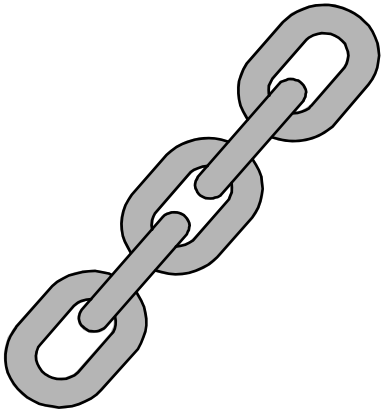
Data Structures and Algorithms (CSE 2101)

Week 03

Single Linked List



Objectives:

- **In this session, you will learn to:**
  - **Identify the features of linked lists**
  - **Implement a singly-linked list**

Linked List

- Suppose you have to write an algorithm to generate and store all prime numbers between 1 and 100,000 and display them.
- How will you solve this problem?

Linked List (Contd.)

- Consider the following algorithm, which uses an array to solve this problem:
  1. Set I = 0
  2. Repeat step 3 varying N from 2 to 100000
  3. If N is a prime number
     a. Set A[I] = N // If N is prime store it in an array
     b. I = I + 1
  4. Repeat step 5 varying J from 0 to I-1
  5. Display A[J] // Display the prime numbers stored in the array

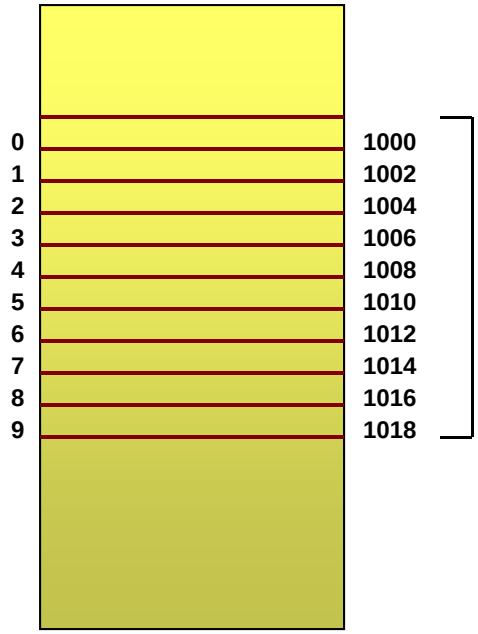
Linked List (Contd.)

- What is the problem in this algorithm?
  - The number of prime numbers between 1 and 100,000 is not known.
    - Since you are using an array to store the prime numbers, you need to declare an array of arbitrarily large size to store the prime numbers.
  - Disadvantages of this approach, suppose you declare an array of size N:
    - If the number of prime numbers between 1 and 100,000 is more than N then all the prime numbers cannot be stored.
    - If the number of prime numbers is much less than N, a lot of memory space is wasted.
- Thus, you cannot use an array to store a set of elements if you do not know the total number of elements in advance.
- How do you solve this problem?
  - By having some way in which you can allocate memory as and when it is required.

Dynamic Memory Allocation

- When you declare an array, a contiguous block of memory is allocated.
- Let us suppose you declare an array of size 10 to store first 10 prime numbers.
- If you know the address of the first element in the array, you can calculate the address of any other elements as shown:
  - Address of the first element + (size of the element × index of the element)
- When memory is allocated dynamically, a block of memory is assigned arbitrarily from any location in the memory.
- Therefore, unlike arrays, these blocks may not be contiguous and may be spread randomly in the memory.

One contiguous block of memory allocated for the array to store 10 elements.

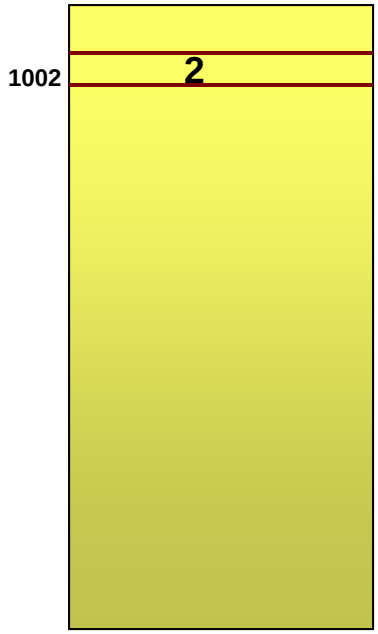


Memory representation

Dynamic Memory Allocation (Contd.)

- Let us see how this happens by allocating memory dynamically for the prime numbers.

