Newton School of Technology

20

Stacks

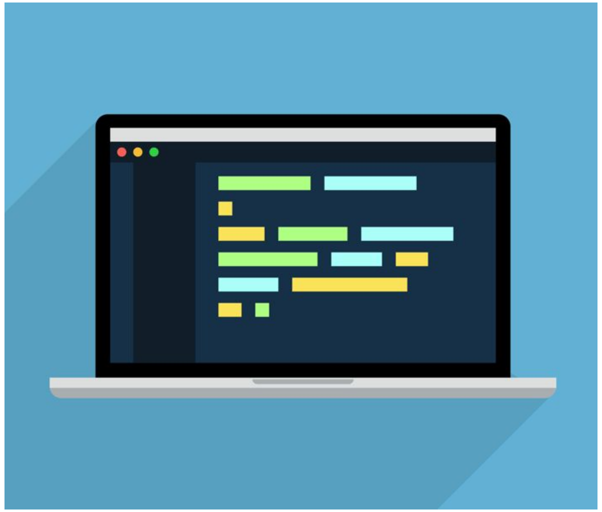
by Gladden Rumao

CSA 221 : DSA

Newton School of Technology

Quick Recap :

- Linked List
  - Singly
  - Doubly
- Insert and delete
  - At beginning
  - At end

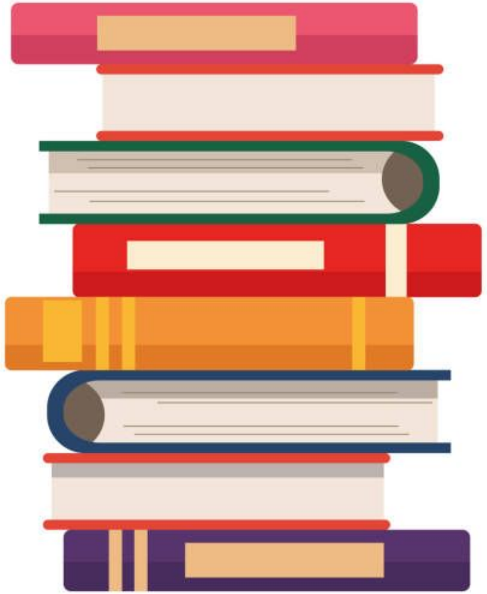


Newton School of Technology

Pre-read Quiz Time!

