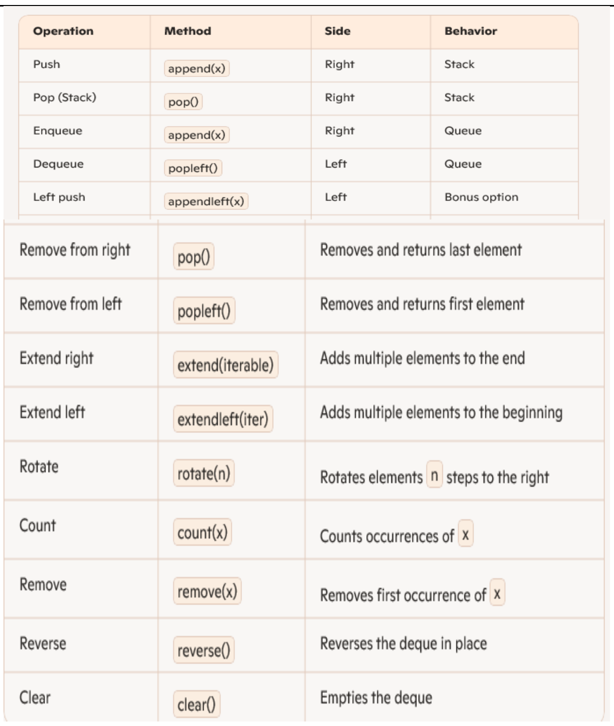
Stack and Queue Operations in Python

```
stack = []

# Push elements
stack.append('A')
stack.append('B')
stack.append('C')
print("Stack after pushes:", stack)  # ['A', 'B', 'C']

# Pop element
top = stack.pop()
print("Popped:", top)               # 'C'
print("Stack after pop:", stack)     # ['A', 'B']

# Peek at top element
print("Top element:", stack[-1])     # 'B'

# Check if stack is empty
print("Is stack empty?", not stack)
# False
#how to append element at the front?
```

```
queue = []

# Enqueue elements
queue.append('X')
queue.append('Y')
queue.append('Z')
print("Queue after enqueues:", queue)  # ['X', 'Y', 'Z']

# Dequeue element
first = queue.pop(0)
print("Dequeued:", first)            # 'X'
print("Queue after dequeue:", queue)   # ['Y', 'Z']

# Peek at front element
print("Front element:", queue[0])      # 'Y'

# Check if queue is empty
print("Is queue empty?", not queue)    # False
```

```
from collections import deque
stack = deque()
# Push elements (LIFO)
stack.append('A')
stack.append('B')
stack.append('C')
print("Stack:", stack)  # deque(['A', 'B', 'C'])
# Pop elements
print("Popped:", stack.pop())  # 'C'
print("Stack after pop:", stack)  # deque(['A',
'B'])
from collections import deque
# Initialize a deque
dq = deque()
# Append to the right (default behavior)
dq.append(10)
dq.append(20)
# Append to the left
dq.appendleft(5)
# Current deque state
print("Deque after appends:", dq)  # Output:
deque([5, 10, 20])
# Pop from the right
right_item = dq.pop()
print("Popped from right:", right_item)  #
Output: 20
# Pop from the left
left_item = dq.popleft()
print("Popped from left:", left_item)  # Output:
5

# Final deque state
print("Deque after pops:", dq)  # Output:
deque([10])
```



Operation	Method	Side	Behavior
Push	append(x)	Right	Stack
Pop (Stack)	pop()	Right	Stack
Enqueue	append(x)	Right	Queue
Dequeue	popleft()	Left	Queue
Left push	appendleft(x)	Left	Bonus option

Remove from right	pop()	Removes and returns last element
Remove from left	popleft()	Removes and returns first element
Extend right	extend(iterable)	Adds multiple elements to the end
Extend left	extendleft(iter)	Adds multiple elements to the beginning
Rotate	rotate(n)	Rotates elements n steps to the right
Count	count(x)	Counts occurrences of x
Remove	remove(x)	Removes first occurrence of x
Reverse	reverse()	Reverses the deque in place
Clear	clear()	Empties the deque