Allocate memory for 2



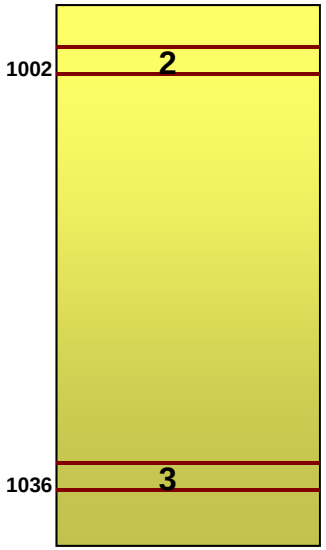
Memory allocated for 2

Memory representation

Dynamic Memory Allocation (Contd.)

- Let us see how this happens

Allocate memory for 3



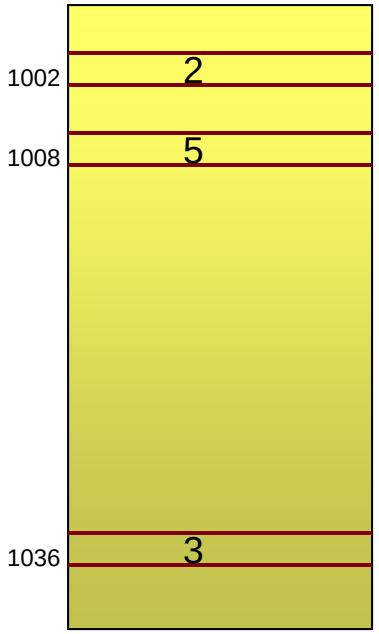
Memory allocated for 3

Memory representation

Dynamic Memory Allocation (Contd.)

- Let us see how this happens

Allocate memory for 5



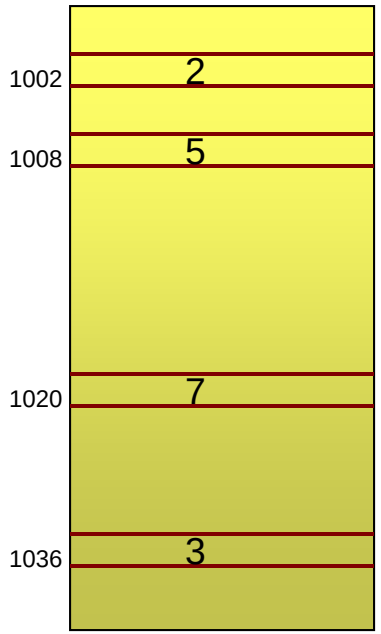
Memory representation

Memory allocated for 5

Dynamic Memory Allocation (Contd.)

- Let us see how this happens

Allocate memory for 7



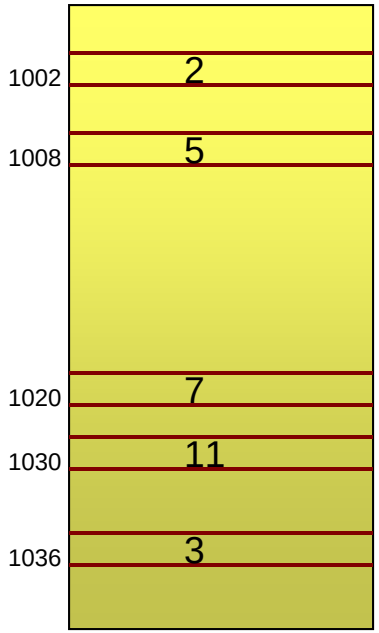
Memory representation

Memory allocated for 7

Dynamic Memory Allocation (Contd.)

- Let us see how this happens

Allocate memory for 11

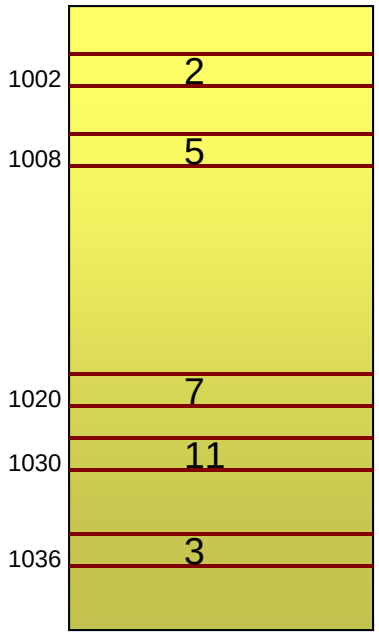


Memory representation

Memory allocated for 11

Dynamic Memory Allocation (Contd.)

