Spring 2019 (Course Number 009952)

Data Structures and Lab
(Lecture 04: Singly Linked List)

Prof. A. P. Shrestha, Ph.D.
Dept. of Computer Science and Engineering, Sejong University

Last Class

- C++ Simple Program Exercises
  - Concepts of **class**, **object** and **constructors**
- Singly Linked List
  - Insertion (at the end) and Deletion (for a given key)

Today

- Singly linked list operations
  -Insertion, Deletion

Next class

- Doubly linked list

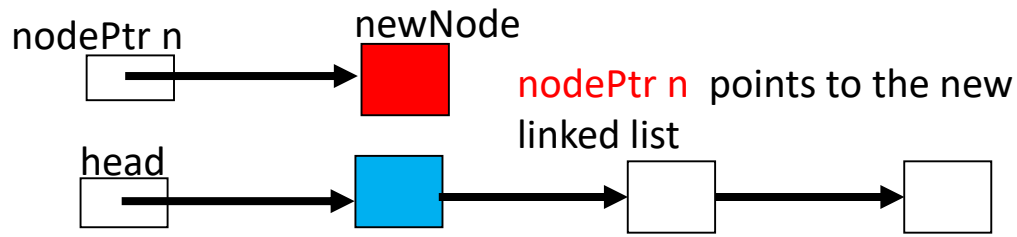
4.1.1 Linked List Operations

- **Add** an item to the linked list
  - 3 situations to consider:
    - insert a node at the front
    - insert a node at a particular position
    - insert a node at the end (covered in last lecture)

- **Delete** an item from the linked list
  - We have 3 situations to consider:
    - delete the node at the front
    - delete a node at a particular position
    - delete the last node
  - Also, delete a node with a given key (covered in last lecture)

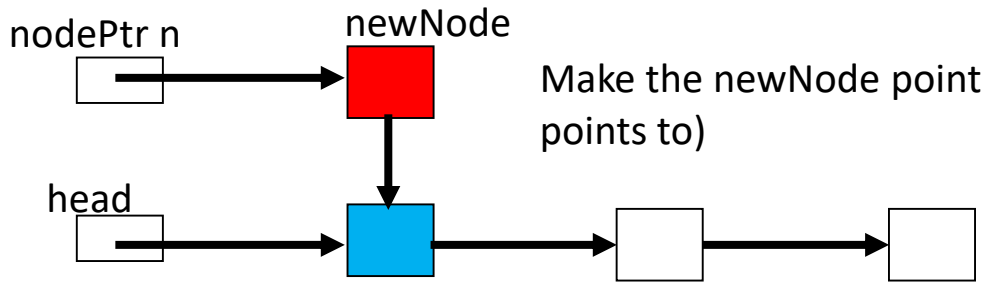
4.1.2 Inserting a Node at the Front

Step 1



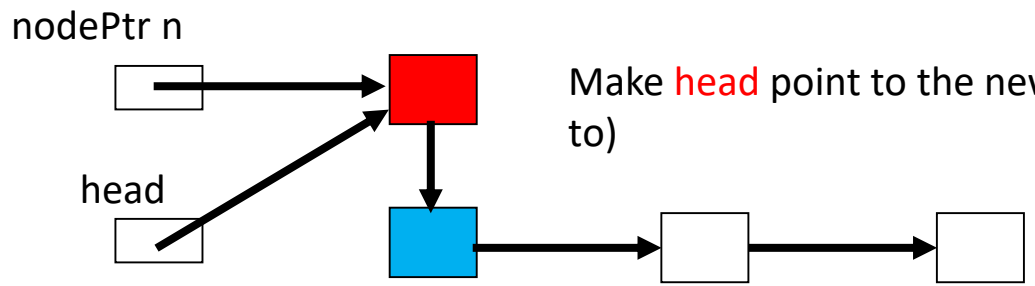
nodePtr n points to the new node, head points to the first node of the linked list

Step 2



Make the newNode point to the first node (i.e. the node that head points to)

