1. What is the time complexity of inserting an element at the head of a singly linked list?

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: A) O(1)

Explanation: We only change the head pointer, no traversal needed.

2. What is the time complexity of inserting an element at the tail of a singly linked list (without a tail pointer)?

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: B) O(n)

Explanation: We need to traverse the entire list to reach the tail.

3. What is the advantage of a doubly linked list over a singly linked list?

A) Requires less memory
B) Faster searching
C) Can be traversed in both directions
D) Uses less pointers

Answer: C) Can be traversed in both directions.

Explanation: Doubly linked lists store both next and prev pointers.

4. What is the default value of a Node**'s** next **reference in Java?**

A) 0
B) null
C) false
D) Garbage value

Answer: B) null

Explanation: In Java, uninitialized object references are null.

5. What is the time complexity of deleting a node from the middle of a singly linked list (given only its reference)?

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: B) O(n).

Explanation: You need the previous node to update its next, which requires traversal.

6. What is the time complexity of searching for an element in a singly linked list?

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: B) O(n)

7. What is the time complexity of inserting an element at the tail of a doubly linked list (with a tail pointer)?

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: A) O(1)

Explanation: Direct access to tail makes insertion constant time.

8. What is the space complexity of a singly linked list with n **nodes?**

A) O(1)
B) O(n)
C) O(log n)
D) O(n²)

Answer: B) O(n).

Explanation:
In a singly linked list:

- Each **node** contains **data** and a **reference (pointer)** to the next node.
- So, for **n nodes**, we need space to store **n data elements** and **n pointers**.

- Therefore, the space used **grows linearly** with the number of nodes.

9. Which of the following statements about circular linked lists is true?

A) The last node points to null
B) The last node points to the head
C) It cannot have more than two nodes
D) It always contains an even number of nodes

Answer: B) The last node points to the head.

Explanation: Each node stores data and a pointer.

10. What is the time complexity of reversing a singly linked list?

A) O(1)
B) O(n)
C) O(log n)
D) $O(n^2)$

Explanation: You need to change the direction of n pointers.

11. What is the main drawback of a singly linked list?

A) It requires extra memory
B) It cannot be traversed backward
C) It has O(n) insertion at the head
D) None of the above

Explanation: There's no prev pointer in singly linked list.

Answer: B) It cannot be traversed backward

12. **What is the time complexity of deleting the last node of a singly linked list (without a tail pointer)?**

A) O(1)
B) O(n)
C) O(log n)
D) $O(n^2)$

Answer: B) O(n)

Explanation: Requires traversal to find the penultimate node.

13. What is a Linked List in Java?
a) A data structure that stores elements in contiguous memory locations
b) A linear data structure where each element is a separate object

c) A collection that cannot grow dynamically
d) None of the above
Answer: b) A linear data structure where each element is a separate object.

Explanation: Nodes are dynamically created and linked.

14. What is the key advantage of a Linked List over an array?
a) Random access is faster
b) Memory allocation is contiguous
c) Dynamic size and efficient insertions/deletions
d) Linked Lists are always smaller in size
Answer: c) Dynamic size and efficient insertions/deletions.

Explanation: No need to shift elements like arrays.

15. Which of the following is not a type of Linked List?
a) Singly Linked List
b) Doubly Linked List
c) Circular Linked List
d) Indexed Linked List
Answer: d) Indexed Linked List

Explanation: Indexing is a feature of arrays, not linked lists.

16. Which class in Java provides an implementation of a Linked List?
a) ArrayList
b) LinkedList
c) HashSet
d) Vector
Answer: b) LinkedList

Explanation: Part of java.util package.

17. What is the main difference between a Singly Linked List and a Doubly Linked List?
a) Singly Linked List stores only one element
b) Doubly Linked List has references to both next and previous nodes
c) Singly Linked List cannot store duplicate values
d) None of the above
Answer: b) Doubly Linked List has references to both next and previous nodes.

Explanation: It allows two-way traversal.

18. What is a Circular Linked List?
a) A Linked List that connects back to itself
b) A Linked List with a fixed size

c) A Linked List that cannot store duplicate values
d) None of the above
Answer: a) A Linked List that connects back to itself.

Explanation: The last node's next points to the head.

19. How do you check if a given Linked List is circular?
a) Using a HashSet to track visited nodes
b) Using Floyd's cycle-finding algorithm
c) Using a slow and fast pointer approach
d) All of the above
Answer: d) All of the above.

Explanation: All methods are valid for cycle detection.

20. What is the time complexity for inserting a node at the tail of a Circular Linked List?
a) O(1)
b) O(n)
c) O(log n)
d) O(n log n)
Answer:b

Explanation: Without a tail pointer, you need to traverse to the last node.

21. What is the main disadvantage of a Linked List over an array?
a) Dynamic size
b) Requires more memory due to reference
c) Faster access to elements
d) Easier to use
Answer: b) Requires more memory due to reference.

Explanation: Each node stores an additional pointer.

22. What does each node of a singly linked list contain?
a) Data only
b) Pointer only
c) Data and pointer to next node
d) Data and pointers to both previous and next nodes
Answer: c) Data and pointer to next node.

Explanation: In singly linked list, each node has data and a next reference.