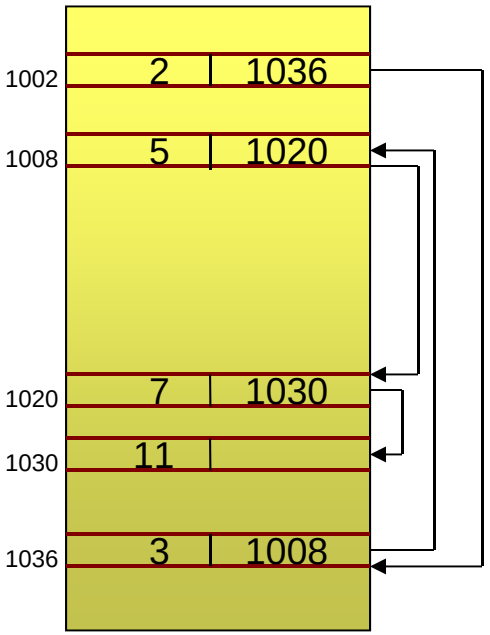
- To access any element, you need to know its address.
- Now, if you know the address of the first element, you cannot calculate the address of the rest of the elements.
- This is because, all the elements are spread at random locations in the memory.



Memory representation

Dynamic Memory Allocation (Contd.)

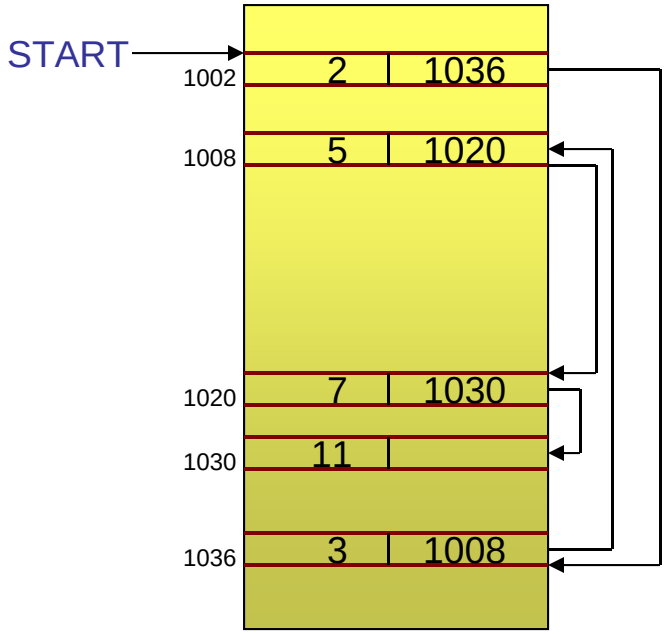
- Therefore, it would be good if every allocated block of memory contains the address of the next block in sequence.
- This gives the list a linked structure where each block is linked to the next block in sequence.



Memory representation

Dynamic Memory Allocation (Contd.)

- You can declare a variable, START, that stores the address of the first block.
- You can now begin at START and move through the list by following the links.

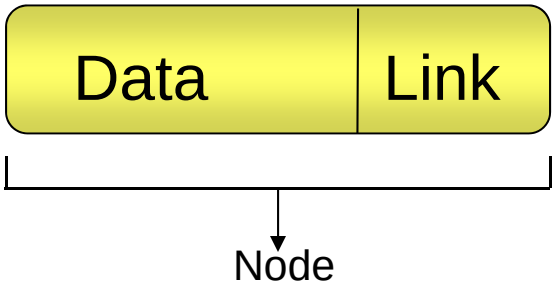


Memory representation

An example of a data structure that implements this concept is a linked list.

