Hello Everyone

Course Code : CSE201

Course Title : Data Structure

Course Teacher:

Md. Jamal Uddin

Assistant Professor

Department of CSE,BSMRSTU,

Gopalganj-8100.

Presented By:

Tushar Sarkar

Student ID: 18CSE035

Department of CSE,BSMRSTU,

Gopalganj-8100.

Linked List

Outline

- Introduction
- Insertion Description
- Deletion Description
- Basic Node Implementation
- Complexity
- Conclusion

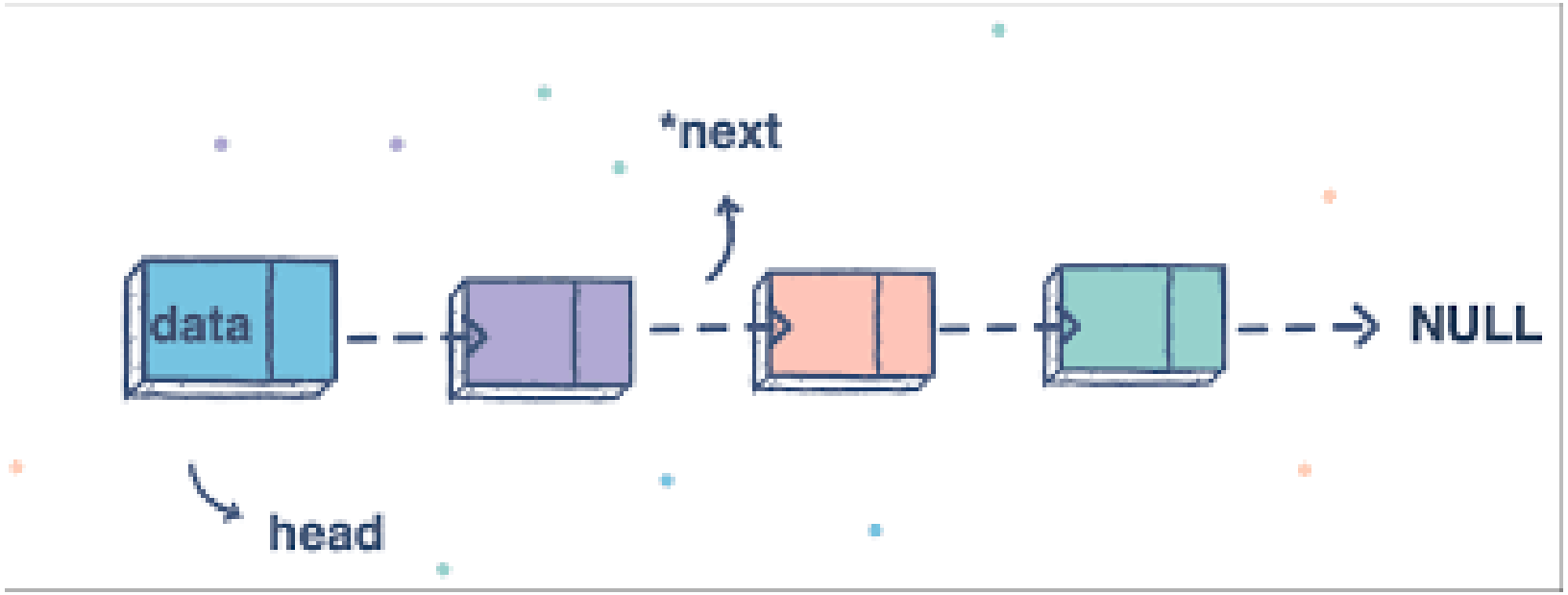
Introduction

- Definitions
  - What is link list?
  - Nodes and pointers
  - Single Linked Lists
  - Double Linked Lists
  - Circular Lists

What is link list?

Link list is the linear collection of data elements called nodes, where the linear order is given by means of pointers.