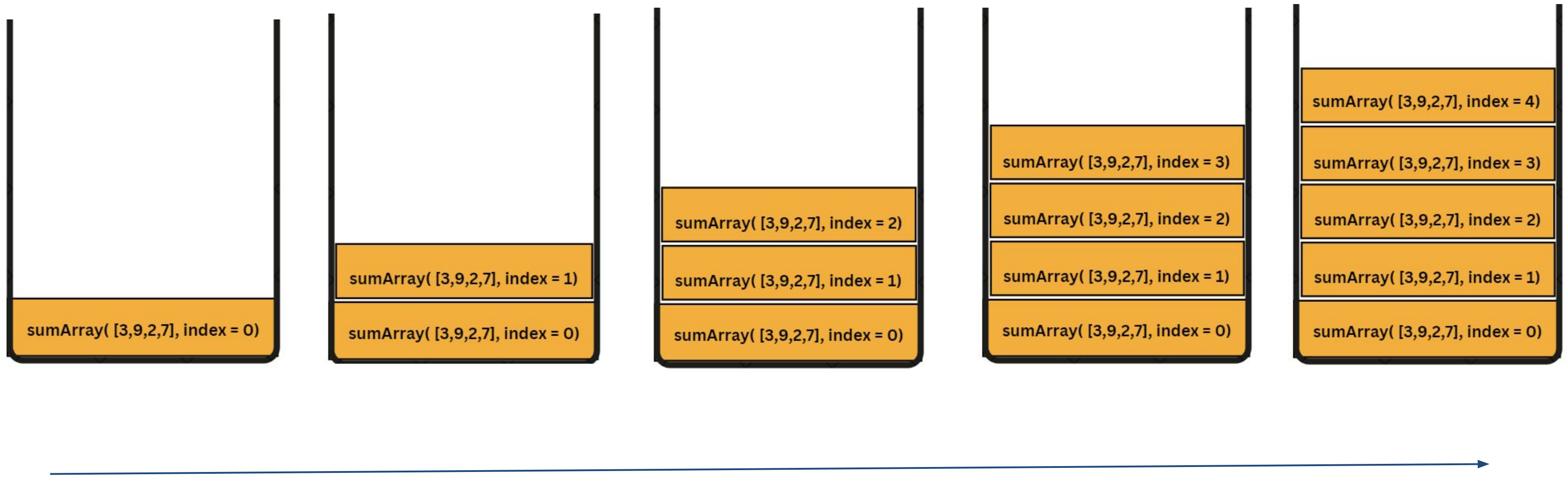
Newton School of Technology

Introduction to Stack



Can you tell ?

One use-case of it you have already seen and used


sumArray([3,9,2,7], index = 0)
sumArray([3,9,2,7], index = 1)
sumArray([3,9,2,7], index = 0)
sumArray([3,9,2,7], index = 2)
sumArray([3,9,2,7], index = 1)
sumArray([3,9,2,7], index = 0)
sumArray([3,9,2,7], index = 3)
sumArray([3,9,2,7], index = 2)
sumArray([3,9,2,7], index = 1)
sumArray([3,9,2,7], index = 0)
sumArray([3,9,2,7], index = 4)
sumArray([3,9,2,7], index = 3)
sumArray([3,9,2,7], index = 2)
sumArray([3,9,2,7], index = 1)
sumArray([3,9,2,7], index = 0)

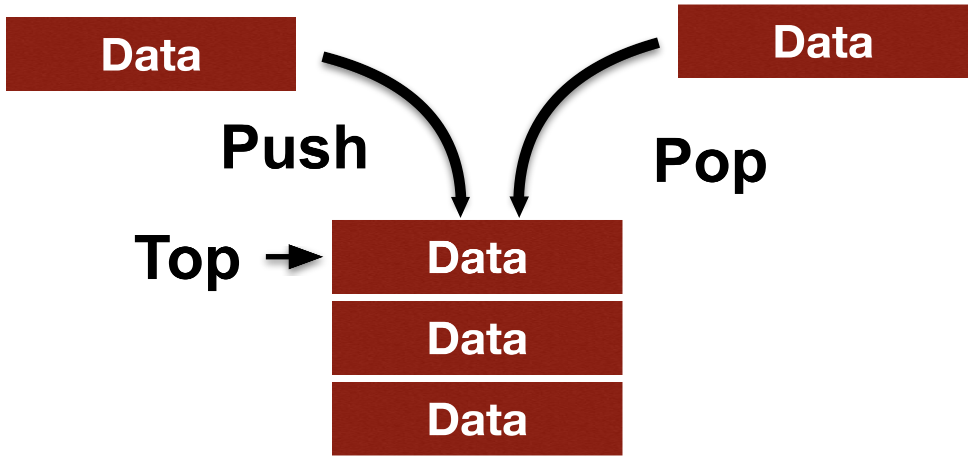
Recursive Stack

Newton School of Technology

What is a stack?

What is a Stack ?

- A stack is a one-ended linear data structure which models a real world stack by having two primary operations, namely push and pop.
- A stack follows the Last In, First Out (LIFO) principle, where the last element pushed is the first to be popped.

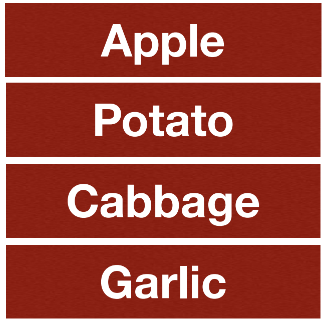


What is a Stack ?

Newton School of Technology

Instructions

pop()
push(‘Onion’)
push(‘Celery’)
push(‘Watermelon’)
pop()
pop()
push(‘Lettuce’)

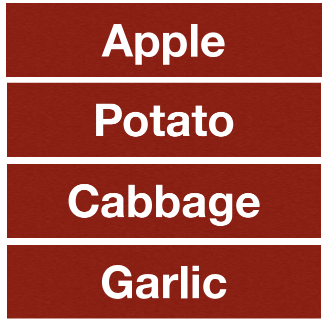


What is a Stack ?

