KAZAKH KBTU BRITISH
TECHNICAL
UNIVERSITY

LECTURE 2 – STACK, QUEUE, DEQUE. SIMPLE & DOUBLE LINKED LISTS.

Prepared by: Izbassar Assylzhan

Course: Algorithms & Data Structures – Fall 2025

LECTURE 2

KAZAKH KBTU BRITISH TECHNICAL UNIVERSITY

In this lecture, we'll continue our study of **stack**, **queue**, and **deque** in greater depth, and solving typical exercises. Then, we'll introduce linked lists, beginning with **singly linked lists** and moving on to **doubly linked lists**, to understand how dynamic memory and node-based organization provide flexibility beyond arrays.

TOPICS WE'LL COVER:

- Stack
- Queue
- Linked List
- Double Linked List

GOALS FOR THIS LECTURE:

- Deepen understanding of stack, queue, and deque by solving typical algorithmic tasks
- Learn the structure and implementation of singly linked lists in C++
- Explore the advantages of doubly linked lists and compare them with arrays and singly linked lists

STACK

Stack

Queue

Linked List

Double Linked List

Stacks are container adaptors that follow the LIFO principle, allowing insertion and removal only from one end.

Basic operations [1]:

- **push** - Insert element
- **pop** - Remove top element
- **empty** - Test whether container is empty
- **size** - Return size
- **swap** - Swap contents

Task: Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid. [2]

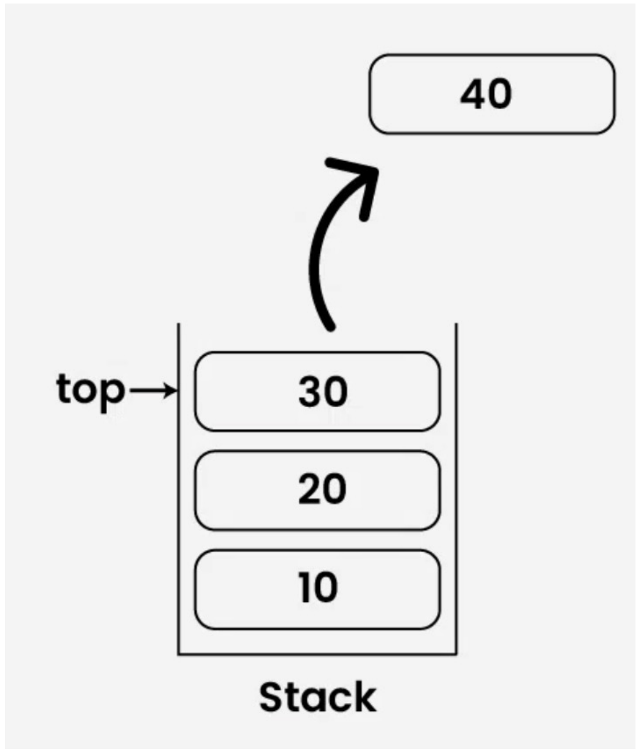


Figure 1 – Stack illustration (ref: geeksforgeeks.org)

[1] https://cplusplus.com/reference/stack/stack/
[2] https://leetcode.com/problems/valid-parentheses/description/?envType=problem-list-v2&envId=stack

QUEUE

Stack

Queue

Linked List

Double Linked List

Queues are container adaptors that work on the FIFO principle, inserting elements at one end and removing them from the other.

Basic operations [3]:

- **front** - Access next element
- **back** - Access last element
- **empty** - Test whether container is empty
- **push** – Insert element to the front
- **pop** – Remove element at the front
- **size** – Return size

