## 📘 UNIT 3 – LINKED LISTS

-------------------------------------------

(Questions 1–20)

Q1. A linked list is a collection of _______.
a) Arrays
b) Objects
c) Nodes
d) Functions

Q2. Each node in a singly linked list contains _______.
a) Data and a pointer
b) Data only
c) Pointer only
d) Two pointers

Q3. The starting node of a linked list is known as _______.
a) Tail
b) Head
c) Start
d) Root

Q4. Fill in the blank: In a singly linked list, the last node’s next pointer contains _______.
a) 0
b) NULL
c) Address of previous node
d) Address of head node

Q5. Which operation requires traversal of the entire linked list?
a) Insert at beginning
b) Insert at end
c) Delete from beginning
d) Display head node

Q6. What is the time complexity to traverse a linked list of n nodes?
a) O(1)
b) O(n)
c) O(n log n)
d) $O(n^2)$

Q7. Which of the following is an advantage of linked lists over arrays?
a) Faster element access
b) Dynamic memory allocation
c) Less memory usage
d) Random access

Q8. Which of the following operations is not possible in a singly linked list?
a) Traversal
b) Searching
c) Reverse traversal
d) Insertion

Q9. Fill in the blank: Insertion at the beginning of a linked list takes _______ time.
a) O(1)
b) O(n)
c) $O(n^2)$
d) O(log n)

Q10. The operation of visiting every node exactly once is called _______.
a) Traversal
b) Searching
c) Sorting

d) Deletion

Q11. What will be the output of the following code?

```
Node* head = NULL;
insertAtBeginning(head, 10);
insertAtBeginning(head, 20);
display(head);
```

a) 10 -> 20
b) 20 -> 10
c) 10 -> NULL
d) 20 -> NULL

Q12. Which of the following is not an advantage of a linked list?
a) Dynamic size
b) Easy insertion and deletion
c) Random access
d) Memory efficient for frequent changes

Q13. The pointer part of the first node in a singly linked list points to _______.
a) Second node
b) Last node
c) NULL
d) Head

Q14. In a circular linked list, the last node points to _______.
a) NULL
b) Second node
c) Head node
d) Random node

Q15. Fill in the blank: To represent a node in a linked list in C++, we use _______.
a) struct
b) class
c) array
d) pointer

Q16. What is the time complexity of inserting a new node after a given node in a linked list?
a) O(1)
b) O(n)
c) $O(n^2)$
d) O(log n)

Q17. What is the primary disadvantage of linked lists over arrays?
a) Fixed size
b) Extra memory for pointers
c) Slower insertion
d) Static allocation

Q18. If we maintain both head and tail pointers, insertion at the end of a singly linked list takes _______.
a) O(1)
b) O(n)
c) $O(n^2)$
d) O(log n)

Q19. Which of the following data structures can be used to implement a stack efficiently?
a) Array
b) Linked list
c) Queue
d) Both a and b

Q20. Which of the following is a real-life analogy of a linked list?
a) Train coaches connected one after another
b) Stack of books
c) Queue of people
d) Circular race track

--------------------------------------------

🔷 ANSWER KEY – UNIT 3 (Q1–Q20)

--------------------------------------------

1. c — Linked list = collection of nodes.
2. a — Each node has data + next pointer.
3. b — The first node is called the head.
4. b — Last node points to NULL.
5. b — Insert at end requires full traversal.
6. b — Traversal takes O(n).
7. b — Linked list supports dynamic memory.
8. c — Singly linked list cannot traverse backward.
9. a — Insertion at beginning is O(1).
10. a — Visiting every node = Traversal.
11. b — After 2 insertions → 20 -> 10.
12. c — Random access not possible in linked list.
13. a — First node points to second node.
14. c — Circular list last node links back to head.
15. a — Nodes are typically represented using `struct`.
16. a — Given node known → insert in O(1).
17. b — Extra pointer memory required.
18. a — With tail pointer, insert at end is O(1).
19. d — Stack can use array or linked list.
20. a — Train coaches represent linked nodes.

