Stack Fundamentals

[Intro To Stacks](#_2m7cj7otdfki)

[Implementation using array](#_a59idepf5f2l)

[Implementation Using LL](#_fwq40atgqev1)

[Balanced Parenthesis](#_i0x50l9n08zu)

[Remove equal pair of consecutive characters](#_v1df9ijbtmh4)

[Evaluates postfix expressions](#_flyklcy8ztco)

# Intro To Stacks

Stacks are a type of container adaptors with LIFO(Last In First Out) type of working, where a new element is added at one end (top) and an element is removed from that end only. Stack uses an encapsulated object of either vector or deque (by default) or list (sequential container class) as its underlying container, providing a specific set of member functions to access its elements.

Inbuilt functionalities of Stacks in C++ STLs / Operations

**empty()** – Returns whether the stack is empty – Time Complexity : O(1)

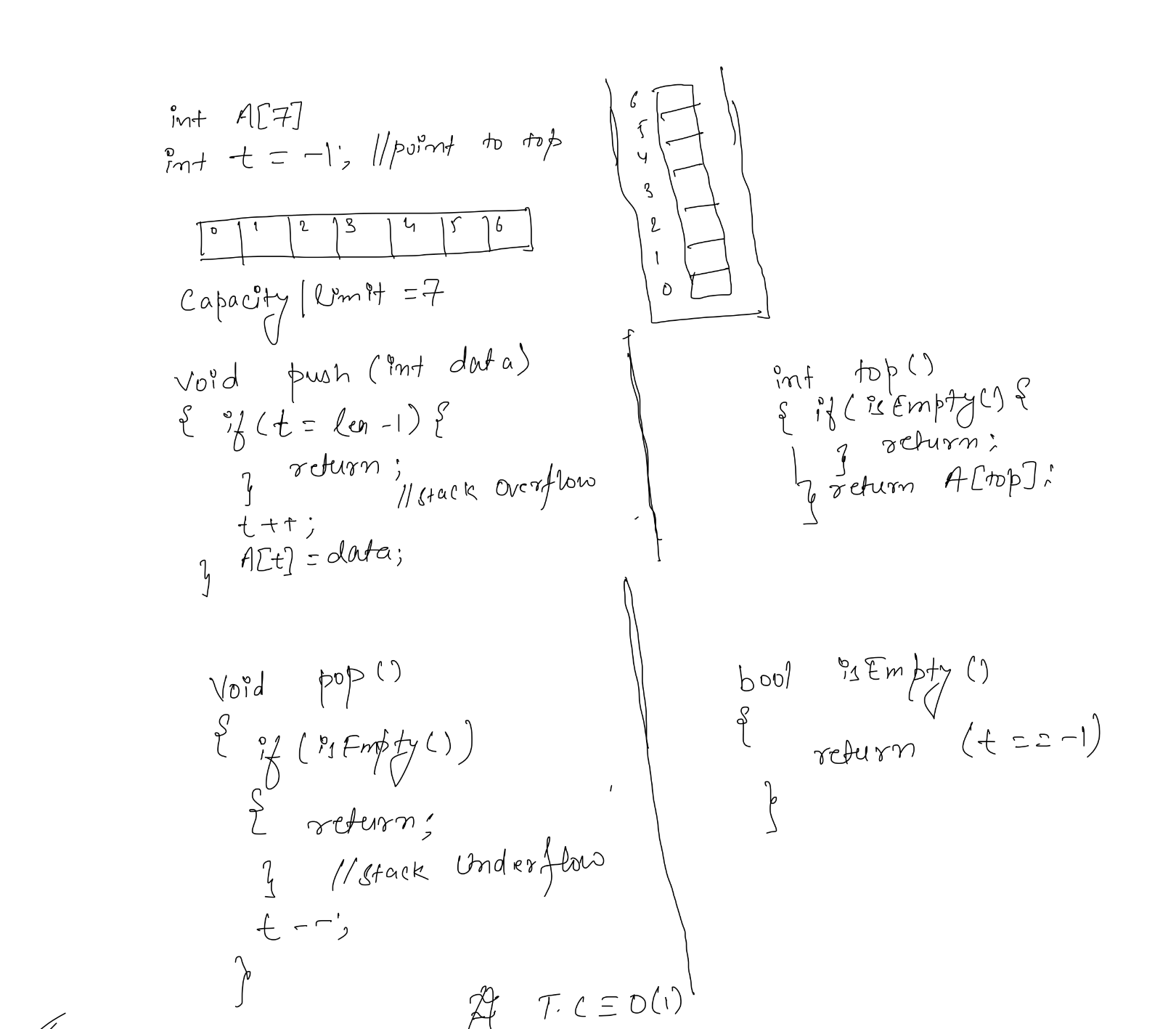
**size()** – Returns the size of the stack – Time Complexity : O(1)