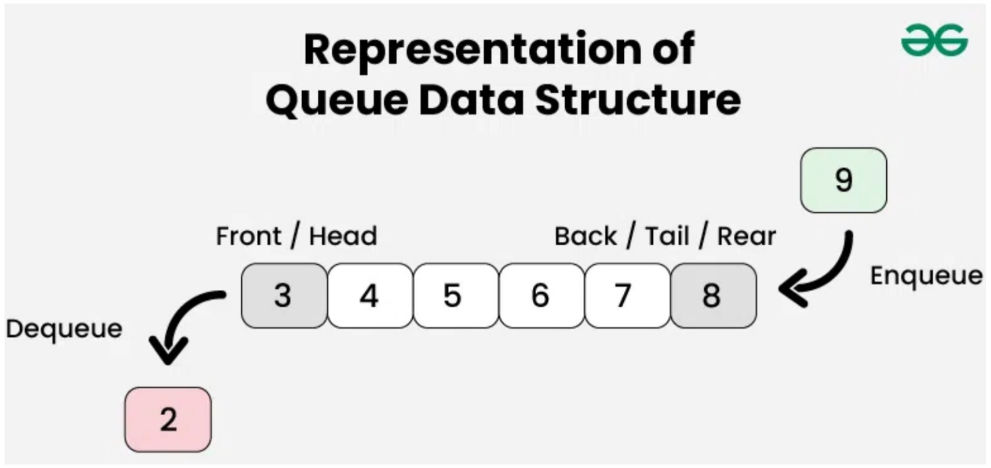


Figure 2 – Queue illustration (ref: geeksforgeeks.org)

Task: Given a string s, find the first non-repeating character in it and return its index. If it does not exist, return -1. [4]

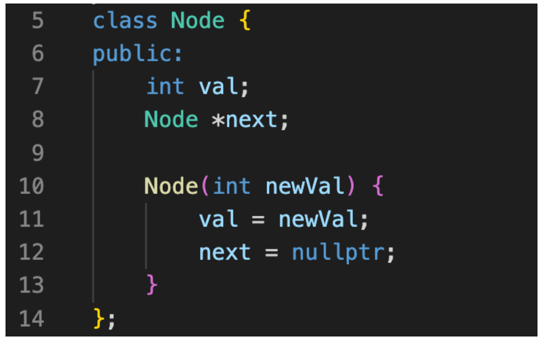
[3] https://www.geeksforgeeks.org/dsa/introduction-to-queue-data-structure-and-algorithm-tutorials/
[4] https://leetcode.com/problems/first-unique-character-in-a-string/?envType=problem-list-v2&envId=queue

LINKED LIST

- Stack
- Queue
- **Linked List**
- Double Linked List

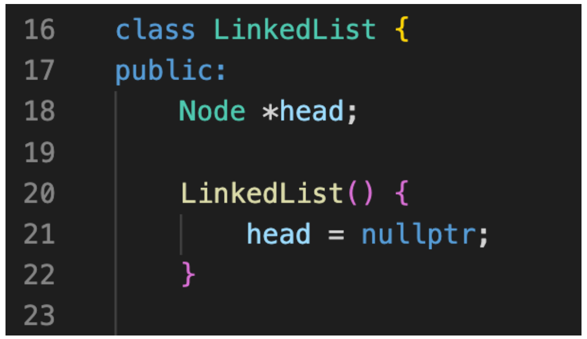
A singly linked list is the simplest linked list where each node holds data and a pointer to the next node.

Task: Given head, the head of a linked list, determine if the linked list has a cycle in it.
- There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Internally, pos is used to denote the index of the node that tail's next pointer is connected to.
- Note that pos is not passed as a parameter. Return true if there is a cycle in the linked list. Otherwise, return false. [6]



```cpp
class Node {
public:
    int val;
    Node *next;

    Node(int newVal) {
        val = newVal;
        next = nullptr;
    }
};
```



```cpp
class LinkedList {
public:
    Node *head;

    LinkedList() {
        head = nullptr;
    }

```

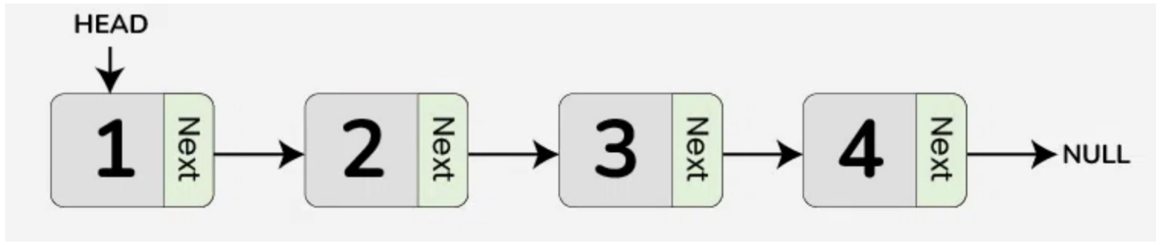


Figure 3– Linked List illustration [5]