--------------------------------------------

📘 END OF SET 1 (UNIT 3: QUESTIONS 1–20)

--------------------------------------------

📘 UNIT 3 – LINKED LISTS

--------------------------------------------

(Questions 21–40)

Q21. Which of the following operations cannot be done efficiently in a singly linked list without a tail pointer?
a) Insert at beginning
b) Insert at end
c) Traverse
d) Delete first node

Q22. In a doubly linked list, each node contains _______.
a) One pointer and one data field
b) Two pointers and one data field
c) Two data fields
d) Three pointers

Q23. Fill in the blank: In a doubly linked list, the left pointer of the first node is _______.
a) NULL
b) Head
c) Tail
d) Random

Q24. In a circular singly linked list, the last node points to _______.
a) NULL
b) Second node

c) Head node
d) Itself

Q25. The time complexity for inserting a node at the end of a singly linked list (without tail pointer) is _______.
a) O(1)
b) O(n)
c) O(n log n)
d) $O(n^2)$

Q26. Which of the following is used to delete a node in a linked list in C++?
a) delete keyword
b) free() function
c) remove() function
d) erase() method

Q27. What is the output of this code?

```
Node* head = NULL;
insertAtBeginning(head, 10);
insertAtEnd(head, 20);
insertAtEnd(head, 30);
display(head);
```

a) 30 -> 20 -> 10
b) 10 -> 20 -> 30
c) 10 -> 30 -> 20
d) 20 -> 10 -> 30

Q28. Which operation is faster in a doubly linked list compared to singly linked list?
a) Traversal
b) Reverse traversal
c) Insertion at end
d) Deletion from beginning

Q29. Fill in the blank: In a circular linked list, the traversal condition is _______.
a) temp != NULL
b) temp == NULL
c) temp != head
d) temp == head

Q30. Which pointer needs to be updated while deleting the first node in a singly linked list?
a) next pointer of head
b) head pointer
c) tail pointer
d) middle pointer

Q31. Which of the following statements about linked lists is TRUE?
a) Linked lists store elements in contiguous memory
b) Linked lists allow random access
c) Linked lists are dynamic in size
d) Linked lists are faster for searching

Q32. What is the space complexity of a singly linked list with n elements?
a) O(n)
b) O(1)
c) O(log n)
d) $O(n^2)$

Q33. Fill in the blank: The process of joining two linked lists into one is called _______.
a) Merging
b) Concatenation

c) Union
d) Linking

Q34. The time complexity to insert a node after a given pointer (node*) is _______.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q35. What will be the output of this pseudo-code?

```
head = NULL;
insertAtBeginning(5);
insertAtBeginning(10);
deleteFromBeginning();
display();
```

a) 10 -> NULL
b) 5 -> NULL
c) NULL
d) Segmentation fault

Q36. Which of the following is TRUE for a doubly linked list?
a) Traversal possible in both directions
b) Only backward traversal possible
c) Only forward traversal possible
d) Cannot delete middle nodes

Q37. Fill in the blank: A doubly linked list node has _______ pointers.
a) One
b) Two
c) Three
d) None

Q38. The time complexity of deleting a node from a doubly linked list when pointer to the node is given is _______.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q39. Which of the following real-world applications can use a circular linked list?
a) Music playlist loop
b) Stack implementation
c) Binary tree
d) Queue with limited size