Example:

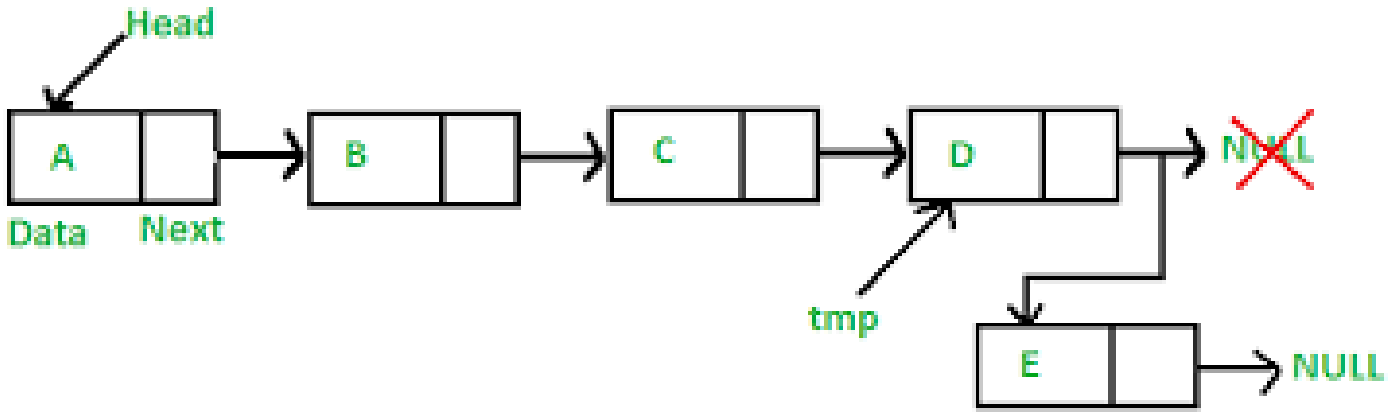


Nodes and pointers

A node is called a self-referential object, since it contains a pointer to a variable that refers to a variable of the same type. For example, a struct Node that contains an int data field and a pointer to another node can be defined as follows.

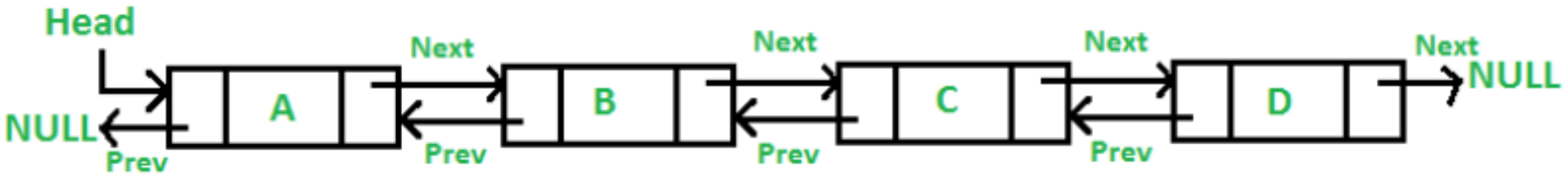
Single linked lists

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image:



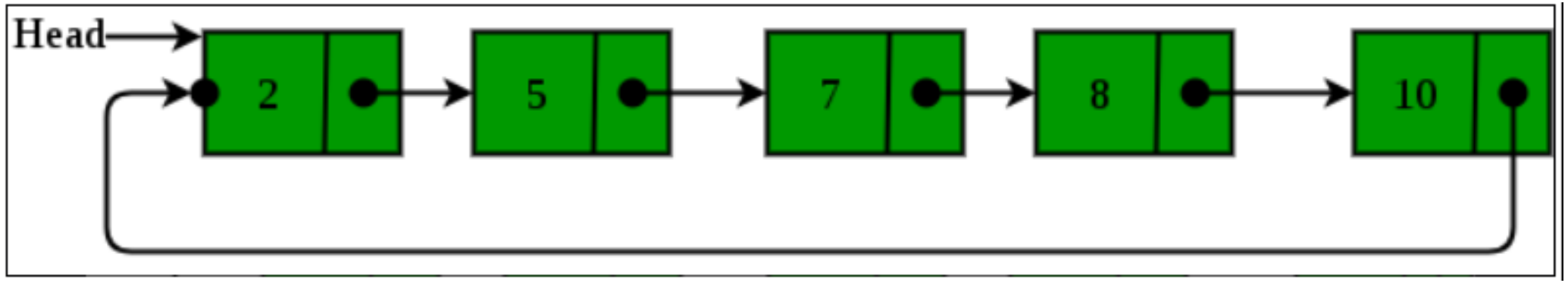
Double Linked Lists

A Doubly Linked List (DLL) contains an extra pointer, typically called *previous pointer*, together with next pointer and data which are there in singly linked list.