[5] https://www.geeksforgeeks.org/cpp/cpp-linked-list/
[6] https://leetcode.com/problems/linked-list-cycle/description/

LINKED LIST (CONT.) - REVERSE

Stack

Queue

Linked List

Double Linked List

Given the **head** of **a linked list**, **reverse the list** and return the new head. [8]

Idea: *The function reverses a linked list recursively by going to the last node, which becomes the new head. On the way back from recursion, it flips each pointer so that* next→next *points back to the current node and then breaks the old link. Finally, it returns the new head of the reversed list.*

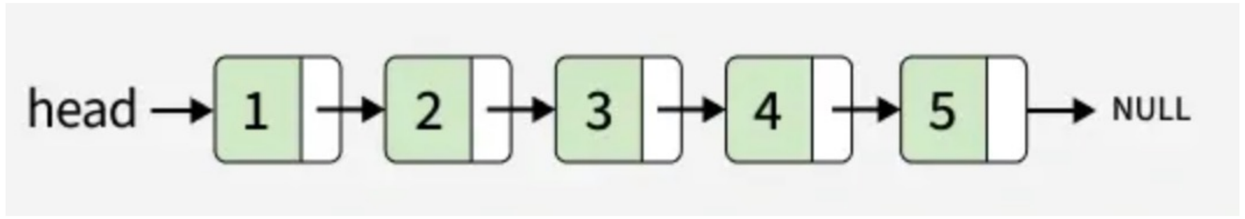


Figure 4 – Input Linked List illustration [7]

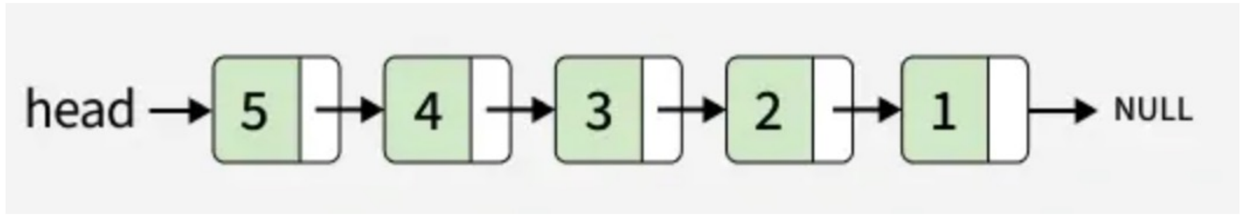


Figure 5 – Output Linked List illustration [7]

[7] https://www.geeksforgeeks.org/dsa/reverse-a-linked-list/
[8] https://leetcode.com/problems/reverse-linked-list/description/

DOUBLE LINKED LIST

Stack

Queue

Linked List

Double Linked List

A doubly linked list is a bidirectional structure where each node contains two pointers: one to the next node and one to the previous node.

Insertion [9]:

```
First, create a newNode
If the list is empty:
        set newNode→next to NULL.
        set newNode→prev to NULL.
        point the head pointer to newNode.
If the list is not empty:
        set newNode→next to head.
        set newNode→prev to NULL.
        set head→prev to newNode.
        point the head pointer to newNode.
```

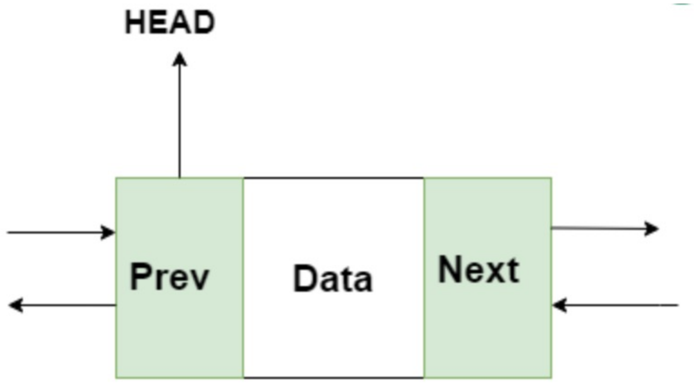


Figure 6 – Node of Double Linked List [9]

Extra task: Given the head of a singly linked list, return true if it is a palindrome or false otherwise. [10]

[9] https://www.geeksforgeeks.org/cpp/doubly-linked-list-in-cpp/
[10] https://leetcode.com/problems/palindrome-linked-list/description/?envType=problem-list-v2&envId=stack

Q & A