Q40. What will be the output of this C++ code snippet?

```
struct Node { int data; Node* next; };
Node* head = new Node{1, NULL};
head->next = new Node{2, NULL};
head->next->next = new Node{3, NULL};
Node* temp = head;
while (temp != NULL) {
    cout << temp->data << " ";
    temp = temp->next;
}
```

a) 1 3 2
b) 1 2 3
c) 3 2 1
d) Error

--------------------------------------------

## 🔷 ANSWER KEY – UNIT 3 (Q21–Q40)

------------------------------------------

21. b — Without tail pointer, insertion at end = O(n).
22. b — DLL has two pointers: prev and next.
23. a — Left pointer of first node points to NULL.
24. c — In circular list, last node points to head.
25. b — Traversal needed → O(n).
26. a — Dynamic memory deleted using delete keyword.
27. b — Insertions → 10 -> 20 -> 30.
28. b — DLL allows reverse traversal efficiently.
29. d — Stop when temp == head in circular list.
30. b — Head must move to head->next.
31. c — Linked lists are dynamic and flexible.
32. a — Each node uses space → O(n).
33. b — Joining lists = concatenation.
34. a — If pointer given, insert is O(1).
35. b — After deletion → only 5 remains.
36. a — DLL supports traversal both ways.
37. b — DLL has two pointers (prev, next).
38. a — Pointer given → delete in O(1).
39. a — Music playlist loops use circular lists.
40. b — Output prints sequentially: 1 2 3.

------------------------------------------

## 📘 END OF SET 2 (UNIT 3: QUESTIONS 21–40)

------------------------------------------

## 📘 UNIT 3 – LINKED LISTS

------------------------------------------

(Questions 41–60)

Q41. Which of the following statements about circular linked lists is TRUE?
a) Last node points to NULL
b) Last node points to the first node
c) First node points to NULL
d) It uses two heads

Q42. Fill in the blank: Circular linked lists are most useful for _______.
a) Reversing data
b) Repeated traversal without restarting
c) Searching random nodes
d) Stack operations

Q43. The time complexity for deleting the first node in a circular linked list is _______.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q44. Which of the following operations requires traversal to the last node in a circular linked list?
a) Insertion at beginning
b) Insertion at end
c) Deletion at beginning
d) Traversal

Q45. In a circular linked list with head pointer, what is the condition for an empty list?
a) head == NULL
b) head->next == NULL
c) head->next == head

d) head == tail

Q46. What is the time complexity for searching an element in a circular linked list of n nodes?
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q47. Fill in the blank: In a circular singly linked list, if head = tail->next, then the list is _______.
a) Empty
b) Non-empty and circular
c) Doubly linked
d) Partially connected

Q48. Which of the following linked list types uses two pointers per node and is circular in nature?
a) Doubly linked list
b) Circular doubly linked list
c) Circular singly linked list
d) Hybrid list

Q49. Which of the following operations is easiest in a circular linked list compared to a singly linked list?
a) Traversal
b) Reverse traversal
c) Queue implementation
d) Search

Q50. The main advantage of circular linked list over singly linked list is _______.
a) Efficient memory
b) No need to check NULL condition while traversing
c) Faster insertion
d) Less code complexity

Q51. What is the output of this pseudo-code?

```
head = createCircularList(3);  // nodes: 1 -> 2 -> 3 -> back to 1
temp = head;
count = 0;
do {
    count++;
    temp = temp->next;
} while (temp != head);
print(count);
```

a) 2
b) 3
c) 4
d) Infinite loop

Q52. Fill in the blank: In a doubly linked list, each node contains _______.
a) next and prev pointers
b) prev pointer only
c) next pointer only
d) Two data values

Q53. In a doubly linked list, to delete a node between two nodes, how many pointers are updated?
a) 1
b) 2
c) 3
d) 4

Q54. The time complexity of deleting the last node in a singly linked list (without tail) is _______.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q55. Fill in the blank: To insert a node at a given position in a doubly linked list, we must modify _______ pointers.
a) 1
b) 2
c) 3
d) 4

Q56. Which of the following is an example of a circular doubly linked list?
a) Browser history (back & forward navigation)
b) Stack
c) Binary search tree
d) Queue

Q57. What is the best way to represent a polynomial expression in programming?
a) Array
b) Stack
c) Linked list
d) Queue

Q58. In a circular doubly linked list, each node's next and previous pointers point to _______.
a) Head and tail respectively
b) Adjacent nodes (both directions)
c) NULL
d) Itself