Circular Lists

Circular linked list is a linked list where all nodes are connected to form a circle. There is no NULL at the end. A circular linked list can be a singly circular linked list or doubly circular linked list.



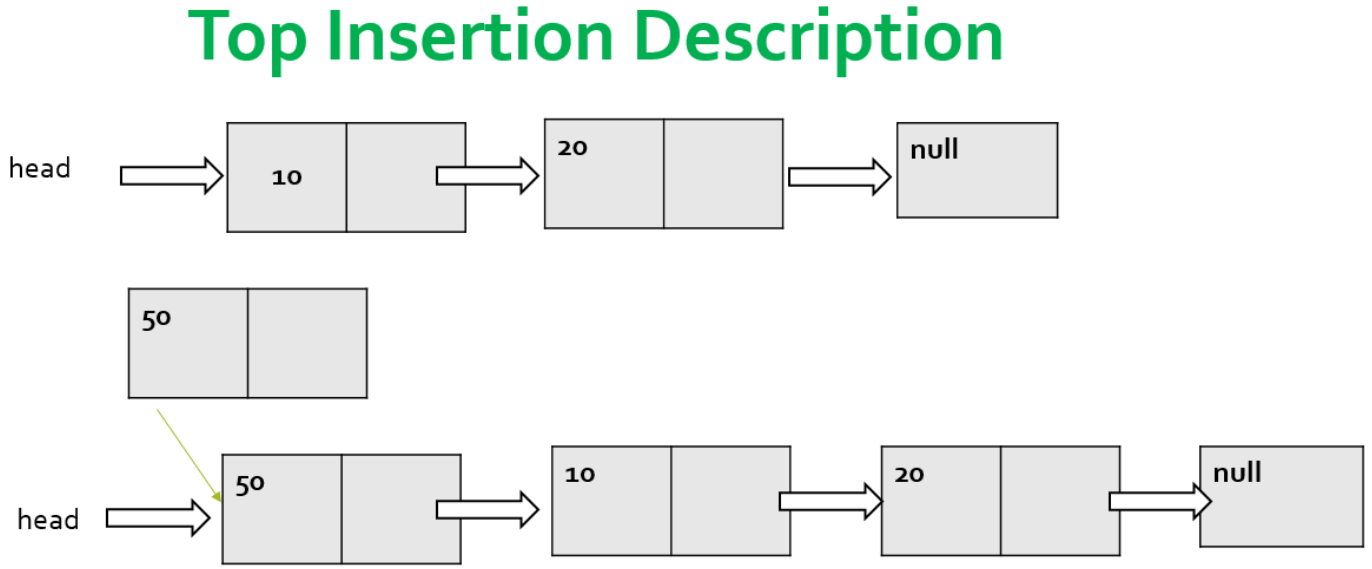
Insertion Description

- Insertion at the top of the list
- Insertion at the end of the list
- Insertion in the middle of the list

Insertion at the top

Steps:

