Table of Contents

- 📘 1. Arrays
  - Concept:
  - Example:
  - Important Questions:
- 📘 2. Strings
  - Concept:
  - Example:
  - Important Questions:
- 📘 3. Linked Lists
  - Concept:
  - Example:
  - Important Questions:
- 📘 4. Stacks & Queues
  - Concept:
  - Example (Stack):
  - Important Questions:
- 📘 5. Trees (Binary Tree, BST, AVL)
  - Concept:
  - Example:
  - Important Questions:
- 📘 6. Graphs (BFS, DFS, Dijkstra)
  - Concept:
  - Example:
  - Important Questions:
- 📘 7. Hashing

- Concept:
- Example:
- Important Questions:

📘 8. Recursion & Backtracking

- Concept:
- Example:
- Important Questions:

📘 9. Greedy & Dynamic Programming

- Greedy: Make the locally optimal choice.
- DP: Solve overlapping subproblems.
- Examples:
- Important Questions:

📘 10. Sorting & Searching

- Sorting Algorithms:
- Searching:
- Important Questions:

- ✅ **Summary Table of Important Questions**

Key Topics:
Arrays, Strings
Linked Lists
Stacks & Queues
Trees (Binary Tree, BST, AVL, etc.)
Graphs (BFS, DFS, Dijkstra, etc.)
Hashing
Recursion and Backtracking
Greedy, Dynamic Programming
Sorting and Searching
expalin this concpets simplwy for my interview and with examples and give all important questions on this topics for my interview and ansl also

Here's a simplified explanation of each key DSA topic, with examples and important interview questions (with answers). These are tailored to help you revise effectively for interviews.

📘 1. Arrays

Concept:

- A collection of elements stored in contiguous memory.
- Indexed from 0.

Example:

```cpp
int arr[5] = {1, 2, 3, 4, 5};
```

Important Questions:

1. Find the maximum subarray sum (Kadane's Algorithm)
   ✅ *Answer*: Initialize `max_so_far = arr[0]`, `max_ending_here = arr[0]`, then iterate and update.
2. Find duplicate in an array
   ✅ *Answer*: Use a hash map or Floyd's Cycle detection if limited space.

📘 2. Strings

Concept:

- Array of characters ending with '\0'.

Example:

```cpp
string s = "hello";
```

Important Questions:

1. Check if two strings are anagrams
   ✅ *Answer*: Sort both strings or use frequency count.
2. Longest Palindromic Substring
   ✅ *Answer*: Use Dynamic Programming or Expand Around Center technique.

📘 3. Linked Lists

Concept:

- Nodes connected by pointers.

Example:

```cpp
struct Node {
  int data;
  Node* next;
};
```

Important Questions:

1. Reverse a linked list
   ✅ *Answer*: Use 3 pointers (prev, curr, next).
2. Detect loop in a linked list
   ✅ *Answer*: Floyd's Cycle Detection (slow and fast pointers).

📘 4. Stacks & Queues

Concept:

- Stack: LIFO (last-in, first-out)
- Queue: FIFO (first-in, first-out)

Example (Stack):

```cpp
stack<int> st;
```

Important Questions:

1. Implement a stack using queues
2. Next Greater Element using a stack
   ✅ *Answer*: Traverse from end, maintain a stack of greater elements.

📘 5. Trees (Binary Tree, BST, AVL)

Concept:

- Hierarchical structure.
- BST: Left < Root < Right
- AVL: Self-balancing BST

Example:

```cpp
struct Node {
  int data;
  Node* left;
  Node* right;
};
```

Important Questions:

1. Inorder, Preorder, Postorder traversal
   ✅ *Answer*: Use recursion or stack.
2. Check if a tree is BST
   ✅ *Answer*: Inorder traversal should be sorted.
3. Lowest Common Ancestor

📘 6. Graphs (BFS, DFS, Dijkstra)

Concept:

- Nodes connected by edges.
- BFS: Queue-based, level-order
- DFS: Stack/recursive, depth-first

Example:

```cpp
vector<int> adj[100];  // adjacency list
```

Important Questions:

1. Detect cycle in a graph
   ✅ *Answer*: Use DFS with parent tracking or Union-Find.
2. Dijkstra's Algorithm for shortest path
   ✅ *Answer*: Use priority queue (min-heap).

📘 7. Hashing

Concept:

- Key-value pair mapping for fast lookup.

Example:

```cpp
unordered_map<int, int> mp;
```

Important Questions:

1. Two Sum Problem
   ✅ *Answer*: Use a hash map to store visited elements.
2. Longest Subarray with Sum K

📘 8. Recursion & Backtracking

Concept:

- Recursion: A function calling itself.
- Backtracking: Trying all possibilities (DFS style).

Example:

```cpp
void recur(int n) {
  if(n == 0) return;
  recur(n - 1);
}
```

Important Questions:

1. N-Queens Problem
   ✅ *Answer*: Use backtracking to try placing queens row by row.
2. Generate all subsets of a string

📘 9. Greedy & Dynamic Programming

Greedy: **Make the locally optimal choice.**

DP: **Solve overlapping subproblems.**

Examples:

- Greedy: Activity selection

- DP: Fibonacci with memoization

Important Questions:

1. 0/1 Knapsack (DP)
   ✅ *Answer*: Use a 2D DP table or space-optimized version.
2. Job Scheduling (Greedy)

📘 10. Sorting & Searching

Sorting Algorithms:

- Quick Sort: Divide and conquer
- Merge Sort: Recursive + merge
- Heap Sort

Searching:

- Binary Search on sorted array

Important Questions:

1. Search in Rotated Sorted Array
   ✅ *Answer*: Modified binary search.
2. Kth Largest Element using Heap

✅ Summary Table of Important Questions

Topic	Key Question	Method/Algorithm
Arrays	Max Subarray Sum	Kadane's
Strings	Longest Palindromic Substring	DP / Center Expansion
Linked List	Reverse Linked List	Iteration (3 pointers)
Stack	Next Greater Element	Stack
Queue	Implement Queue using Stacks	Stack manipulation
Tree	Lowest Common Ancestor	Recursive DFS
BST	Validate BST	Inorder Traversal

Topic	Key Question	Method/Algorithm
Graph	Detect Cycle in Undirected Graph	DFS + Parent Tracking
Dijkstra	Shortest Path	Min Heap / PriorityQueue
Hashing	Two Sum	Hash Map
Recursion	N-Queens	Backtracking
Greedy	Activity Selection	Sort by finish time
DP	0/1 Knapsack	DP Table
Sorting	Merge Sort	Divide and Conquer
Searching	Binary Search	Recursion / Iteration

Would you like a printable PDF or flashcards for fast revision?