Step 3



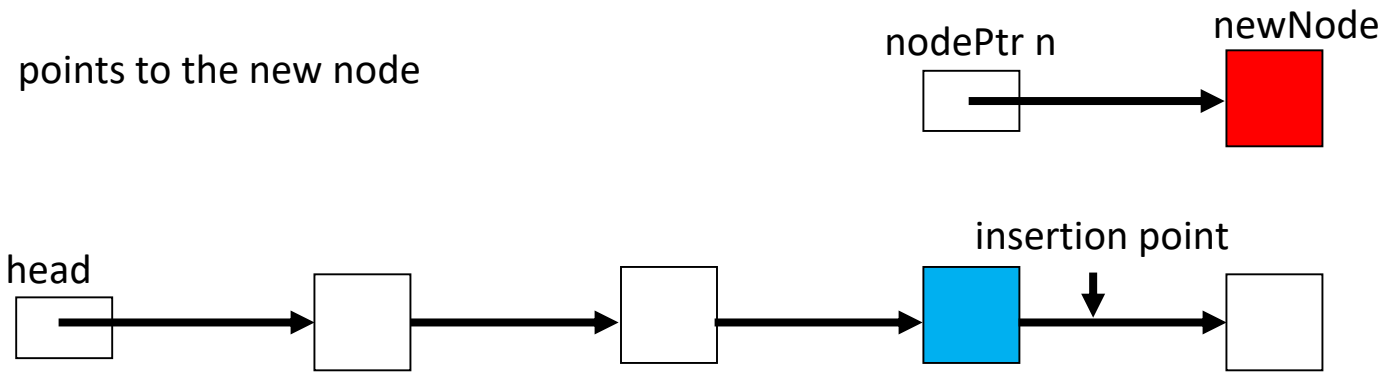
Make head point to the new node (i.e the node that nodePtr n points to)