**top()** – Returns a reference to the top most element of the stack – Time Complexity : O(1)

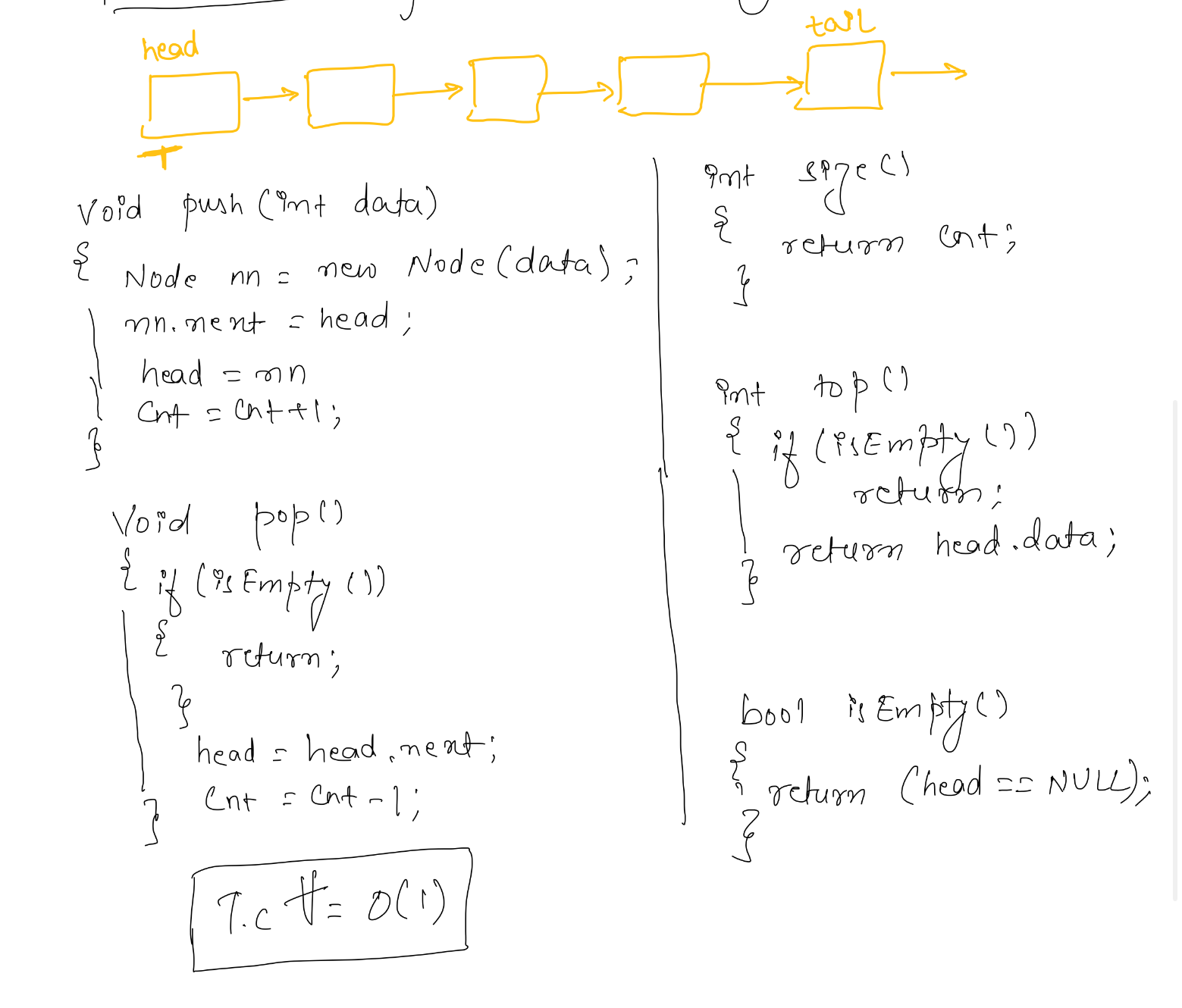
**push(g)** – Adds the element ‘g’ at the top of the stack – Time Complexity : O(1)

