Algorithm for deleting an item from singly linked list from the beginning

Input: A pointer to the first node of the singly linked list, say HEAD.

Output: First node of the list will be deleted otherwise suitable unsuccessful message.

Data structure used: A singly linked list where each node contains a data element, say DATA and the address of the immediate next node, say LINK with HEAD holding the address of the first node.

Steps:

1. Begin
2. Set temp = HEAD // 'temp', pointer to node holding the address in HEAD.
3. If temp = NULL
4. Then
5. Print "Linked list is empty, deletion not possible"
6. Else
7. Set HEAD = temp →LINK
8. free(temp) // free(), a procedure to de-allocate memory of a node
9. End

Algorithm for deleting an item from singly linked list from the end

Input: A pointer to the first node of the singly linked list, say HEAD.

Output: Last node of the list will be deleted otherwise suitable unsuccessful message.

Data structure used: A singly linked list where each node contains a data element, say DATA and the address of the immediate next node, say LINK with HEAD holding the address of the first node.

Steps:

1. Begin
2. Set temp1 = HEAD // 'temp', pointer to node holding the address in HEAD.
3. If temp1 = NULL, Then
4. Print "Linked list is empty, deletion not possible"
5. Else
6. If temp1 →LINK == NULL, Then // List contains only one item
7. Set HEAD = temp1 →LINK
8. free(temp1)
9. Else
10. Set temp2 = temp1
11. While (temp2! = NULL)
12. Begin
13. Set temp1 = temp2
14. Set temp2 = temp2 → LINK
15. End While
16. free(temp2)
17. Set temp1 → LINK = NULL
18. End If
19. End If
20. End

Algorithm to delete node from single linked list containing a specific value

Input: A pointer to the first node of the singly linked list, say HEAD.

Output: First node of the list will be deleted otherwise suitable unsuccessful message.

Data structure used: A singly linked list where each node contains a data element, say DATA and the address of the immediate next node, say LINK with HEAD holding the address of the first node.

Steps:

1. Begin
2. If HEAD = NULL, Then
3. Print “Linked list is empty, deletion not possible”
4. Else
5. Set temp1 = HEAD
6. Set temp2 = HEAD
7. While (temp2 != NULL)
8. Begin
9. If temp2 → DATA = VAL, Then
10. Break
11. End If
12. Set temp1 = temp2
13. Set temp2 = temp2 → LINK
14. End While
15. If (temp2 = NULL), Then //search for node with ‘VAL’ is unsuccessful
16. Print “Value not found”
17. Else
18. If temp2 = HEAD // ‘VAL’ is found in the first node
19. Then
20. Set HEAD = temp2 →LINK // updating the address of the first node
21. free(temp2) // deletion of node with ‘VAL’ found in first node
22. Else
23. Set temp1 →LINK = temp2→LINK //updating the address of the previous node
24. free(temp2) // deletion of node with ‘VAL’
25. End If
26. End If
27. End If
28. End

Algorithm to delete node from single linked list from particular position

Input: A pointer to the first node of the singly linked list, say HEAD.

Output: First node of the list will be deleted otherwise suitable unsuccessful message.

Data structure used: A singly linked list where each node contains a data element, say DATA and the address of the immediate next node, say LINK with HEAD holding the address of the first node.

Steps:

1. Begin

2. temp1 = HEAD // ‘temp1’, pointer to node holding the address in HEAD.
3. If temp1 = NULL, Then
4. Print “Linked list is empty, insertion not possible in the beginning”
5. Exit
6. End If
7. Set count = 0
8. While temp1 != NULL
9. Begin
10. count = count +1
11. temp = temp → link
12. End While
13. If count < POS Or POS <= 0 Then
14. Print “POS specified exceeds the no of nodes in the list or Invalid POS specified ”
15. Exit
16. End If
17. temp1 = HEAD
18. If POS =1 Then
19. HEAD = temp1 →link // Second node is made the first node
20. Free(temp1)
21. Else If POS = count Then
22. temp2 = temp1
23. While (temp2→link != NULL)
24. Begin
25. temp1= temp2
26. temp2 = temp→link
27. End While
28. free(temp2)
29. temp1 → link = NULL
30. Else
31. Set i = 1
32. temp2 = HEAD
33. While i<POS
34. Begin
35. temp1 = temp2
36. temp2 = temp2 →link
37. Set i = i+1
38. End While
39. temp1 →link = temp2 →link
40. Free(temp2)
41. End If
42. End