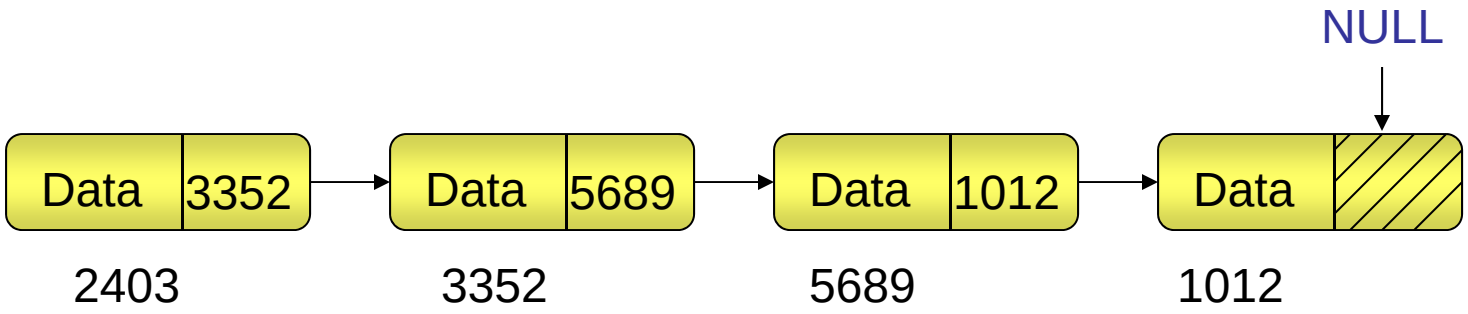
Defining Linked List

- Linked list:
  - Is a dynamic data structure.
  - Allows memory to be allocated as and when it is required.
  - Consists of a chain of elements, in which each element is referred to as a node.
- A node is the basic building block of a linked list.
- A node consists of two parts:
  - Data: Refers to the information held by the node
  - Link: Holds the address of the next node in the list



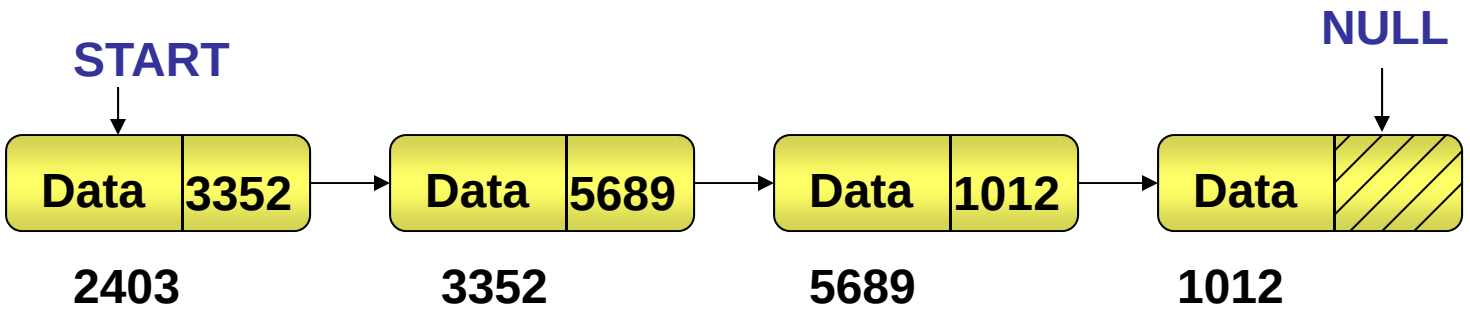
Defining Linked List

- All the nodes in a linked list are present at arbitrary memory locations.
- Therefore, every node in a linked list has link field that stores the address of the next node in sequence.
- The last node in a linked list does not point to any other node. Therefore, it points to NULL.



Defining Linked List

- To keep track of the first node, declare a variable/pointer, START, which always points to the first node.
- When the list is empty, START contains null.

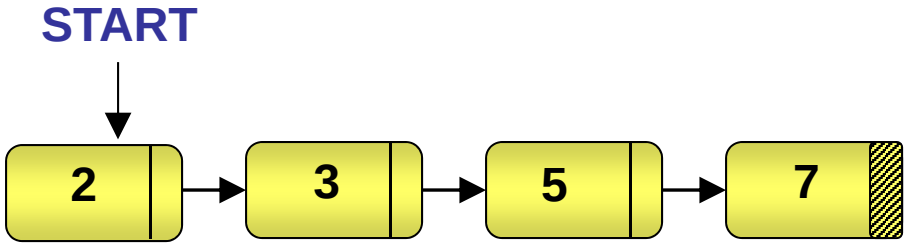


Implementing a Singly-Linked List

- Let us solve the given problem of storing prime numbers using a linked list.
  1. Repeat step 2 varying N from 2 to 100000.
  2. If N is a prime number, insert it in the linked list:
     - Allocate memory for a new node.
     - Store the prime number in the new node.
     - Attach the new node in the linked list.
  3. Display the prime numbers stored in the linked list.

Inserting a Node in a Singly-Linked List

- Consider the following linked list that stores prime numbers.



