Question 1

Graphs: Which traversal naturally finds the shortest path in an **unweighted** graph?

A) BFS

B) DFS

C) Dijkstra

D) Topological sort

Question 1

Graphs: Which traversal naturally finds the shortest path in an **unweighted** graph?

A) BFS

B) DFS

C) Dijkstra

D) Topological sort

Solution

Correct option: A

Question 2

Big-O: Checking if an array of size n contains any duplicate using a HashSet.

A) O(n log n) time, O(1) space

B) $O(n^2)$ time, O(1) space

C) O(n) time, O(n) space

D) O(log n) time, O(n) space

Question 2

Big-O: Checking if an array of size n contains any duplicate using a HashSet.

A) O(n log n) time, O(1) space

B) $O(n^2)$ time, O(1) space

C) O(n) time, O(n) space

D) O(log n) time, O(n) space

Solution

Correct option: C

Question 3

Sorting: Which sorting algorithm is **stable** in its standard form?

A) Heap sort

B) Selection sort

C) Quick sort

D) Merge sort

Question 3

Sorting: Which sorting algorithm is **stable** in its standard form?

A) Heap sort

B) Selection sort

C) Quick sort

D) Merge sort

Solution

Correct option: D

Question 4

Binary search: What must be true to use classic binary search on an array?

A) Array must have unique elements

B) Array must be sorted/monotonic by key

C) Array must have only positive numbers

D) Array must have even length

Question 4

Binary search: What must be true to use classic binary search on an array?

A) Array must have unique elements

B) Array must be sorted/monotonic by key

C) Array must have only positive numbers

D) Array must have even length

Solution

Correct option: B

Question 5

Best pattern for “pair sum = target” on a **sorted** array.

A) Two pointers (left/right)

B) Sliding window

C) Heap

D) DFS

Question 5

Best pattern for “pair sum = target” on a **sorted** array.

A) Two pointers (left/right)

B) Sliding window

C) Heap

D) DFS

Solution

Correct option: A

Question 6

Sliding window: Which is the key condition that makes sliding window work well?

A) Input must be sorted

B) Graph must be acyclic

C) Problem is about a **contiguous** subarray/substring

D) Values must be unique

Question 6

Sliding window: Which is the key condition that makes sliding window work well?

A) Input must be sorted

B) Graph must be acyclic

C) Problem is about a **contiguous** subarray/substring

D) Values must be unique

Solution

Correct option: C

Question 7

String building in loops: Which is typically better for repeated appends in a loop?

A) String with +

B) charAt() on a String

C) StringTokenizer

D) StringBuilder

Question 7

String building in loops: Which is typically better for repeated appends in a loop?

A) String with +

B) charAt() on a String

C) StringTokenizer

D) StringBuilder

Solution

Correct option: D

Question 8

Complexity: Two nested loops each running n times, but the inner loop breaks after 1 iteration always.

A) $O(n^2)$

B) $O(n)$

C) $O(n \log n)$

D) $O(1)$

Question 8

Complexity: Two nested loops each running n times, but the inner loop breaks after 1 iteration always.

A) $O(n^2)$

B) $O(n)$

C) $O(n \log n)$

D) $O(1)$

Solution

Correct option: B

Question 9

Recursion: The most common reason recursion causes stack overflow is:

A) Missing/incorrect base case

B) Using arrays

C) Too many parameters

D) Using global variables

Question 9

Recursion: The most common reason recursion causes stack overflow is:

A) Missing/incorrect base case

B) Using arrays

C) Too many parameters

D) Using global variables

Solution

Correct option: A

Question 10

Backtracking: In “choose → explore → unchoose”, what does “unchoose” usually mean?

A) Sorting the array

B) Returning the final answer immediately

C) Removing last choice / undoing state

D) Marking visited as true

Question 10

Backtracking: In “choose → explore → unchoose”, what does “unchoose” usually mean?

A) Sorting the array

B) Returning the final answer immediately

C) Removing last choice / undoing state

D) Marking visited as true

Solution

Correct option: C

Question 11

Memoization: What problem symptom most strongly signals memoization will help?

A) Input is already sorted

B) All elements are unique

C) Always one pass

D) Same state recomputed many times (overlapping subproblems)

Question 11

Memoization: What problem symptom most strongly signals memoization will help?

A) Input is already sorted

B) All elements are unique

C) Always one pass

D) Same state recomputed many times (overlapping subproblems)

Solution

Correct option: D

Question 12

Linked list: Best approach to find the middle node in one pass.

A) Hashing

B) Fast & slow pointers

C) Sorting

D) Stack

Question 12

Linked list: Best approach to find the middle node in one pass.

A) Hashing

B) Fast & slow pointers

C) Sorting

D) Stack

Solution

Correct option: B

Question 13

Linked list reversal (iterative): At each step, you primarily rewire:

A) current.next

B) head.next

C) tail.next

D) current.prev

Question 13

Linked list reversal (iterative): At each step, you primarily rewire:

A) current.next

B) head.next

C) tail.next

D) current.prev

Solution

Correct option: A

Question 14

Cycle detection : If slow and fast pointers meet, the linked list:

A) Has duplicates

B) Is sorted

C) Has a cycle

D) Has odd length

Question 14

Cycle detection : If slow and fast pointers meet, the linked list:

A) Has duplicates

B) Is sorted

C) Has a cycle

D) Has odd length

Solution

Correct option: C

Question 15

Stack: Classic use case.