Q59. The time complexity to concatenate two singly linked lists of lengths m and n is _______.
a) O(1)
b) O(m + n)
c) O(m)
d) $O(n^2)$

Q60. Fill in the blank: In a circular linked list implementation of a queue, when front == rear->next, the queue is _______.
a) Full
b) Empty
c) Overflowing
d) Half

--------------------------------------------

🔷 ANSWER KEY – UNIT 3 (Q41–Q60)

--------------------------------------------

41. b — Circular list's last node connects to first.
42. b — Useful for continuous/repeated traversal.
43. a — Head can be updated directly in O(1).
44. b — To insert at end, we need last node.
45. a — Empty circular list when head == NULL.
46. b — Each node checked once → O(n).
47. b — head = tail->next means circular connection exists.
48. b — Circular DLL has both prev and next pointers.
49. c — Circular lists are perfect for queues.
50. b — No NULL checking while traversing circular list.
51. b — Traverses 3 times before returning to head.
52. a — DLL has next and prev pointers.
53. b — Two pointers (prev->next and next->prev) updated.

54. b — Traversal needed to find last node → O(n).
55. c — prev->next, new->next, and next->prev pointers updated.
56. a — Browser history navigation = circular DLL.
57. c — Linked list efficiently represents polynomials.
58. b — Each node connects to previous and next nodes.
59. c — Need to traverse first list → O(m).
60. b — front == rear->next indicates empty queue.

--------------------------------------------

📘 END OF SET 3 (UNIT 3: QUESTIONS 41–60)

--------------------------------------------

📘 UNIT 3 – LINKED LISTS

--------------------------------------------

(Questions 61–80)

Q61. Which of the following best describes a linked list implementation of a stack?
a) Elements are added/deleted from the end
b) Elements are added/deleted from the beginning
c) Elements are added in sorted order
d) Elements are added randomly

Q62. Fill in the blank: In a linked list-based queue, deletion happens from the _______ and insertion happens at the _______.
a) Front, rear
b) Rear, front
c) Middle, front
d) Rear, rear

Q63. Which of the following is TRUE about memory usage in linked lists?
a) Memory is fixed
b) Memory grows dynamically
c) Memory cannot be reused
d) Memory is allocated at compile time

Q64. What happens when you delete a node without updating its previous node’s pointer?
a) Memory leak
b) Program crash
c) Both (a) and (b)
d) Nothing

Q65. What is the space complexity of a singly linked list with n nodes?
a) O(1)
b) O(n)
c) O($n^2$)
d) O(log n)

Q66. The time complexity to search for a value in an unsorted linked list of n nodes is _______.
a) O(1)
b) O(n)
c) O(n log n)
d) O($n^2$)

Q67. Which of the following functions is used in C++ to dynamically allocate a node in a linked list?
a) malloc()
b) new
c) allocate()
d) create()

Q68. Fill in the blank: In C++, a linked list node is created using a _______ or a _______.
a) class, struct
b) array, pointer
c) function, variable
d) queue, stack

Q69. What will be the output of this pseudo-code?

```
insertAtBeginning(10);
insertAtBeginning(20);
insertAtEnd(30);
deleteFromEnd();
display();
```

a) 10 -> 20
b) 20 -> 10
c) 20 -> 10 -> NULL
d) 20 -> 10

Q70. In a singly linked list, deleting a node at a given position requires access to _______.
a) Only the head node
b) The previous node of the target
c) The tail node
d) All nodes

Q71. Fill in the blank: To free all nodes in a linked list, we traverse the list and use the _______ keyword in C++.
a) delete
b) remove
c) free()
d) erase()

Q72. Which of the following is a drawback of linked list over arrays?
a) Dynamic size
b) Extra memory for pointers
c) Easy insertion
d) Sequential access

Q73. Which of the following operations can cause memory leakage if not handled properly in a linked list?
a) Insertion
b) Deletion
c) Traversal
d) Searching

