self.stack) > 0:  
 return self.stack.pop() # Remove from the stack  
 elif len(self.queue) > 0:  
 while len(self.queue) > 0:  
 self.stack.append(self.queue.pop(0)) # Move all elements to the stack  
 return self.stack.pop()  
 else:  
 raise IndexError("Pop from empty deque")  
  
  
# Example usage of DequeUsingTwoStacks (Part A)  
print("Part A: Deque Using Two Stacks\n")  
deque1 = DequeUsingTwoStacks()  
deque1.push\_to\_left(10)  
deque1.push\_to\_right(20)  
deque1.push\_to\_left(5)  
print("Numbers in the deque for Part A:")  
print("Pop from right: ", deque1.pop\_from\_right()) # Output: 20  
print("Pop from left: ", deque1.pop\_from\_left()) # Output: 5  
print("Pop from left: ", deque1.pop\_from\_left()) # Output: 10  
  
# Example usage of DequeUsingStackQueue (Part B)  
print("\nPart B: Deque Using Stack and Queue\n")  
deque2 = DequeUsingStackQueue()  
deque2.push\_to\_left(30)  
deque2.push\_to\_right(40)  
deque2.push\_to\_left(20)  
print("Numbers in the deque for Part B:")  
print("Pop from left: ", deque2.pop\_from\_left()) # Output: 20  
print("Pop from right: ", deque2.pop\_from\_right()) # Output: 40  
print("Pop from left: ", deque2.pop\_from\_left()) # Output: 30