23. What is an advantage of a doubly linked list over a singly linked list?
a) Faster search

b) Uses less memory
c) Can be traversed in both directions
d) Cannot be modified
Answer: c) Can be traversed in both directions.

Explanation: Nodes have next and prev.

24. What is the time complexity for inserting a node in the middle of a doubly linked list?
a) O(1)
b) O(n)
c) O(log n)
d) O(n log n)
Answer: b) O(n)

Explanation: Requires traversal to find the correct insertion point.

25. What is the time complexity for deleting a node from a doubly linked list?
a) O(1) if node address is given
b) O(n) if node address is not given
c) Both a and b
d) O(n log n)
Answer: c) Both a and b.

Explanation:

- **O(1)** if node reference is known
- **O(n)** if search is required

26. How do you traverse a circular linked list?
a) Using recursion
b) Using a do-while loop
c) Using a queue
d) Using an extra pointer
Answer: b) Using a do-while loop.

Explanation: Useful since we check after visiting the node (circular check).

27. Which data structure follows the Last In First Out (LIFO) principle?
a) Queue
b) Stack
c) Linked List
d) Hash Table
Answer: b) Stack

Explanation:- Stack = LIFO - Java's Stack class behaves this way: push() adds, pop() removes the most recent item.

28. What is an algorithm?
a) A programming language
b) A flowchart
c) A set of well-defined steps to solve a problem
d) A computer program
Answer: c) A set of well-defined steps to solve a problem.

Explanation:- Algorithm - A logical sequence of steps, which can be implemented in Java methods.

29. What is the time complexity of searching an element in an unsorted array?
a) O(1)
b) O(log n)
c) O(n)
d) O(n log n)
Answer: c) O(n)

Explanation:- Unsorted Array Search = O(n) - You must check each element; no binary search possible unless sorted.

30. Which of the following notations is the worst-case time complexity?
a) Big-O (O)
b) Big-Omega (Ω)
c) Big-Theta (Θ)
d) Small-O (o)
Answer: a) Big-O (O)

Explanation:- Big-O - Worst-case performance measure (e.g., $O(n^2)$).

31. What is the best case time complexity of QuickSort?
a) O(n)
b) O(n log n)
c) $O(n^2)$
d) O(log n)
Answer: b) O(n log n)

QuickSort Best Case = O(n log n) - When partitioning divides arrays evenly.

32. Which notation represents the average-case complexity?
a) O(n)
b) Θ(n)

c) Ω(n)
d) o(n)
Answer: b) Θ(n)

Explanation:- Θ (Theta) - Average-case complexity notation.

33. If an algorithm has a complexity of O(log n), how does its execution time change when input size doubles?
a) Doubles
b) Increases logarithmically
c) Remains constant
d) Becomes linear
Answer: b) Increases logarithmically

Explanation:- O(log n) means the time grows slowly with input size.

Doubling n increases log n only slightly:

34. Which sorting algorithm has the worst-case time complexity of $O(n^2)$?
a) Merge Sort
b) Quick Sort
c) Bubble Sort
d) Radix Sort
Answer: c) Bubble Sort

Explanation: Bubble Sort repeatedly compares and swaps adjacent elements if they are in the wrong order. In the worst case (reverse sorted array), it performs $O(n^2)$ comparisons and swaps, making it inefficient for large datasets.

35. What is the time complexity of Binary Search?
a) O(n)
b) O(log n)
c) O(n log n)
d) $O(n^2)$
Answer: b) O(log n)

Explanation: Binary Search divides the sorted array in half repeatedly, reducing the search space by a factor of 2 with each step. This results in a logarithmic time complexity of O(log n).

36. Which of the following sorting algorithms is NOT based on comparison?
a) Merge Sort
b) Quick Sort
c) Bubble Sort

d) Counting Sort
Answer: d) Counting Sort.

Explanation: Counting Sort uses an auxiliary array to count occurrences of elements and then calculates positions based on cumulative counts. It does not compare elements directly, unlike Merge, Quick, or Bubble Sort.

37. Which sorting algorithm is best for nearly sorted data?
a) Quick Sort
b) Bubble Sort
c) Insertion Sort
d) Merge Sort
Answer: c) Insertion Sort

Explanation: Insertion Sort is efficient for small or nearly sorted datasets because it minimizes movements. In the best case (already sorted), its time complexity is O(n).

38. Which search algorithm works efficiently on sorted data?
a) Linear Search
b) Binary Search
c) Breadth-First Search
d) Depth-First Search
Answer: b) Binary Search

Explanation:- Binary Search is designed specifically for sorted arrays. It quickly narrows the search interval by comparing the target with the middle element, achieving O(log n) time efficiency.

39. What is the worst case time complexity of inserting a node in a doubly linked list?
a) O(nlogn)
b) O(logn)
c) O(n)
d) O(1)

Answer: c
Explanation: In the worst case, the position to be inserted maybe at the end of the list, hence you have to traverse through the entire list to get to the correct position, hence O(n).

40. How do you calculate the pointer difference in a memory efficient double linked list?
a) head xor tail
b) pointer to previous node xor pointer to next node

c) pointer to previous node – pointer to next node
d) pointer to next node – pointer to previous node

Answer: b
Explanation: The pointer difference is calculated by taking XOR of pointer to previous node and pointer to the next node.