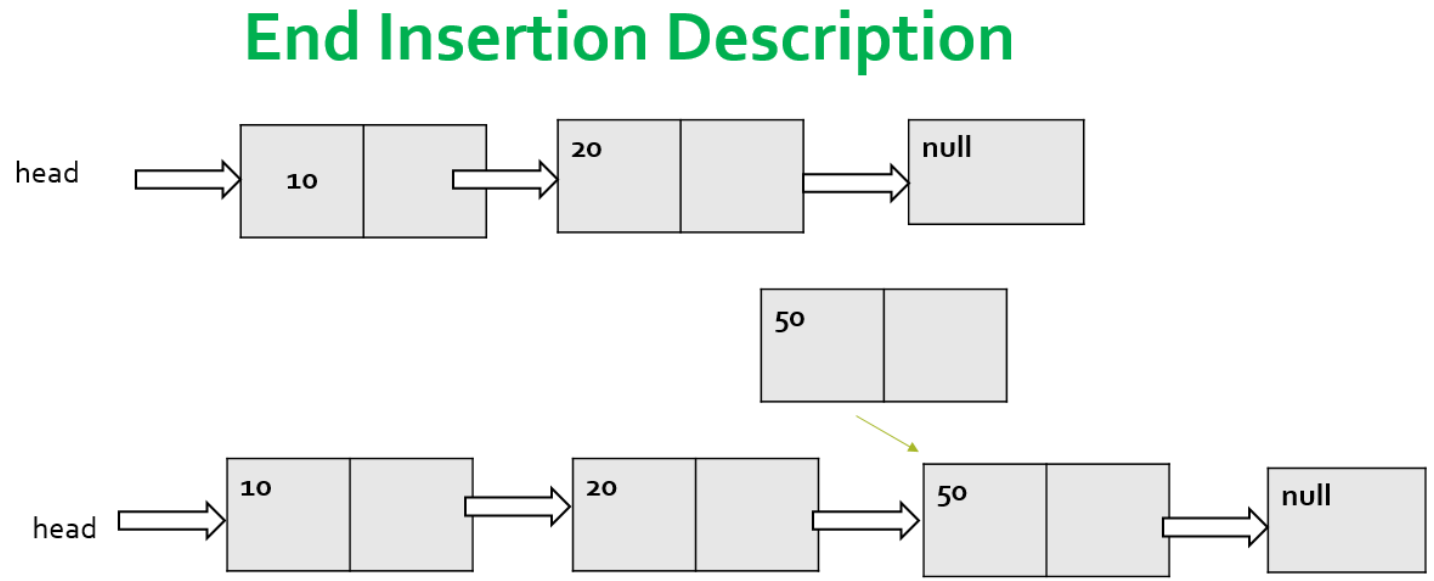
- Create a Node
- Set the node data Values
- Connect the pointers



Insertion at the end

Steps:

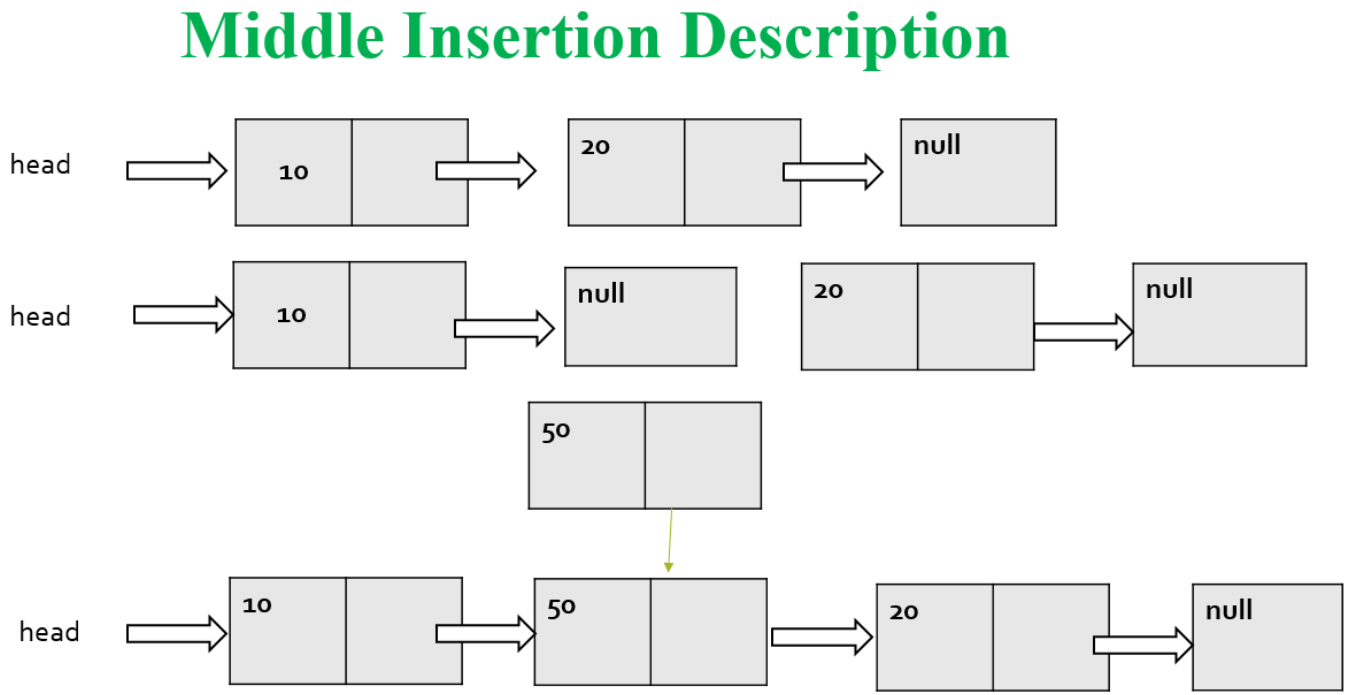
- Create a Node
- Set the node data Values
- Connect the pointers