- When a new prime number is generated, it should be inserted at the end of the list.
- Initially, when the list is empty, START contains NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- Consider the given algorithm to insert a node in a linked list.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- Consider that the list is initially empty.
- START = NULL
- Insert a prime number 2.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- START = NULL
- Insert a prime number 2.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- START = NULL

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

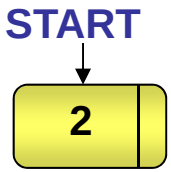
Inserting a Node in a Singly-Linked List (Contd.)

- START = NULL

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

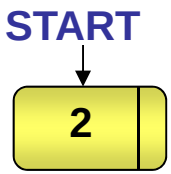
Inserting a Node in a Singly-Linked List (Contd.)

- START = NULL



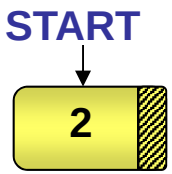
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

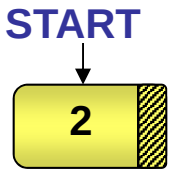


Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

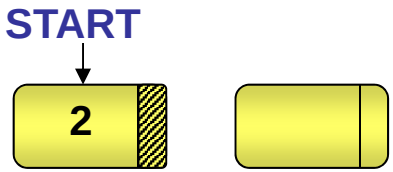
- Insert a prime number 3.



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

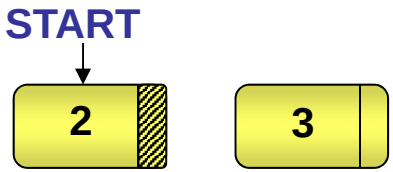
Inserting a Node in a Singly-Linked List (Contd.)

- Insert a prime number 3.



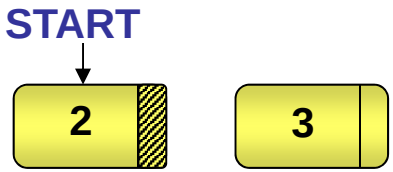
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



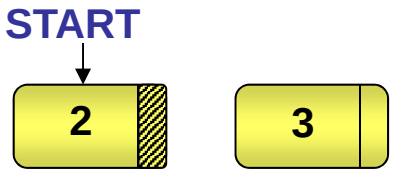
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



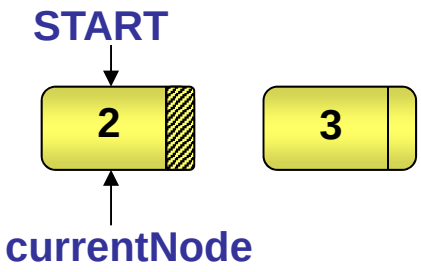
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



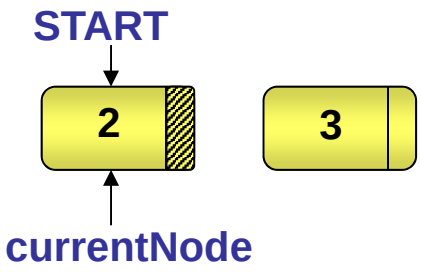
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



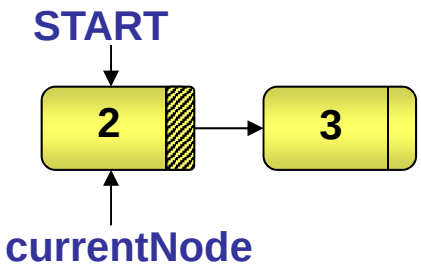
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



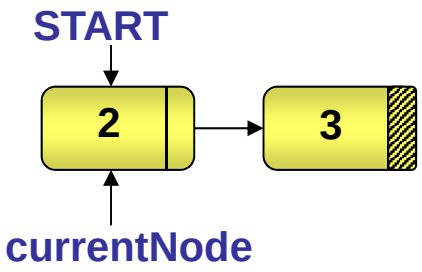
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

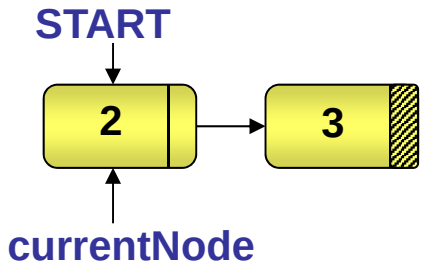


Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

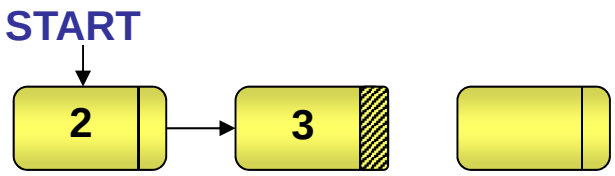
- Insert a prime number 5.