**pop()** – Deletes the most recent entered element of the stack – Time Complexity : O(1)

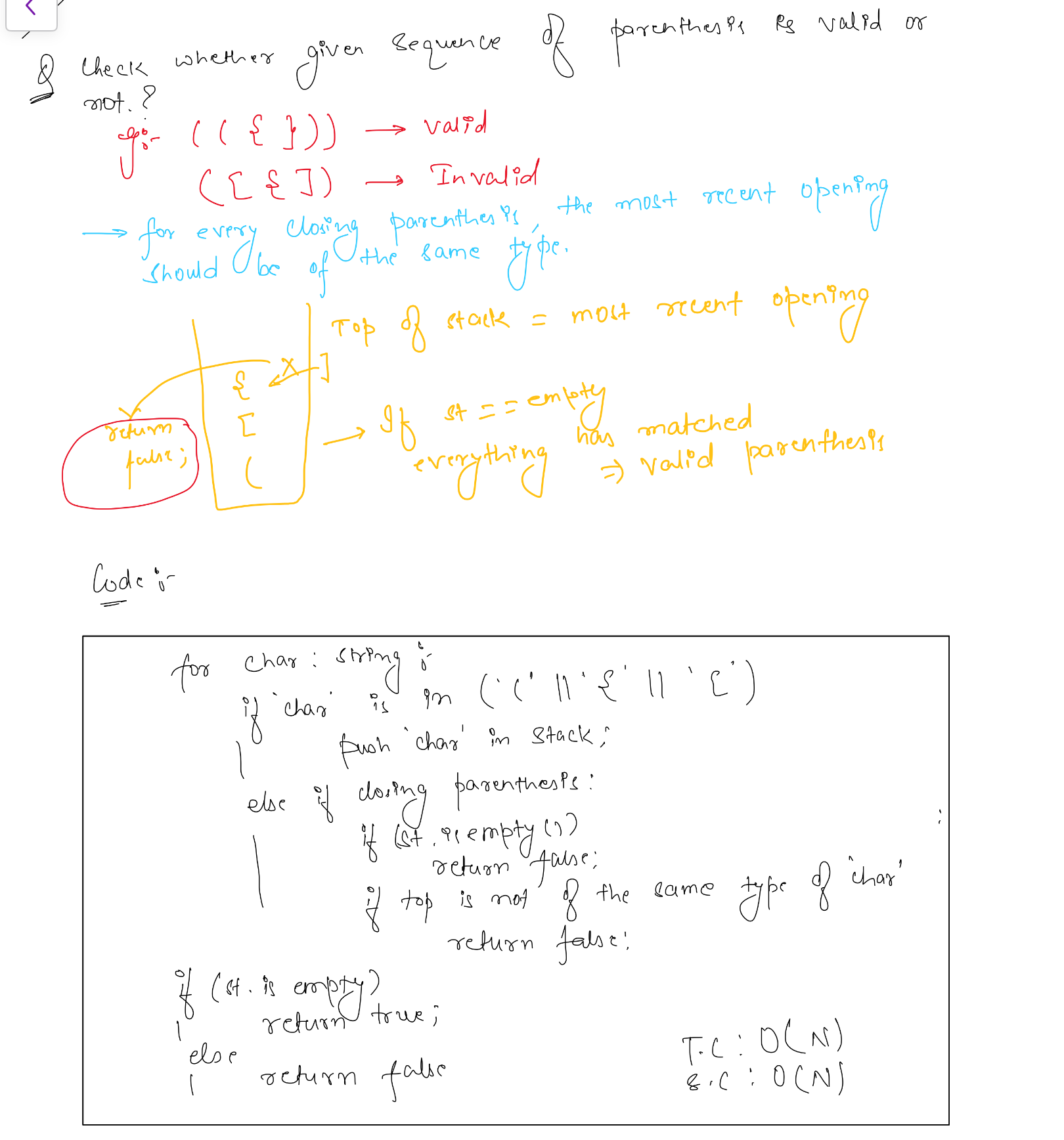
# Implementation using array