Insertion in the middle

Steps:

- Create a Node
- Set the node data Values
- Break pointer connection
- Re-connect the pointers



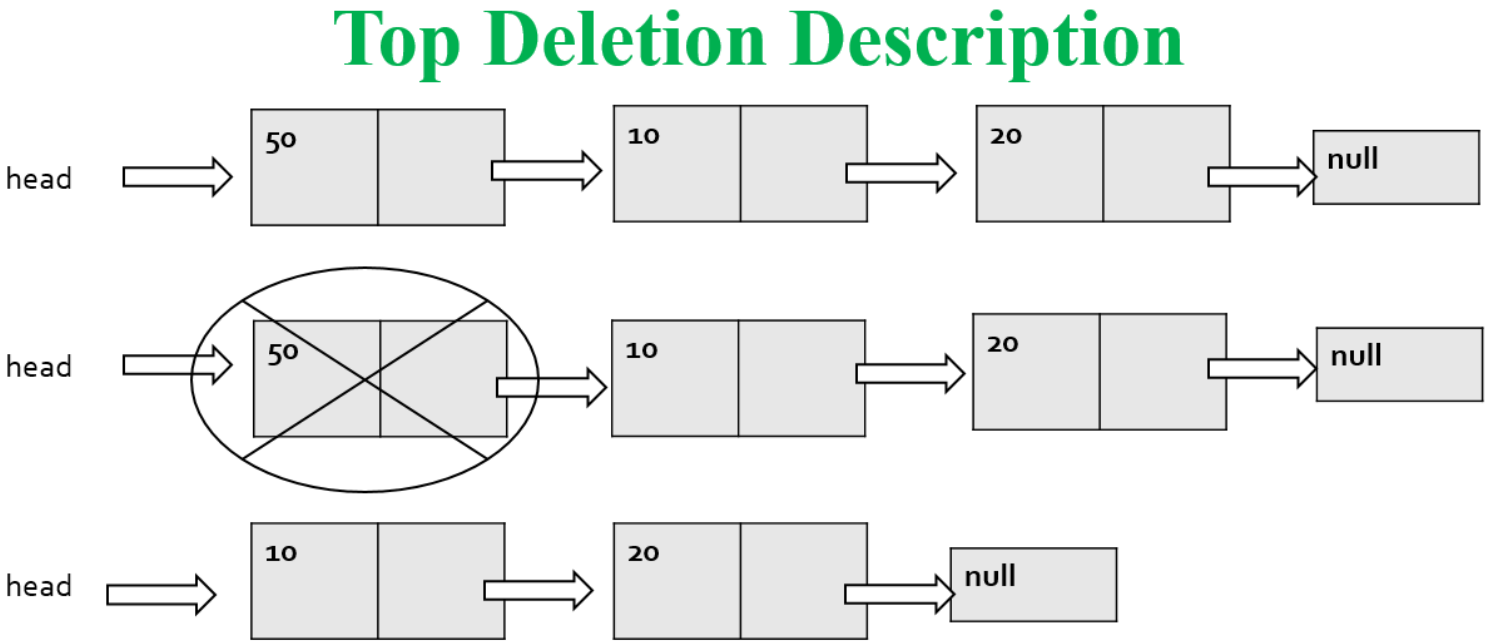
Deletion Description

- Deleting from the top of the list
- Deleting from the end of the list
- Deleting from the middle of the list