Newton School of Technology

Instructions

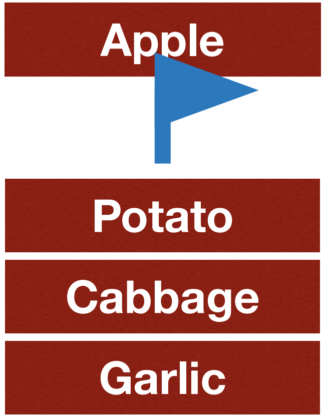
```
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')
```



What is a Stack ?

<ins>**Instructions**</ins>

→ pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

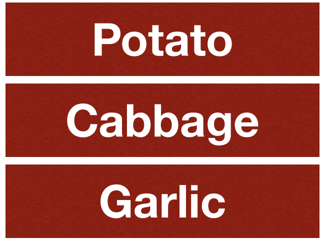


What is a Stack ?

Newton School of Technology

<u>Instructions</u>

```
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')
```

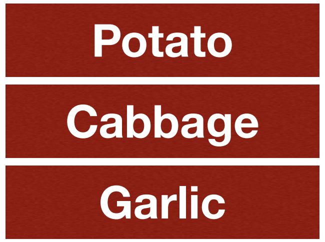


What is a Stack ?

Newton School of Technology

Instructions

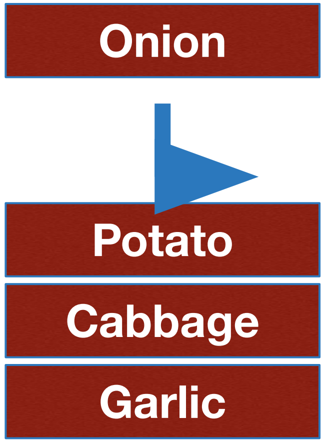
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')



What is a Stack ?

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

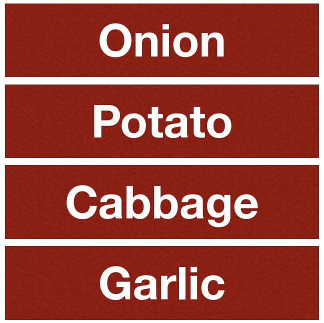


What is a Stack ?

Newton School of Technology

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

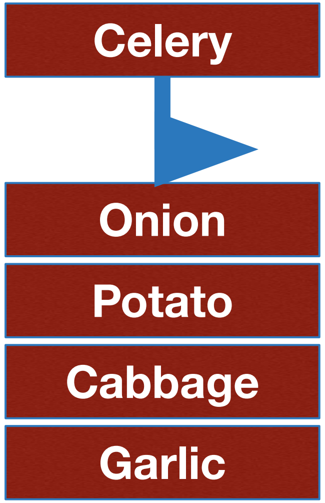


What is a Stack ?

Newton School of Technology

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

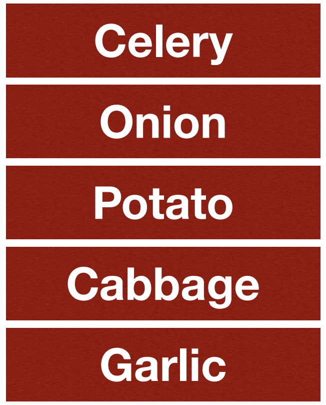


What is a Stack ?

Newton School of Technology

Instructions

```
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')
```

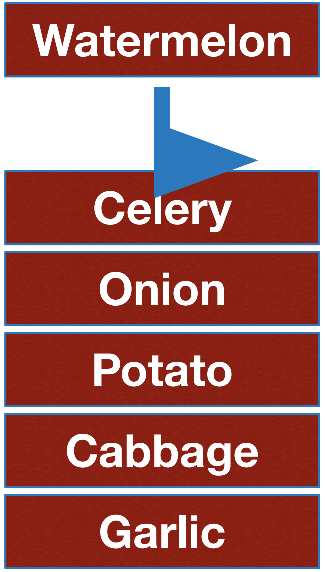


What is a Stack ?