4.1.3 Inserting a Node at a Particular Position

Let's insert the new node *after the third node* in the linked list

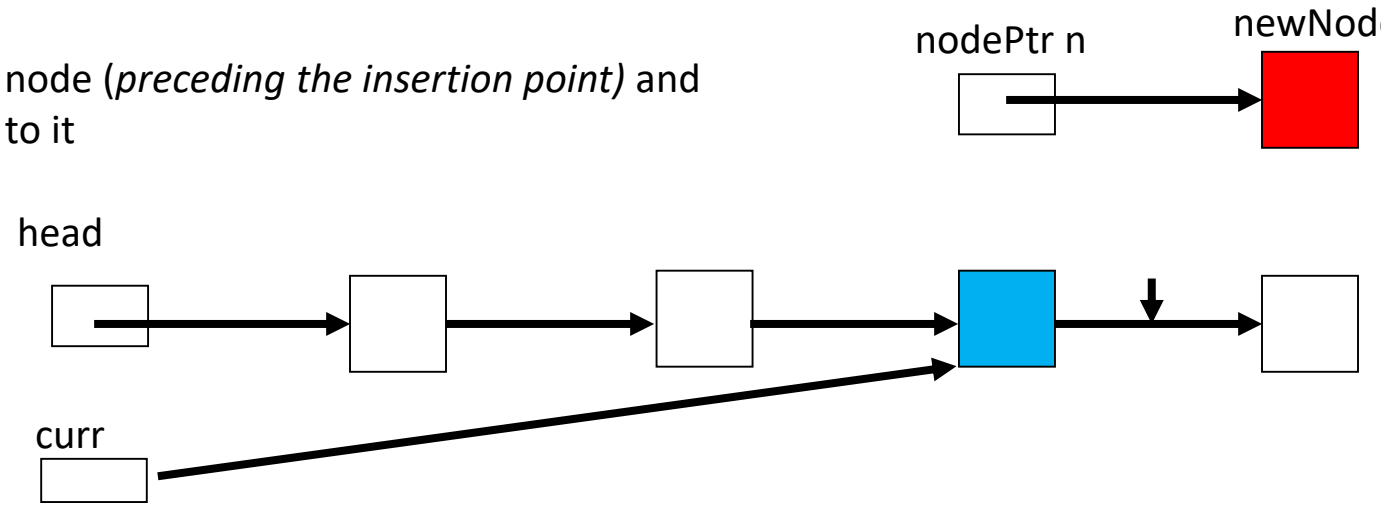
Step 1

nodePtr n points to the new node



Step 2

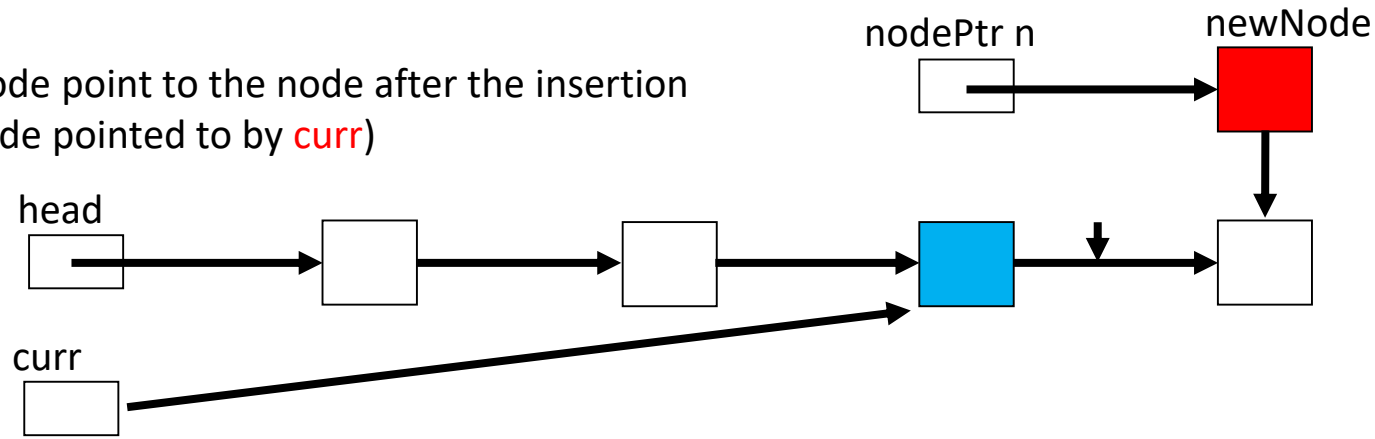
Locate the third node (*preceding the insertion point)* and make curr point to it



4.1.4 Inserting a Node at a Particular Position

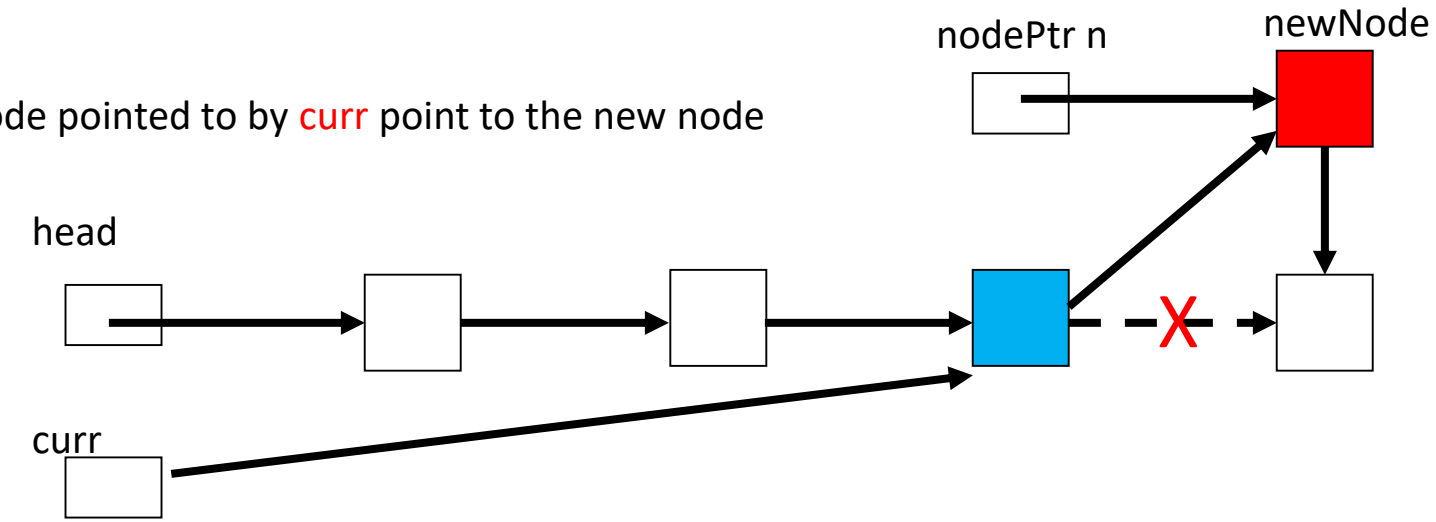
Step 3

Make the newNode point to the node after the insertion point (i.e. the node pointed to by curr)



Step 4

Make the node pointed to by curr point to the new node



Q) Is inserting a node at the end a special case?