Deleting from the top

Steps:

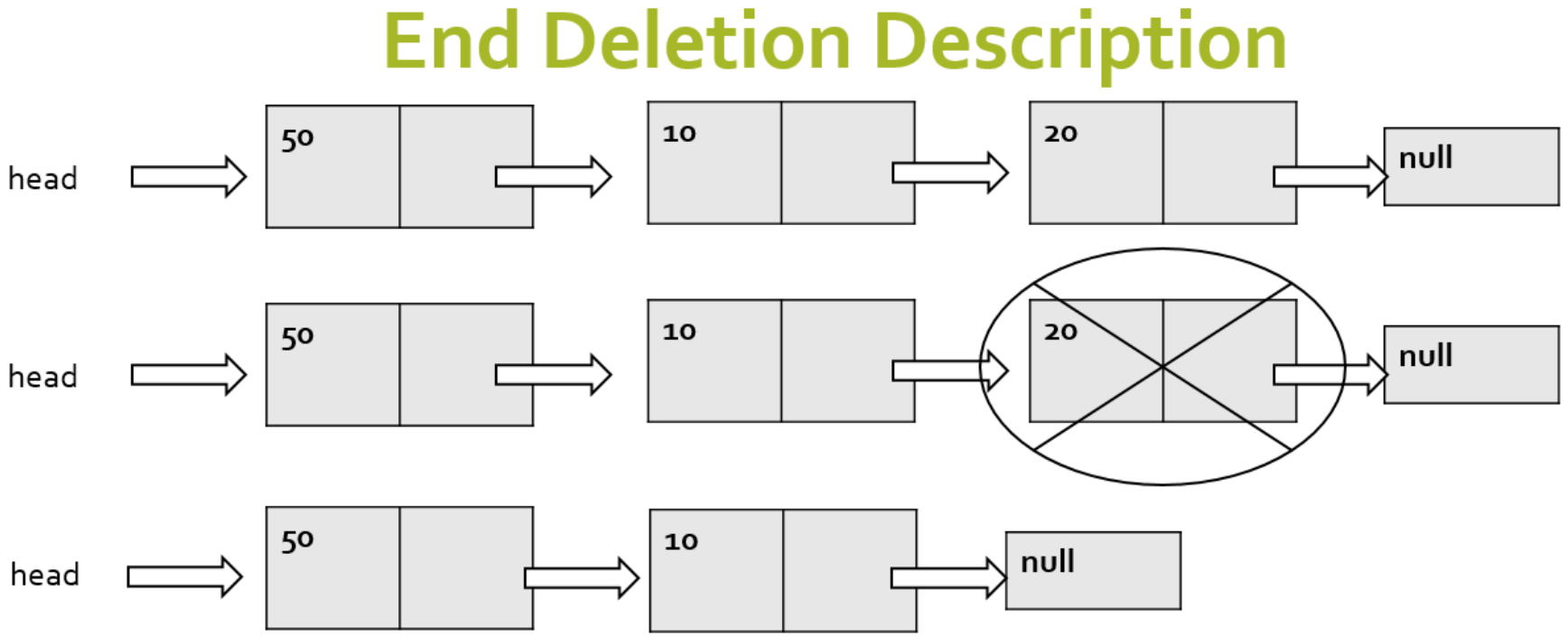
- Break the pointer connection
- Re-connect the nodes
- Delete the node



Deleting from the end

Steps:

