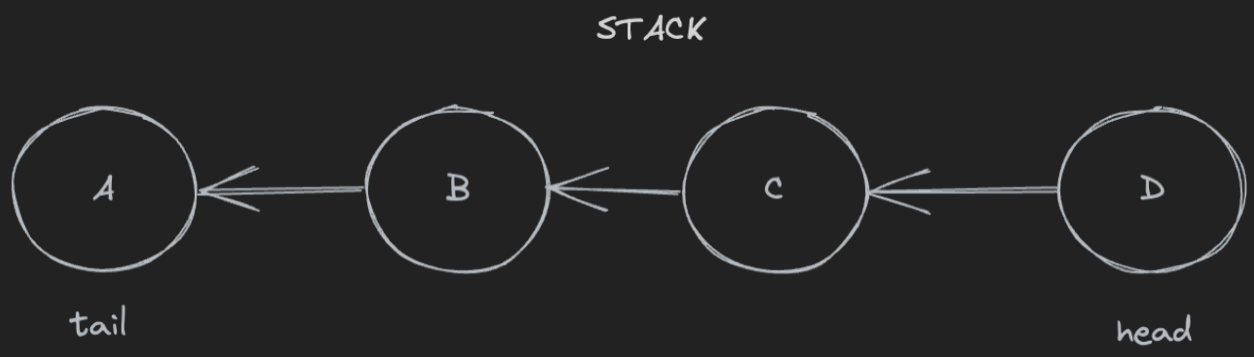
Operation	Stack
Pushing	O(1)
Popping	O(1)
Peeking	O(1)
Searching	O(n)
Size	O(1)

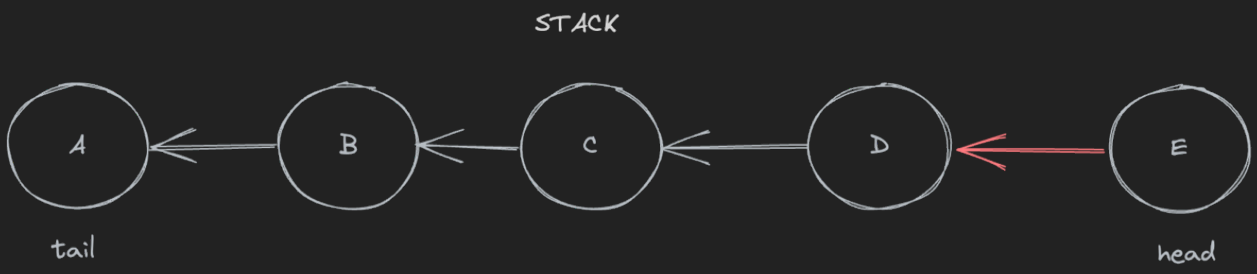
Definition

A one-ended linear data structure, allowing you to primarily push onto the stack, and pop off the stack.
Uses **LIFO** (Last In First Out)



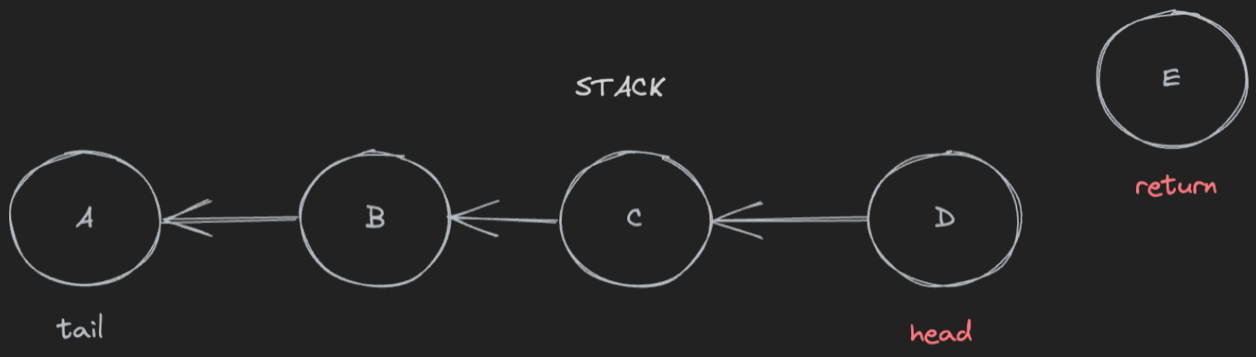
Adding to a Stack (Push)

You can only append items to the top of the stack



Removing from a Stack (Pop)

You can only remove the top item of the stack

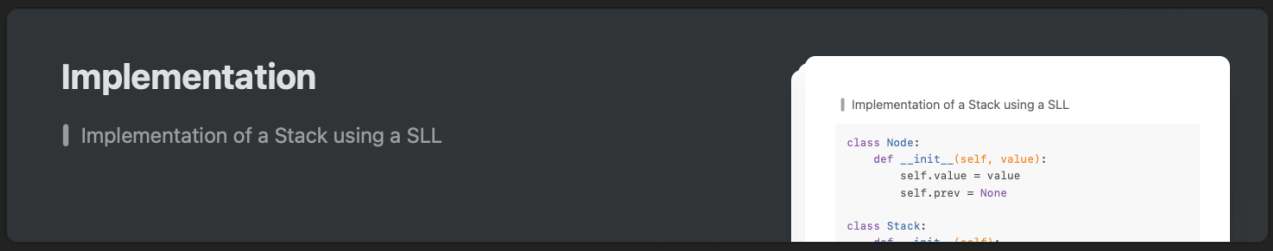


Peek

Check what the first value of the Stack is, without removing it

Use Cases

- Used by undo mechanics
- Used in compiler syntax checking for matching brackets and braces
- Can be used to model stacking books or plates
- Used (BTS) to support recursive function calls
- Can be used to do a depth first search (DFS) on a graph



↑ Stack

Implementation

> Implementation of a Stack using a SLL

```
class Node:
    def __init__(self, value):
        self.value = value
        self.prev = None

class Stack:
    def __init__(self):
        self.length = 0
        self.head = None

    def push(self, item):
        node = Node(item)
        self.length += 1

        if self.head is None:
            self.head = node
            return

        node.prev = self.head
        self.head = node

    def pop(self):
        self.length = max(0, self.length - 1)

        if self.length == 0:
            head = self.head
            self.head = None
            return head.value

        head = self.head
        self.head = head.prev

        return head.value

    def peek(self):
        return self.head.value if self.head is not None else None
```