Newton School of Technology

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

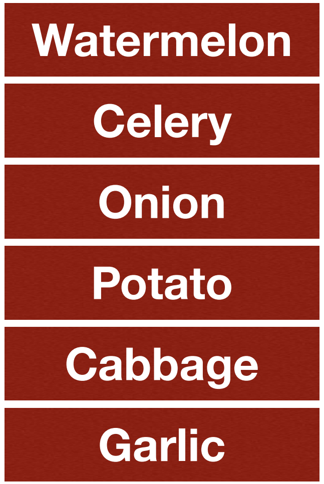


What is a Stack ?

Newton School of Technology

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

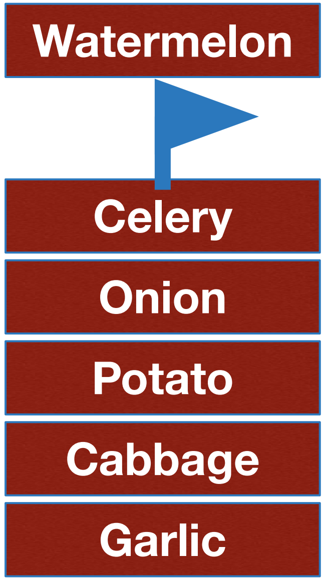


What is a Stack ?

Newton School of Technology

Instructions

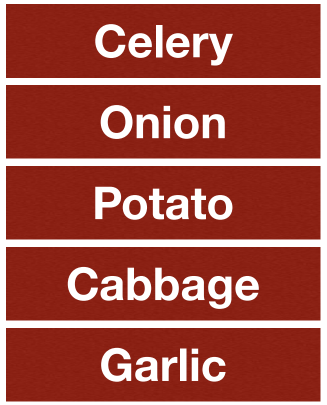
pop()
push('Onion')
push('Celery')
push('Watermelon')
→ pop()
pop()
push('Lettuce')



What is a Stack ?

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

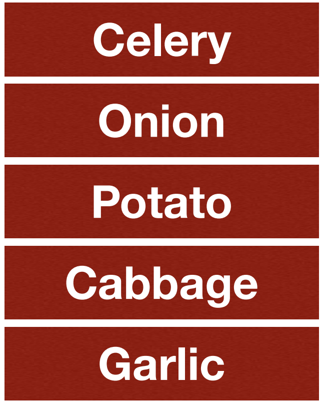


What is a Stack ?

Newton School of Technology

Instructions

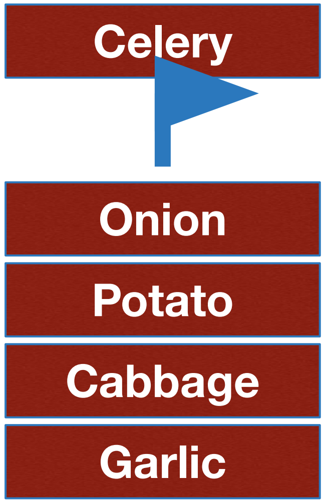
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')



What is a Stack ?

Instructions

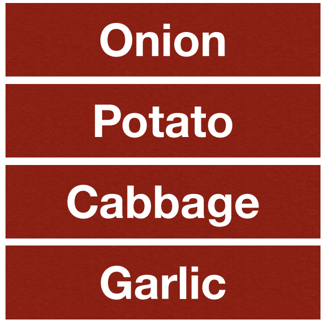
```
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')
```



What is a Stack ?

Instructions

pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')

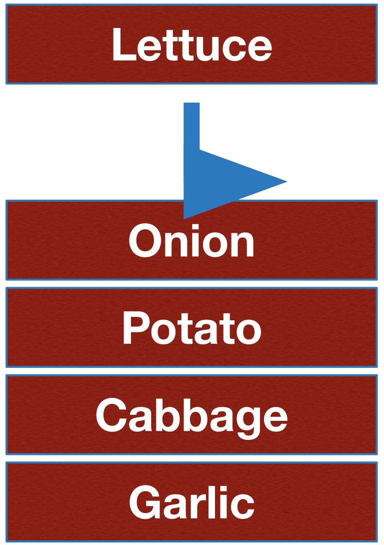


What is a Stack ?