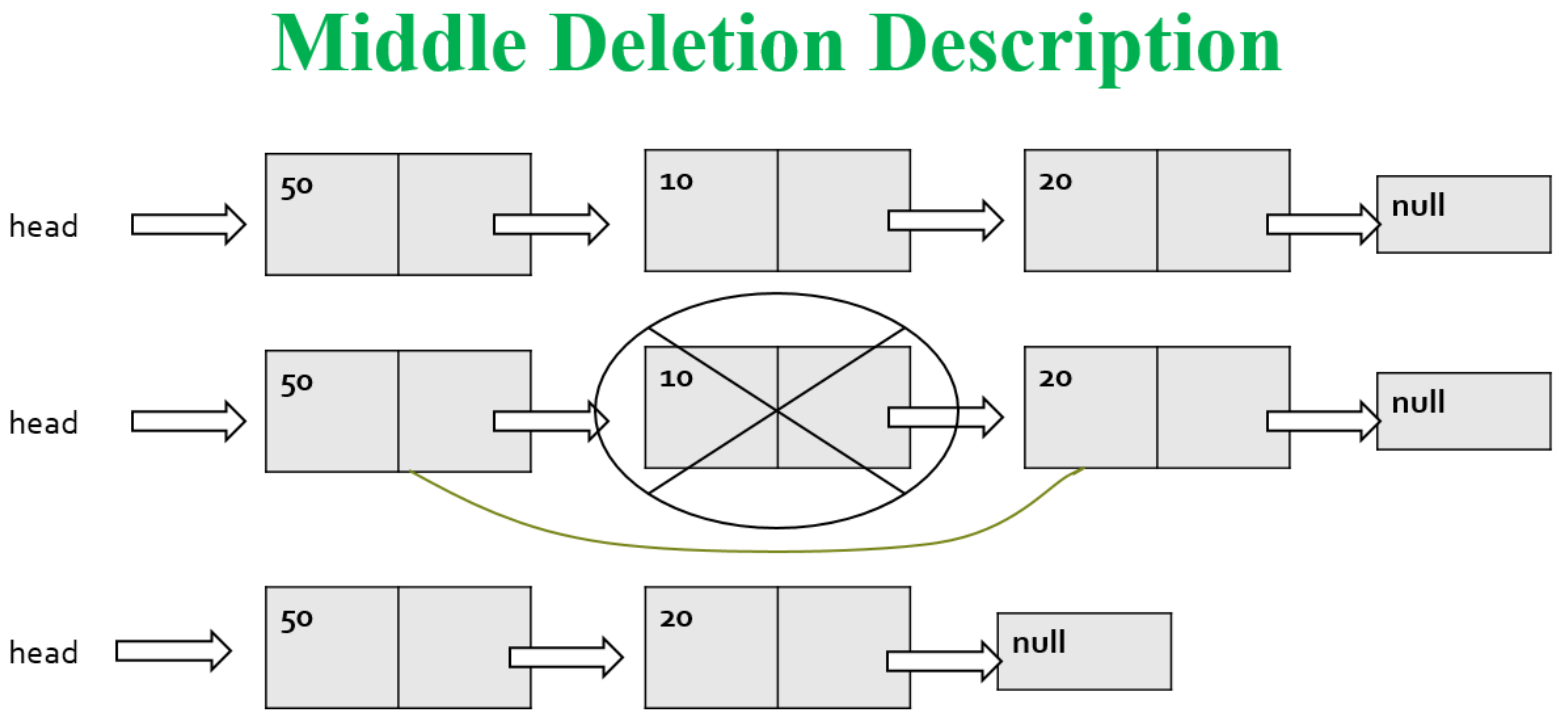
- Break the pointer connection
- Set previous node pointer to NULL
- Delete the node



