QUESTIONS ON SINGLE LINKED LIST

1. Create an identical linked list.
2. Print the middle of given linked list.
3. Write a function to get nth node in a linked list.
4. Print till nth node starting from the last node.
5. Move last element to front of a given single linked list.
6. Make middle node head of the list.
7. Delete alternate nodes of a linked list.
8. Add 1 to a number represented as linked list.

   Ex: 1999 is represented as (1-> 9-> 9 -> 9) and adding 1 to it should change it to (2->0->0->0).
9. Add two numbers represented by two different linked lists.

   Input:

   List1: 5->6->3 // represents number 563

   List2: 8->4->2 // represents number 842

   Output:

   Resultant list: 1->4->0->5 // represents number 1405

   Explanation: 563 + 842 = 1405

   Input:

   List1: 7->5->9->4->6 // represents number 75946

   List2: 8->4 // represents number 84

   Output:

   Resultant list: 7->6->0->3->0// represents number 76030

   Explanation: 75946+84=76030
10. Subtract two numbers represented by linked lists.

    Examples:

    Input: l1 = 1 -> 0 -> 0 -> NULL, l2 = 1 -> NULL

    Output: 0->9->9->NULL

    Explanation: Number represented as

    lists are 100 and 1, so 100 - 1 is 099

    Input: l1 = 7-> 8 -> 6 -> NULL, l2 = 7 -> 8 -> 9 NULL

    Output: 3->NULL

Explanation: Number represented as lists are 786 and 789, so 789 - 786 is 3, as the smaller value is subtracted from the larger one.

11. Find the sum of last n nodes of the given linked list.

    Examples:

    Input: 10->6->8->4->12, n = 2

    Output: 16

    Sum of last two nodes:

    12 + **4** = 16

    Input: 15->7->9->5->16->14, n = 4

    Output: 44

12. Pairwise swap elements of list.

    Input: 1->2->3->4->5->6->NULL

    Output: 2->1->4->3->6->5->NULL

    Input: 1->2->3->4->5->NULL

    Output: 2->1->4->3->5->NULL

    Input: 1->NULL

    Output: 1->NULL

13. Remove duplicates from sorted linked list.
14. Find length of loop in linked list
15. Function to check if a singly linked list is palindrome
16. Remove duplicates from an unsorted linked list
17. Remove all occurrences of duplicates from a sorted Linked List
18. Swap nodes in a linked list without swapping data
19. Intersection point of two Linked Lists.

    Test Data and Expected Output:

    List-1: 1 2 7

    List-2: 3 4 5 7

    Intersection found at node with data: 7

    List-3: 1 2 5

    List-4: 3 4 5 7

    No intersection found.

20. Iteratively Reverse a linked list using only 2 pointers.

21. Segregate even and odd nodes in a Linked List

22. Rearrange a Linked List in Zig-Zag fashion(alternate).

23. Adding two polynomials using Linked List.

24. Write a C program that converts a singly linked list into a string and returns it.

    Test Data and Expected Output :

    ```
    Linked List: Convert a Singly Linked list into a string
    -------------------------------------------------------
    Input the number of nodes: 3
     Input data for node 1 : 10
     Input data for node 2 : 20
     Input data for node 3 : 30
    Return data entered in the list as a string:
    The linked list: 10 20 30
    ```

25. Write a C program that converts a singly linked list into an array and returns it.

    Test Data and Expected Output :

    ```
    Linked List: Convert a Singly Linked list into an array
    -----------------------------------------------------------
    Input the number of nodes: 3
     Input data for node 1 : 10
     Input data for node 2 : 20
     Input data for node 3 : 30
    Return data entered in the list as an array:
    The linked list: 10 20 30
    ```

26. Write a C program to merge two sorted singly linked lists into a single sorted linked list.

    Test Data and Expected Output :

    ```
    Two sorted singly linked lists:
    1 3 5 7
    2 4 6
    After merging the said two sorted lists:
    1 2 3 4 5 6 7
    ```

27. Write a C program to remove duplicates from a single unsorted linked list.

    Test Data and Expected Output :

Original Singly List:

1 2 3 3 4

After removing duplicate elements from the said singly list:

1 2 3 4

Original Singly List:

1 2 3 3 4 4

After removing duplicate elements from the said singly list:

1 2 3 4

28. Write a C program that rotates a singly linked list to the right by k places.

Test Data and Expected Output :

Original List: 1 3 4 7 9

Rotate the said singly linked list to the right by 1 places:

9 1 3 4 7

Rotate the said singly linked list to the right by 2 places:

4 7 9 1 3

Rotate the said singly linked list to the right by 4 places:

7 9 1 3 4

29. Write a C program to swap Kth node from the beginning with Kth node from the end in a singly linked list.

Original List: 1 2 3 4 5

List after swapping 2nd node from beginning and end: 1 4 3 2 5

List after swapping 1st node from beginning and end: 5 4 3 2 1

List after swapping 3rd node from beginning and end: 5 4 3 2 1

30. Write a C program that removes elements with even indices from a singly linked list.

Test Data and Expected Output :

Original linked list:

7 6 5 4 3 2 1

Linked list after removing even indices:

6 4 2

31. Write a C program to remove the Nth node from the end of a singly linked list.

Test Data and Expected Output :

Original Singly List:

1 2 3 4 5

Remove 1st node from the end of a singly linked list:

1 2 3 4

Remove 3rd node from the end of a singly linked list:

1 3 4

32. Write a C program to merge k sorted linked lists into a single sorted linked list.

    Test Data and Expected Output :

    List-1:

    10 20 50

    List-2:

    30 40 60

    List-3:

    10 70 100

    After merging the said three sorted lists into one sorted list:

    10 0 20 30 40 50 60 70 100

33. Write a C program to create and reorder a linked list placing all even-numbered nodes ahead of all odd-numbered nodes.

    Test Data and Expected Output :

    Original Singly List:

    1 2 3 4 5 6

    Reorder the said linked list placing all even-numbered
    nodes ahead of all odd-numbered nodes:

    1 3 5 2 4 6

34. Write a C program to reverse a singly linked list in pairs.

    Test Data and Expected Output :

    Original List: 1 2 3 4 5 6

    Reverse a singly linked list in pairs:

    2 1 4 3 6 5

    Original List: 1 2 3 4 5

    Reverse a singly linked list in pairs:

    2 1 4 3 5

35. Write a C program to reverse a singly linked list starting at the first position in blocks of size k.

Test Data and Expected Output :

Given linked list:

1 2 3 4 5 6 7 8

Reverse the first 3 nodes of the said Linked list:

3 2 1 6 5 4 8 7

Reverse the first 5 nodes of the said Linked list:

5 6 1 2 3 7 8 4

36. Write a C program to remove all elements from a singly linked list that are greater than a given value x.

Test Data and Expected Output :

Original singly linked list:

1 2 3 4 5 6 7

Remove all elements from the said linked list that are greater than 5:

1   2 3 4 5

37. Write a C program to find a pair in a singly linked list whose sum is equal to a given value.

Test Data and Expected Output :

Original singly linked list:

1 2 3 4 5 6 7

Find a pair whose sum is equal to 4:

(1,3)

Find a pair whose sum is equal to 11:

(4,7) (5,6)

Find a pair whose sum is equal to 5:

(1,4) (2,3)

Find a pair whose sum is equal to 14:

Pair not found.