Newton School of Technology

Instructions

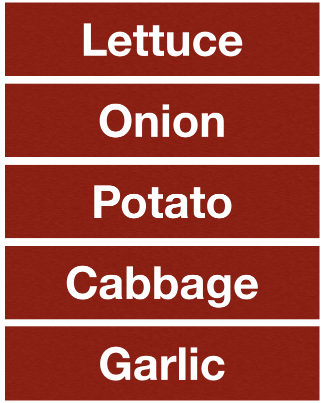
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')



What is a Stack ?

Instructions

```
pop()
push('Onion')
push('Celery')
push('Watermelon')
pop()
pop()
push('Lettuce')
```



When and where stack is used?

- Used by undo mechanisms in text editors.
- Used in compiler syntax checking for matching brackets and braces.
- Can be used to model a pile of books or plates.
- Used behind the scenes to support recursion by keeping track of previous function calls.

How stack?

Newton School of Technology

Stack Using Linked List

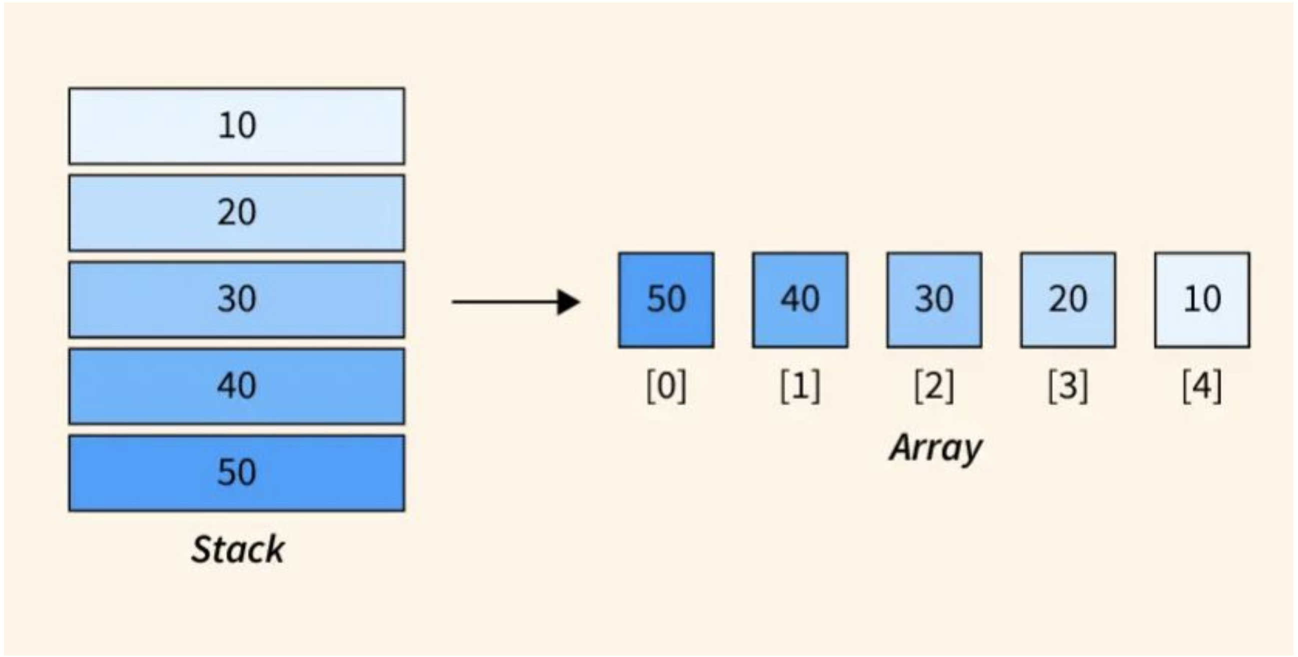
When and where stack is used?