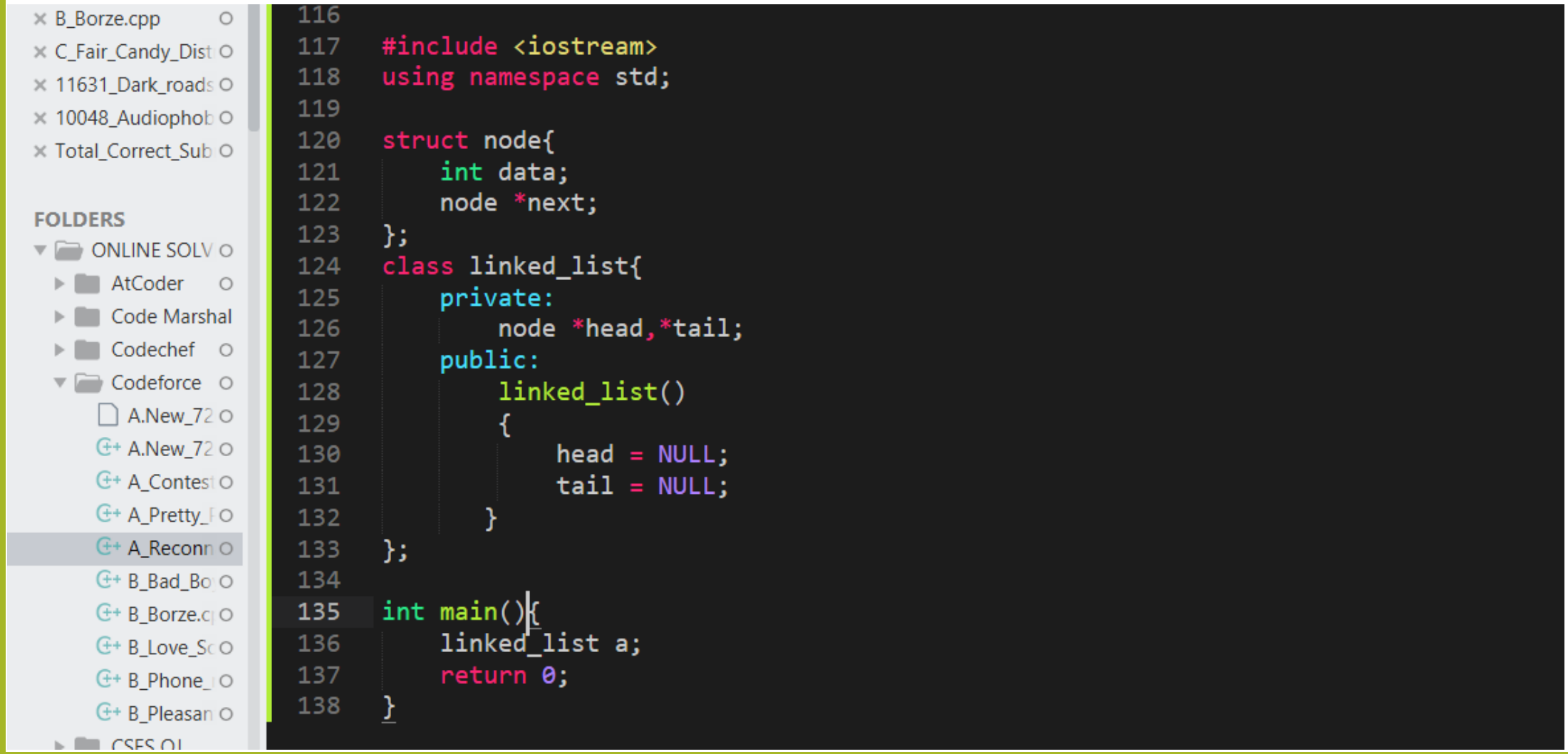
Deleting from the Middle

Steps:

- Set previous Node pointer to next node
- Break Node pointer connection
- Delete the node



Basic Node Implementation



Complexity

Operation \ List	LinkedList	ArrayList
get(int index)	O(n/4) average	O(1)
add(E element)	O(1)	O(1) amortized O(n) worst-case
add(int index, E element)	O(n/4) average O(1) if index is 0	O(n/2) average
remove	O(n/4) average	O(n/2) average
Iterator.remove()	O(1)	O(n/2) average
ListIterator.add(E element)	O(1)	O(n/2) average

Conclusion

- Linked list is similar to an array that it contains data that is best organized in a list fashion
- Its dynamic structure make it expandable or shrinkable execution.
- This dynamic quality make it appealing to use in certain situations where the static nature of arrays will be wasteful

Thank You