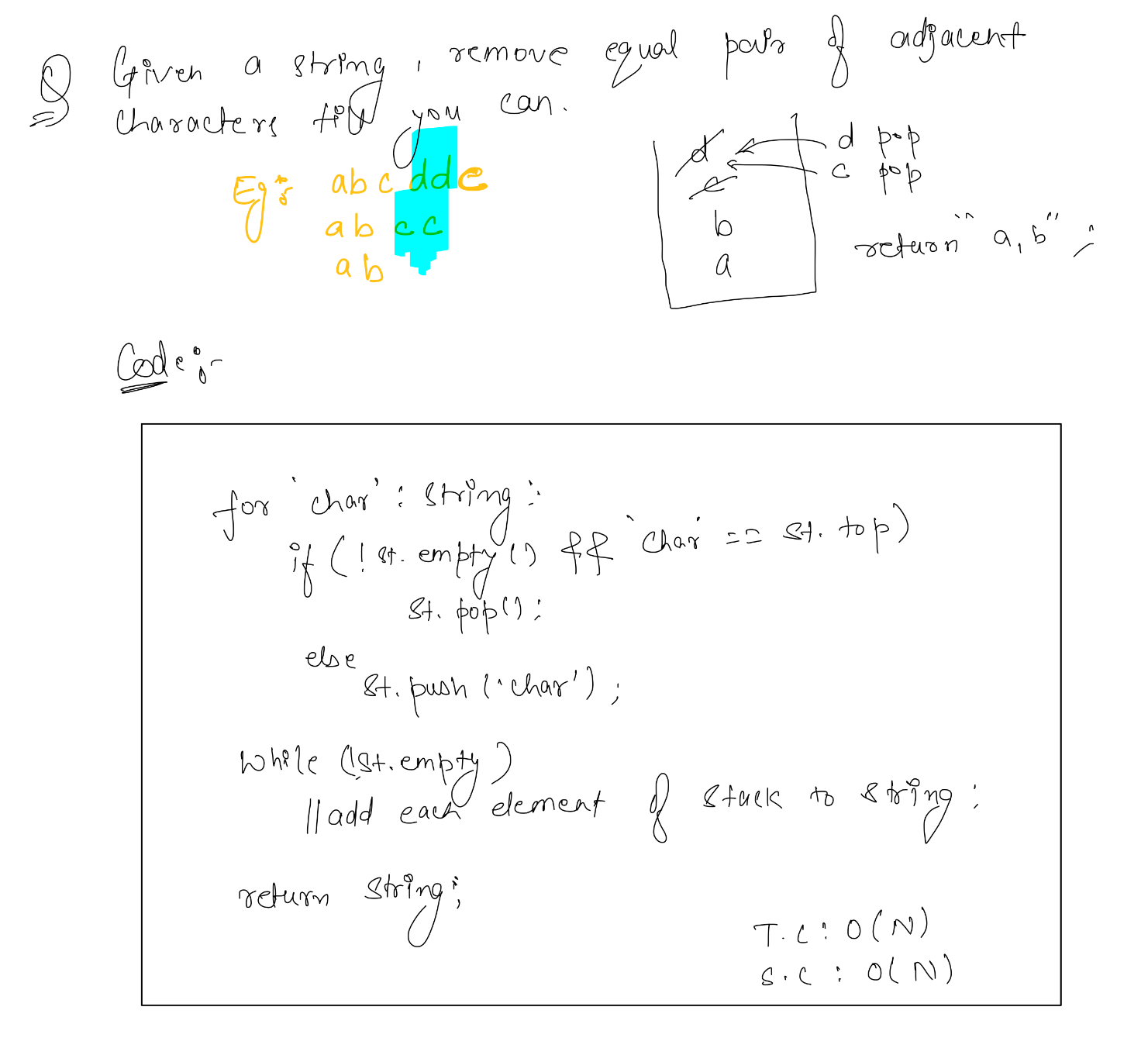
# Implementation Using LL



# Balanced Parenthesis



# Remove equal pair of consecutive characters



# 

# Evaluates postfix expressions