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

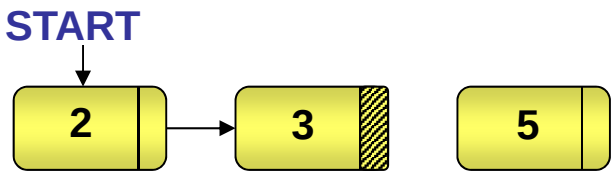
Inserting a Node in a Singly-Linked List (Contd.)

- Insert a prime number 5.



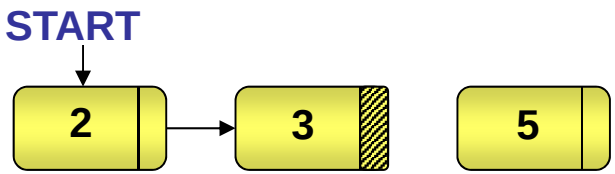
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



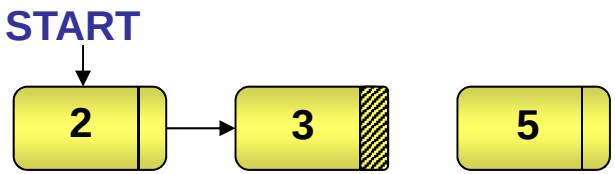
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



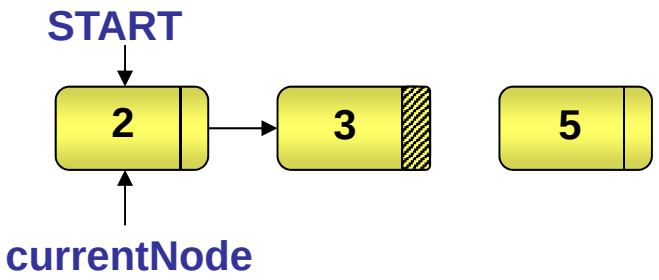
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



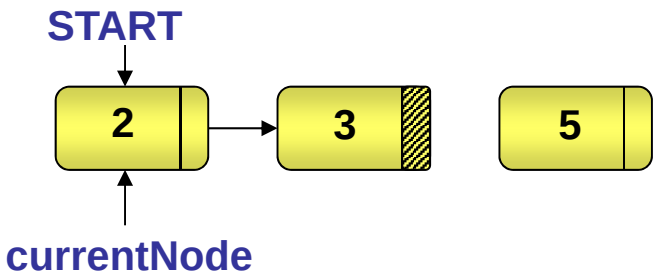
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



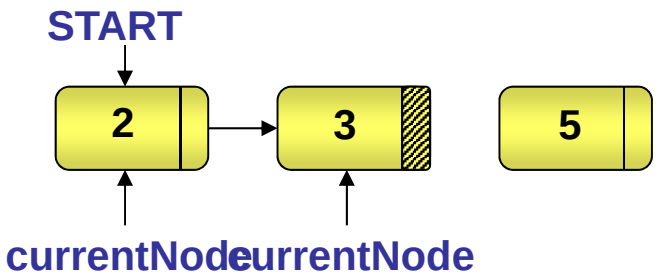
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



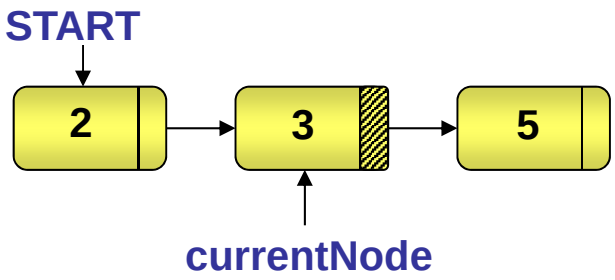
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