Q74. The time complexity to reverse a singly linked list is _______.
a) O(1)
b) O(n)
c) $O(n^2)$
d) O(log n)

Q75. Fill in the blank: In a linked list, each node is stored in _______ memory locations.
a) Contiguous
b) Random
c) Sequential
d) Same

Q76. Which of the following is a real-world example of using a linked list?
a) Undo/Redo in text editors
b) CPU scheduling
c) Page replacement algorithm
d) All of the above

Q77. The main reason for segmentation faults while using linked lists in C++ is _______.
a) Null pointer dereferencing
b) Syntax errors
c) Stack overflow
d) Compilation failure

Q78. What is the output of this code snippet?

```
Node* head = NULL;
insertAtBeginning(head, 5);
insertAtBeginning(head, 10);
insertAtBeginning(head, 15);
deleteFromPosition(head, 2);
display(head);
```

a) 10 -> 5
b) 15 -> 10
c) 15 -> 5
d) 5 -> 15

Q79. The auxiliary space required to reverse a singly linked list using recursion is _______.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q80. Fill in the blank: In a circular linked list used for a queue, insertion and deletion are done using _______ and _______ pointers.
a) Head, Tail
b) Rear, Front
c) Start, End
d) Prev, Next

--------------------------------------------

## 🔷 ANSWER KEY – UNIT 3 (Q61–Q80)

--------------------------------------------

61. b — Stack adds/removes from the beginning (LIFO).
62. a — Queue deletes from front, inserts at rear.
63. b — Memory grows dynamically at runtime.
64. c — Both memory leak and crash possible.
65. b — Each node takes fixed space → O(n).
66. b — Linear search required → O(n).
67. b — C++ uses `new` for dynamic allocation.
68. a — Nodes can be structs or classes.
69. b — After ops: 20 -> 10.
70. b — Need previous node to update link.
71. a — `delete` keyword frees memory.
72. b — Extra pointer storage = overhead.
73. b — Improper deletion leaks memory.
74. b — Reversal traverses all nodes → O(n).
75. b — Linked list nodes stored randomly.
76. d — All these applications use linked lists.
77. a — Dereferencing NULL pointer causes crash.
78. c — After deleting position 2 → 15 -> 5.
79. b — Recursion adds stack frames per node → O(n).
80. b — Queue via circular list uses front & rear pointers.

--------------------------------------------

📘 END OF SET 4 (UNIT 3: QUESTIONS 61–80)

--------------------------------------------

## 📘 UNIT 3 – LINKED LISTS

--------------------------------------------

(Questions 81–100)

Q81. Which of the following linked list types allows traversal in both directions?
a) Singly linked list
b) Doubly linked list
c) Circular singly linked list
d) None

Q82. The extra memory required for a doubly linked list with n nodes is _______.
a) O(1)
b) O(n)
c) $O(n^2)$
d) O(log n)

Q83. Fill in the blank: In a circular doubly linked list, the previous pointer of the head node points to _______.
a) NULL
b) The tail node
c) The second node
d) Itself

Q84. Which linked list type is best suited for implementing an **Undo/Redo** feature?
a) Singly linked list
b) Doubly linked list
c) Circular singly linked list
d) Stack

Q85. Which of the following operations has O(1) time complexity in a doubly linked list (if pointer to node is known)?
a) Searching a node
b) Insertion after a node
c) Finding the middle element
d) Counting nodes

Q86. What will be the output of the following code?

```
Node* head = NULL;
insertAtEnd(head, 10);
insertAtEnd(head, 20);
insertAtEnd(head, 30);
deleteFromPosition(head, 2);
display(head);
```

a) 10 -> 30
b) 10 -> 20 -> 30
c) 20 -> 30
d) 30 -> 10

Q87. The **tail pointer** in a singly linked list helps to _______.
a) Traverse backwards
b) Insert efficiently at the end
c) Reduce memory usage
d) Access middle node