A) Level order traversal

B) BFS shortest path in an unweighted graph

C) Topological sorting

D) Valid parentheses / matching brackets

Question 15

Stack: Classic use case.

A) Level order traversal

B) BFS shortest path in an unweighted graph

C) Topological sorting

D) Valid parentheses / matching brackets

Solution

Correct option: D

Question 16

Queue/Deque in Java: A common choice for queue operations :

A) Deque (e.g., ArrayDeque)

B) HashSet

C) TreeMap

D) ArrayList

Question 16

Queue/Deque in Java: A common choice for queue operations :

A) Deque (e.g., ArrayDeque)

B) HashSet

C) TreeMap

D) ArrayList

Solution

Correct option: A

Question 17

Hashing: Why is a HashMap commonly used in “Two Sum”?

A) Maintains sorted order

B) Average O(1) lookup for complement

C) Uses recursion internally

D) Guarantees no collisions

Question 17

Hashing: Why is a HashMap commonly used in “Two Sum”?

A) Maintains sorted order

B) Average O(1) lookup for complement

C) Uses recursion internally

D) Guarantees no collisions

Solution

Correct option: B

Question 18

Hashing: For grouping anagrams, the most common key is:

A) String length only

B) First character

C) Sorted characters (or a frequency signature)

D) Original string

Question 18

Hashing: For grouping anagrams, the most common key is:

A) String length only

B) First character

C) Sorted characters (or a frequency signature)

D) Original string

Solution

Correct option: C

Question 19

Trees: Which traversal visits nodes in “Root, Left, Right”?

A) Inorder (Left, Root, Right)

B) Postorder (Left, Right, Root)

C) Level order

D) Preorder (Root, Left, Right)

Question 19

Trees: Which traversal visits nodes in “Root, Left, Right”?

A) Inorder (Left, Root, Right)

B) Postorder (Left, Right, Root)

C) Level order

D) Preorder (Root, Left, Right)

Solution

Correct option: D

Question 20

Trees: Level-order traversal uses:

A) Queue

B) Stack

C) PriorityQueue

D) Set

Question 20

Trees: Level-order traversal uses:

A) Queue

B) Stack

C) PriorityQueue

D) Set

Solution

Correct option: A

Question 21

Tree height: If height is defined as the number of **edges** on the longest root→leaf path, height of a single-node tree is:

A) 1

B) 0

C) n

D) Depends on BFS/DFS

Question 21

Tree height: If height is defined as the number of **edges** on the longest root→leaf path, height of a single-node tree is:

A) 1

B) 0

C) n

D) Depends on BFS/DFS

Solution

Correct option: B

Question 22

BST: Key property is:

A) All leaves at the same depth

B) Always balanced

C) Left subtree values < node < right subtree values

D) Each node has at most one child

Question 22

BST: Key property is:

A) All leaves at the same depth

B) Always balanced

C) Left subtree values < node < right subtree values

D) Each node has at most one child

Solution

Correct option: C

Question 23

BST validation: A common correct strategy is:

A) Check only parent vs child values

B) Level order should be increasing

C) Postorder should be increasing

D) Inorder strictly increasing (or track min/max range)

Question 23

BST validation: A common correct strategy is:

A) Check only parent vs child values

B) Level order should be increasing

C) Postorder should be increasing

D) Inorder strictly increasing (or track min/max range)

Solution

Correct option: D

Question 24

Heap: In a max-heap, which is always true?

A) Parent ≥ children

B) Left child ≥ right child always

C) Array must be sorted

D) Parent ≤ children

Question 24

Heap: In a max-heap, which is always true?

A) Parent ≥ children

B) Left child ≥ right child always

C) Array must be sorted

D) Parent ≤ children

Solution

Correct option: A

Question 25

Top K frequent elements: Most common combo is:

A) Stack + Queue

B) HashMap + PriorityQueue

C) Bubble sort + Binary search

D) DFS only

Question 25

Top K frequent elements: Most common combo is:

A) Stack + Queue

B) HashMap + PriorityQueue

C) Bubble sort + Binary search

D) DFS only

Solution

Correct option: B

Question 26

Topological sorting is only possible if the directed graph is:

A) Complete

B) Connected

C) A DAG (no directed cycles)

D) Weighted

Question 26

Topological sorting is only possible if the directed graph is:

A) Complete

B) Connected

C) A DAG (no directed cycles)

D) Weighted

Solution

Correct option: C

Question 27

Time complexity: Average time for quicksort (typical implementation) is:

A) O(n)

B) O(log n)

C) $O(n^2)$

D) O(n log n)

Question 27

Time complexity: Average time for quicksort (typical implementation) is:

A) O(n)

B) O(log n)

C) $O(n^2)$

D) O(n log n)

Solution

Correct option: D