4.1.5 Inserting a Node at the End

Q) Discuss the steps to insert a node at the end.

4.1.5 Inserting a Node at the End

Q) Discuss the steps to insert a node at the end.

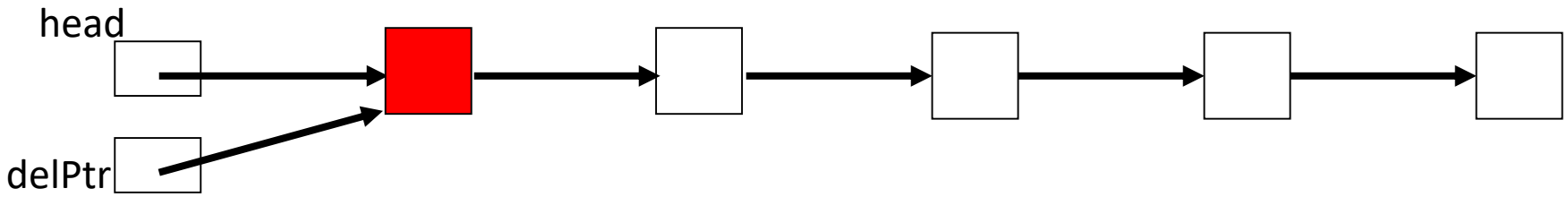
1. The nodePtr n points to the newNode (the next pointer of the newNode points to NULL)
2. Traverse to the last node
3. The next pointer of the last node points to the new node.

4.2.1 Deleting the First Node

Step 1

The head points to the first node in the linked list, which points to the second node

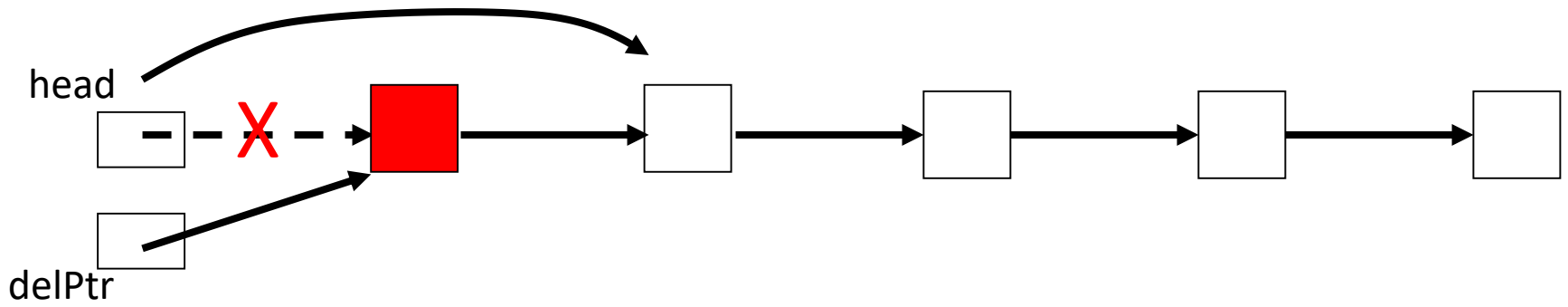
Make delPtr also point to the same node as that of head



Step 2

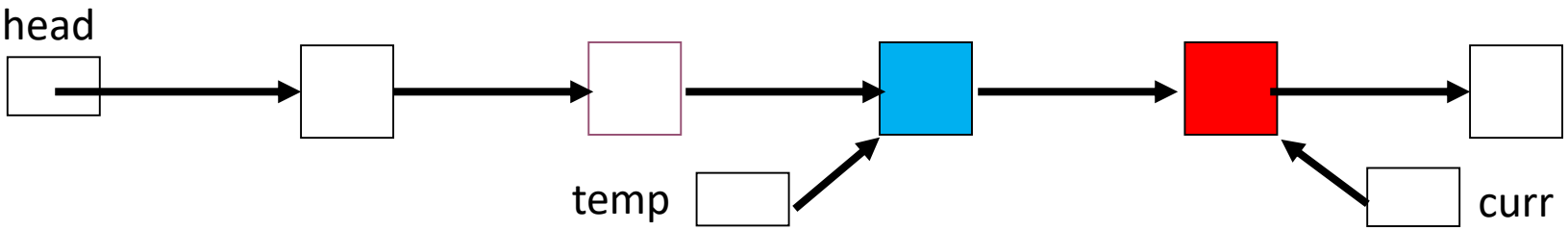
Make head point to the second node (i.e. the node pointed to by the first node)

Dispose of the node pointed by delPtr.