- Used by undo mechanisms in text editors.
- Used in compiler syntax checking for matching brackets and braces.
- Can be used to model a pile of books or plates.
- Used behind the scenes to support recursion by keeping track of previous function calls.

Newton School of Technology

Stack Using Array

Stack Using Array



Stack Using Array

Stack operation	Operation on Array
Push	Append at the end of an array
Pop	Delete the last element of an array
Peek / Top	Return the element at the last index.
Size	Return len of the list.

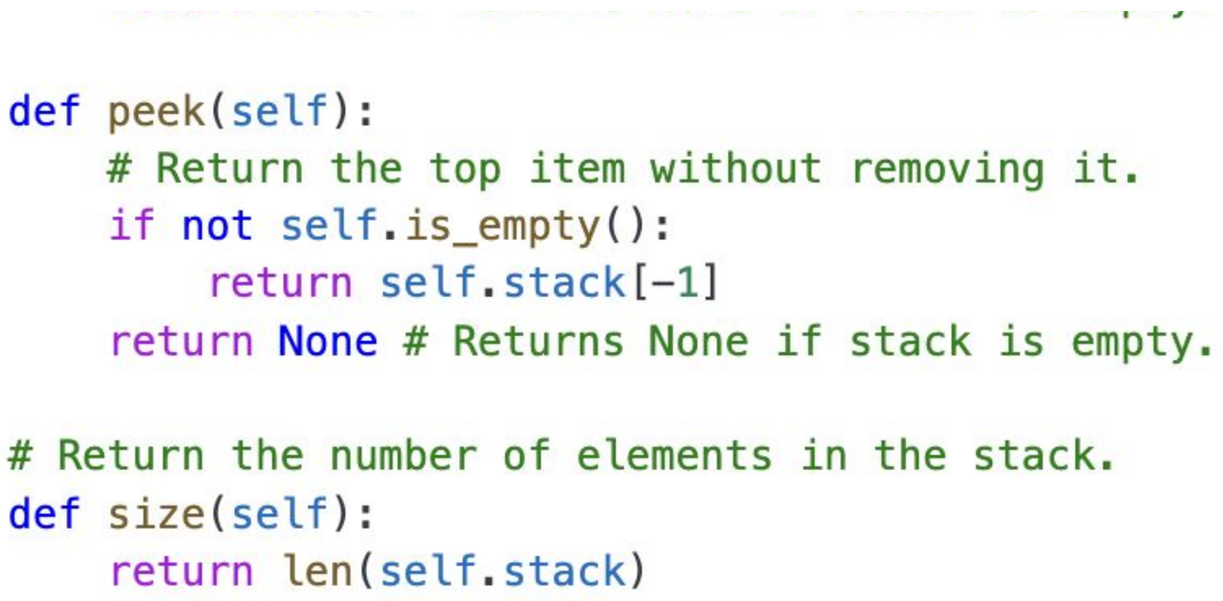
Implementation: push and pop

```python
class Stack:
    def __init__(self):
        self.stack = []

    # Add an item to the top of the stack.
    def push(self, item):
        self.stack.append(item)

    def pop(self):
        # Remove and return the top item of the stack.
        if not self.is_empty():
            return self.stack.pop()
        return None # Returns None if stack is empty.
```

Implementation: peek and size



```python
def peek(self):
    # Return the top item without removing it.
    if not self.is_empty():
        return self.stack[-1]
    return None # Returns None if stack is empty.

# Return the number of elements in the stack.
def size(self):
    return len(self.stack)
```