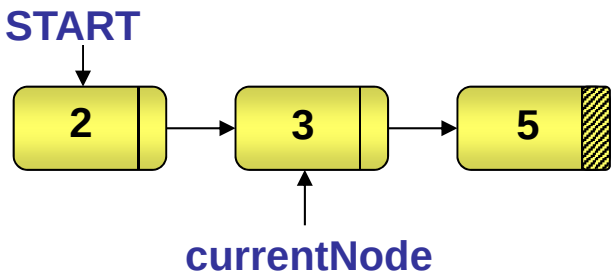
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- What is the problem with this algorithm?
  - Consider a list of 15000 numbers.
  - To insert a number at the end of the list, you first need to locate the last node in the list.
  - To reach the last node, you have to start from the first node and visit all the preceding nodes before you reach the last node.
  - This approach is very time consuming for lengthy lists.
- That is, a better approach is to maintain a variable, LAST, which will store the address of the last node in the list.
- Hence, you need to maintain two variables, START and LAST, to keep track of the first and last nodes in the list respectively.
- If the list is empty, START and LAST point to NULL.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then:
   a. Make START point to the new node.
   b. Go to step 6.
4. Locate the last node in the list, and mark it as currentNode. To locate the last node in the list, execute the following steps:
   a. Mark the first node as currentNode.
   b. Repeat step c until the successor of currentNode becomes NULL.
   c. Make currentNode point to the next node in sequence.
5. Make the next field of currentNode point to the new node.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- Refer to the given algorithm to insert a node at the end of the linked list.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

- Consider that the list is initially empty
  - START = NULL
  - LAST = NULL
- Insert a prime number 2.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

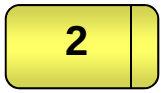
Inserting a Node in a Singly-Linked List (Contd.)

- START = NULL
- LAST = NULL
- Insert a prime number 2.

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

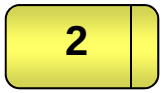
- START = NULL
- LAST = NULL



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

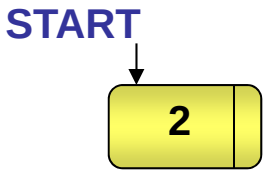
- START = NULL
- LAST = NULL



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

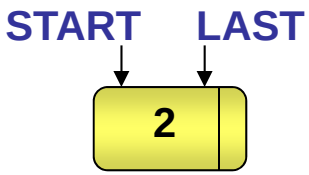
- START = NULL
- LAST = NULL



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

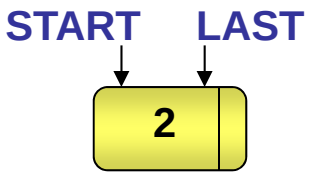
Inserting a Node in a Singly-Linked List (Contd.)

- LAST = NULL



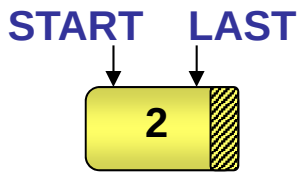
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

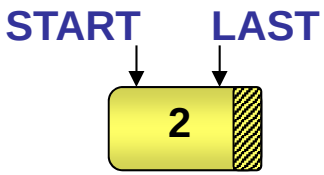


Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

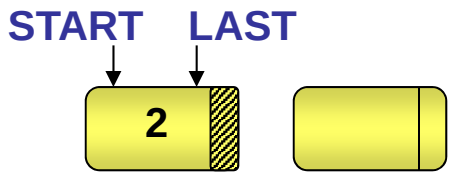
- Insert a prime number 3.



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

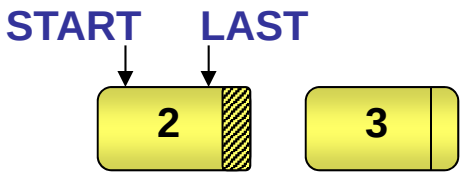
Inserting a Node in a Singly-Linked List (Contd.)

- Insert a prime number 3.



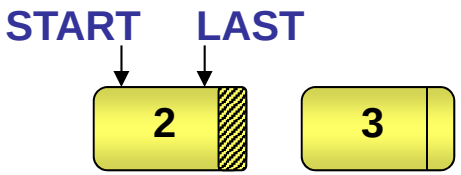
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



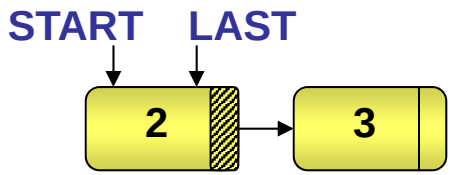
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



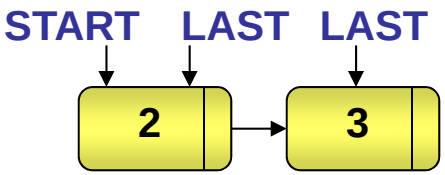
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



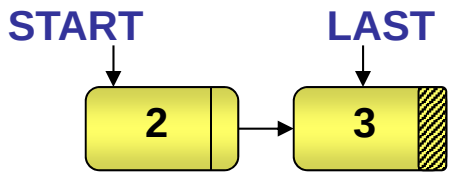
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