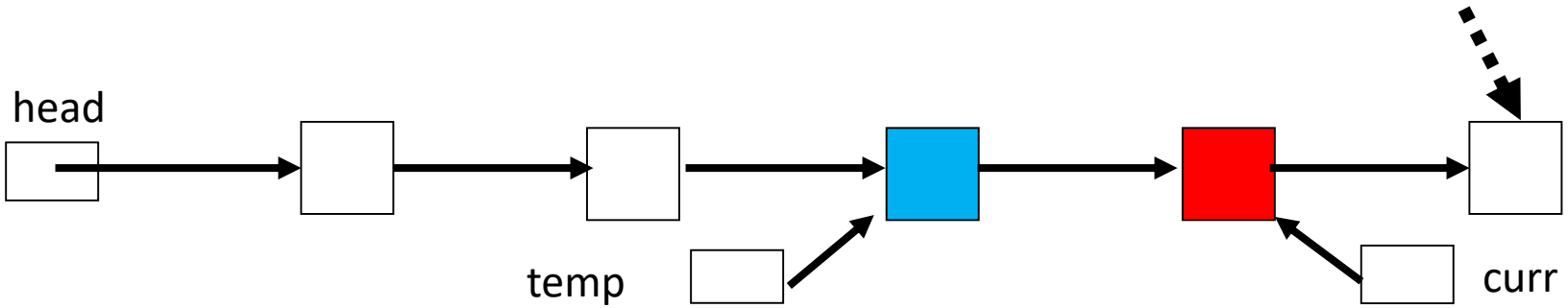
4.2.2 Deleting a Node at a Particular Position

Step 1



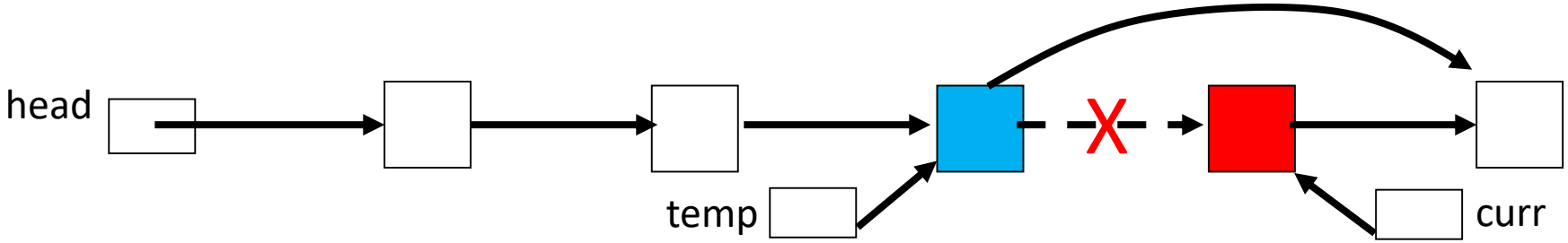
Traverse the linked list so that curr points to the node to be deleted and temp points to the node prior to the one to be deleted

Step 2



Locate the node following the one to be deleted (i.e. the node pointed to by the node that curr points to)