Q88. Fill in the blank: In a circular doubly linked list, if head == NULL, the list is _______.
a) Full
b) Empty
c) Corrupted

d) Circular

Q89. The time complexity for finding the middle node of a linked list using the **fast and slow pointer** method is _______.
a) O(1)
b) O(log n)
c) O(n)
d) O($n^2$)

Q90. Which of the following is the main disadvantage of doubly linked lists?
a) Cannot traverse backwards
b) Requires extra memory for pointers
c) Difficult to implement
d) Cannot insert nodes

Q91. Which linked list type can efficiently represent **Josephus Problem** (circular elimination)?
a) Singly linked list
b) Doubly linked list
c) Circular linked list
d) Stack

Q92. Fill in the blank: Reversing a singly linked list changes the direction of all _______.
a) Data fields
b) Pointer links
c) Head and tail data
d) Nodes

Q93. What will be the output of this code snippet?

```
Node* head = NULL;
insertAtBeginning(head, 1);
insertAtBeginning(head, 2);
insertAtBeginning(head, 3);
cout << head->next->data;
```

a) 3
b) 2
c) 1
d) 4

Q94. The time complexity to concatenate two doubly linked lists is _______.
a) O(1)
b) O(n)
c) O(log n)
d) O($n^2$)

Q95. Fill in the blank: To avoid memory leaks in C++, every “new” operation must be paired with a corresponding _______.
a) malloc()
b) delete
c) free()
d) dispose()

Q96. Which of the following linked list operations is **not constant time** even with a tail pointer?
a) Insertion at end
b) Deletion from beginning
c) Searching a node
d) Insertion at beginning

Q97. Which type of linked list is best for implementing an **MRU (Most Recently Used)** cache system?

a) Singly linked list
b) Doubly linked list
c) Circular singly linked list
d) Stack

Q98. Fill in the blank: The time complexity of deleting a node with value x in a linked list of n nodes is _______.
a) O(1)
b) O(n)
c) O(log n)
d) O($n^2$)

Q99. Which of the following is TRUE for linked lists and arrays?
a) Both allow random access
b) Linked lists have variable size, arrays are fixed
c) Arrays use more memory
d) Linked lists are stored contiguously

Q100. The main reason to prefer linked lists over arrays is _______.
a) Faster searching
b) Constant-time access
c) Dynamic memory allocation
d) Simpler syntax

--------------------------------------------
🔷 ANSWER KEY – UNIT 3 (Q81–Q100)
--------------------------------------------

81. b — DLL supports forward and backward traversal.
82. b — Each node uses two pointers → O(n) space.
83. b — Head’s prev pointer points to tail in circular DLL.
84. b — DLL supports bidirectional movement (undo/redo).
85. b — Insertion after a known node takes O(1).
86. a — After deletion of position 2: 10 -> 30.
87. b — Tail pointer allows O(1) insertion at end.
88. b — head == NULL → list is empty.
89. c — Two-pointer traversal still O(n).
90. b — Extra memory needed for prev pointer.
91. c — Josephus problem uses circular list elimination.
92. b — Reversal swaps all next pointers.
93. b — Head:3, next:2, output=2.
94. a — Adjust one pointer → O(1).
95. b — `delete` frees memory allocated by `new`.
96. c — Search still O(n).
97. b — DLL efficiently supports MRU updates/removals.
98. b — Need to search node first → O(n).
99. b — Arrays fixed; linked lists dynamic.
100. c — Dynamic allocation is the main advantage.

--------------------------------------------
📘 END OF SET 5 (UNIT 3: QUESTIONS 81–100)
--------------------------------------------

✅ UNIT 3 COMPLETE — LINKED LISTS (100 QUESTIONS)
-------------------------------------------------
Topics Covered:
- Singly, Doubly & Circular Linked Lists
- Operations (Insertion, Deletion, Traversal, Search)
- Code tracing and pointer logic
- Real-world applications
- Complexity and memory management