TK-Tips: DSA Cheat Sheet – Part 1 (25 Concepts + Diagrams)

SECTION 1 — TIME & SPACE COMPLEXITY

1. Big-O Notation

Describes algorithm performance.
Common complexities:
O(1) < O(log n) < O(n) < O(n log n) < O(n²)

Diagram:
O(1): -------
O(log n): ----////
O(n): ----------
O(n log n): ----///////
O(n²): / / / / / / /

2. Best, Average, Worst Case

Best: Ideal scenario
Avg: Expected performance
Worst: Maximum operations

3. Space Complexity

Memory consumed by algorithm including input + auxiliary space.

SECTION 2 — ARRAYS & STRINGS

4. Array Characteristics

Fixed size, index-based access O(1).
Diagram:
[10][20][30][40][50]

5. Sliding Window Technique

Used for subarray problems like max sum, longest substring.
Diagram:
[L----R] → move window

6. Two Pointer Technique

Used for sorted arrays.
Diagram:
L → ← R

7. String Reversal

Two-pointer swap method.

8. Prefix Sum

Precompute cumulative sums.
Diagram:
arr: [1,2,3]
prefix: [1,3,6]

SECTION 3 — LINKED LISTS

9. Singly Linked List

Nodes pointing to next node.
Diagram:
[10]→[20]→[30]→NULL

10. Fast & Slow Pointer

Used for loops/middle element detection.
Diagram:
S→ F----→

11. Reversing Linked List

Iterative pointer reversal.

12. Detect Loop (Floyd's Algorithm)

Slow + fast pointer.
Diagram:
S→ F→ (cycle)

13. Merge Two Sorted Lists

Use dummy node and merge step-by-step.

SECTION 4 — STACK AND QUEUE

14. Stack (LIFO)

push(), pop(), peek()
Diagram:

Top → [30]
[20]
[10]

15. Queue (FIFO)

enqueue(), dequeue()
Diagram:
Front→[10][20][30]←Rear

16. Monotonic Stack

Used for next greater element problems.

17. Circular Queue

Rear loops to front using modulo.

18. Min Stack

Stack with constant-time min retrieval.

SECTION 5 — TREES

19. Binary Tree

Each node has ≤ 2 children.
Diagram:
10
/ \
5 20

20. Binary Search Tree (BST)

Left < Root < Right

21. Tree Traversals

Preorder, Inorder, Postorder, Level-order.

22. Height vs Depth

Height = longest path to leaf
Depth = distance from root

23. Balanced Tree

Height difference ≤1 everywhere.

SECTION 6 — GRAPHS & ALGORITHMS

24. BFS

Level-order traversal using queue.
Diagram:
Start → neighbors → next level

25. DFS

Recursive or stack-based traversal.
Diagram:
Deep path first.