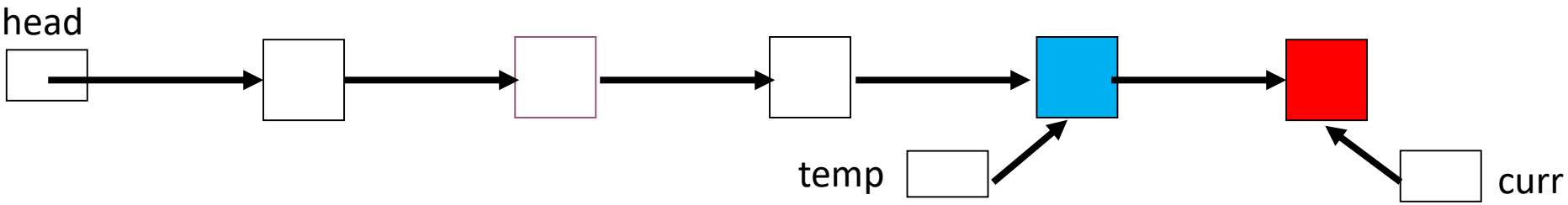
Step 3



Make the node that temp points to point the node following the one to be deleted(i.e. curr->next) and dispose the node pointed by curr

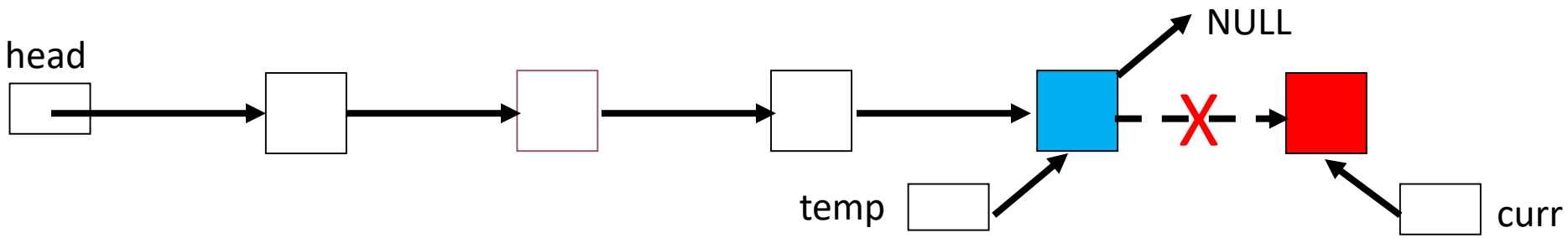
4.2.3 Deleting a Node at the End

Step 1



Traverse the linked list with curr all the way to end while maintaining the previous node address also (temp) also.

Step 2



Update the temp's next pointer to NULL

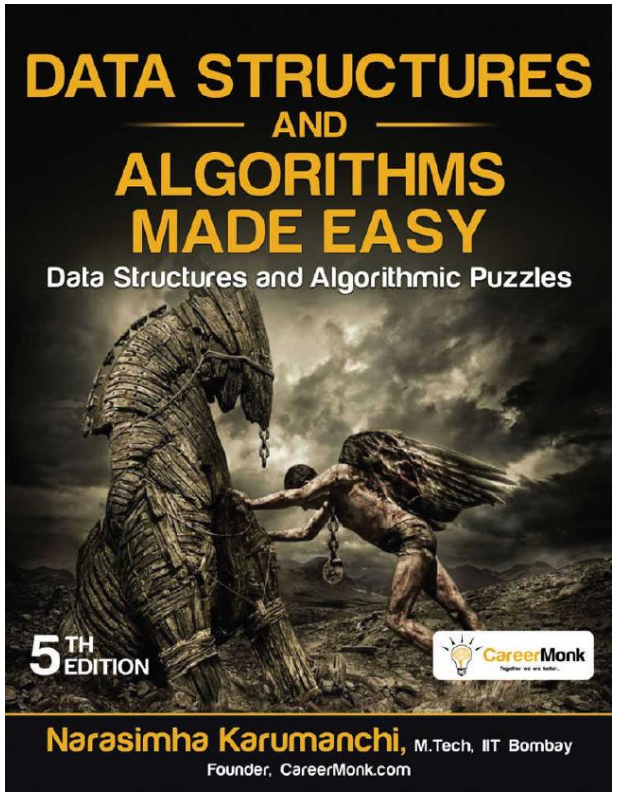
Dipose the tail node pointed by curr.

4.3 Deleting Singly Linked List

- Store curr (current node) in temp (temporary variable)
- Dispose (free/delete) curr
- Go to next node using next pointer in temp
- Repeat this process for all nodes

Note:

- If C++ is difficult, you can check out this book. The codes are in C language.
- Explanation is also very easy and uses simple English!!
- Don’t hesitate to contact me if you need any help.



Q & A ?