Newton School of Technology

Stack Using Linked List

Stack Using Linked List

Newton School of Technology

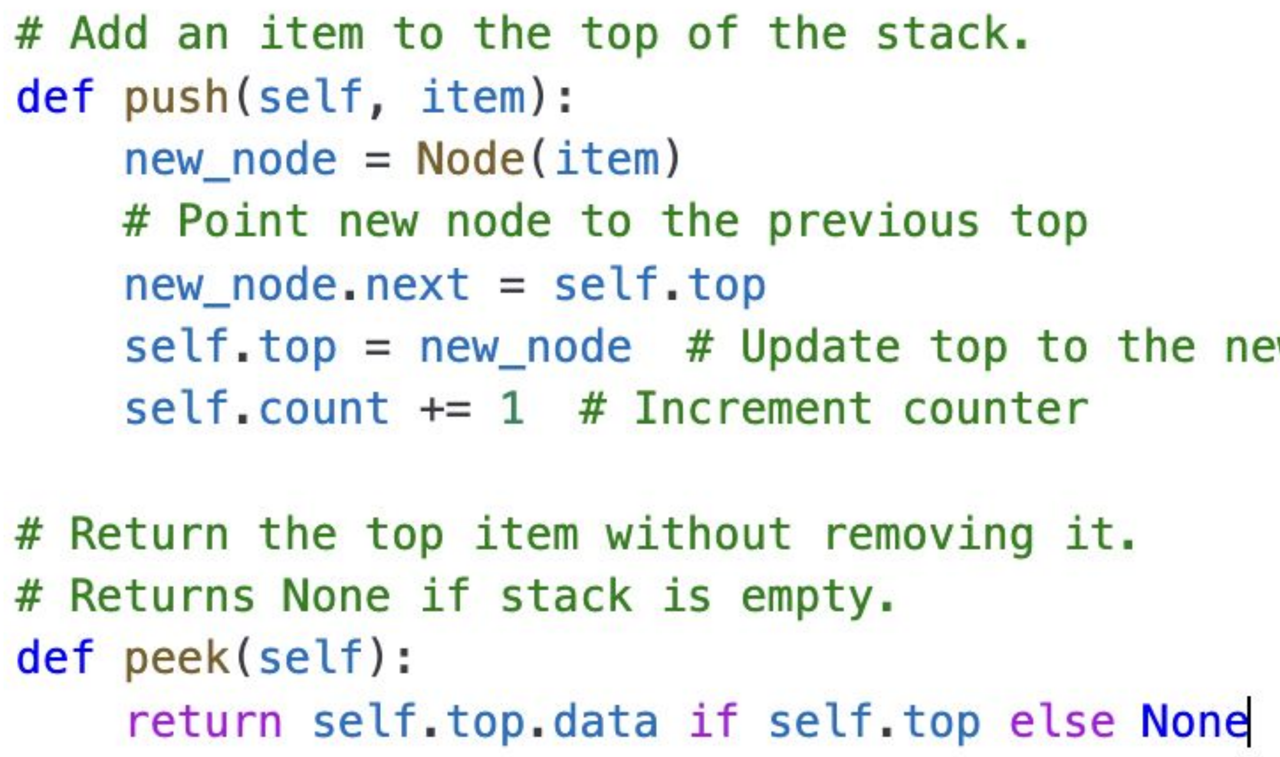
Stack operation	Operation on Linked List
Push	Insert at beginning of the linked list.
Pop	Delete at beginning of the linked list.
Peek / Top	Return the value at the head of the linked list.
Size	Return len of the array

Implementation: Node & stack class

```python
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class Stack:
    def __init__(self):
        self.top = None
        self.count = 0  # Counter to track stack size
```

Implementation: push and peek



```python
# Add an item to the top of the stack.
def push(self, item):
    new_node = Node(item)
    # Point new node to the previous top
    new_node.next = self.top
    self.top = new_node  # Update top to the new node
    self.count += 1  # Increment counter

# Return the top item without removing it.
# Returns None if stack is empty.
def peek(self):
    return self.top.data if self.top else None
```

Implementation: pop and size

```python
# Remove and return the top item of the stack.
# Returns None if stack is empty.
def pop(self):
    if self.is_empty():
        return None
    popped_item = self.top.data
    # Move top to the next node
    self.top = self.top.next
    self.count -= 1  # Decrement counter
    return popped_item

def size(self):
    # Return the number of elements in the stack in O(1).
    return self.count
```

Newton School of Technology

Complexity Analysis

Time Complexity

Newton School of Technology

Pushing	O(1)
Popping	O(1)
Peeking	O(1)
Searching	O(n)
Size	O(1)

Newton School of Technology

Summary Quiz

Newton School of Technology

END