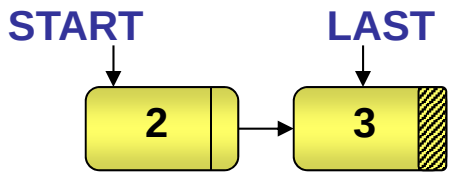


Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

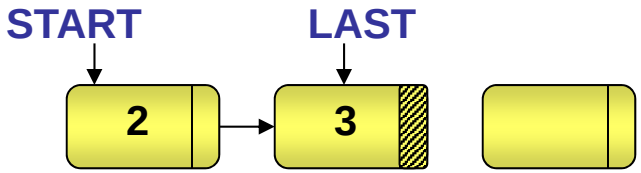
- Insert a prime number 5.



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

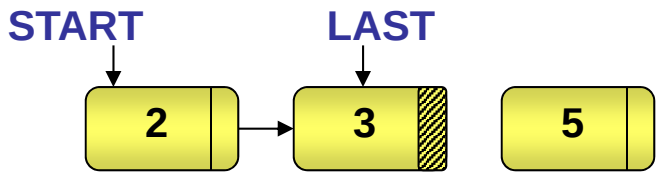
Inserting a Node in a Singly-Linked List (Contd.)

- Insert a prime number 5.



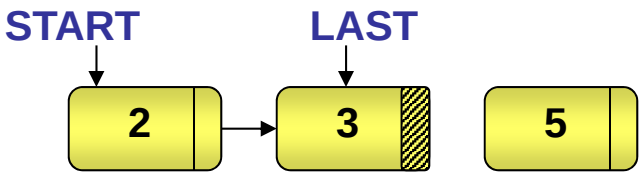
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



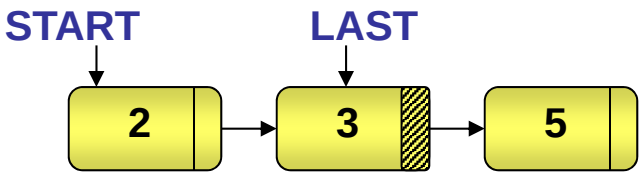
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



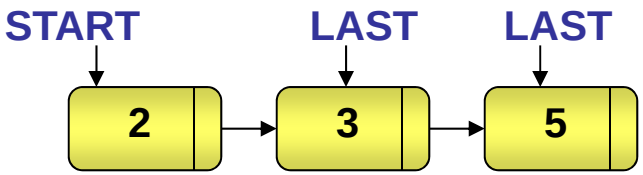
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



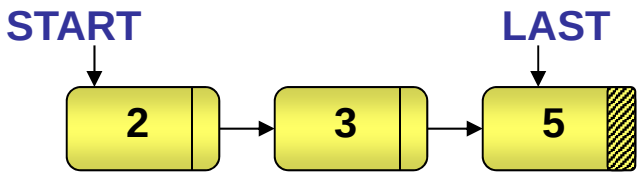
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)



Insertion complete

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. If START is NULL, then (If the list is empty):
   a. Make START point to the new node.
   b. Make LAST point to the new node.
   c. Go to step 6.
4. Make the next field of LAST point to the new node.
5. Mark the new node as LAST.
6. Make the next field of the new node point to NULL.

Inserting a Node in a Singly-Linked List (Contd.)

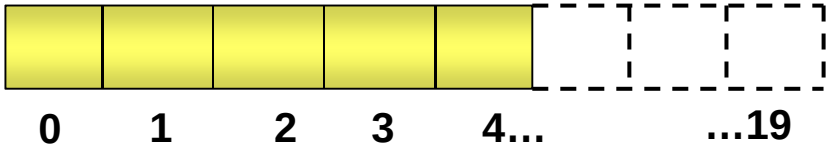
- Now consider another problem.
  - Suppose you have to store the marks of 20 students in the ascending order.
  - How will you solve this problem?

Inserting a Node in a Singly-Linked List (Contd.)

- Consider the following algorithm, which uses an array to solve this problem.
  1. Set N = 0 // Number of marks entered
  2. Repeat until N = 20
     a. Accept marks.
     b. Locate position I where the marks must be inserted.
     c. For J = N – 1 down to I
        Move A[J] to A[J+1] // Move elements to make
        // place for the new element
     d. Set A[I] = marks
     e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

- Let us perform some insert operations in the array by placing the elements at their appropriate positions to get a sorted list.

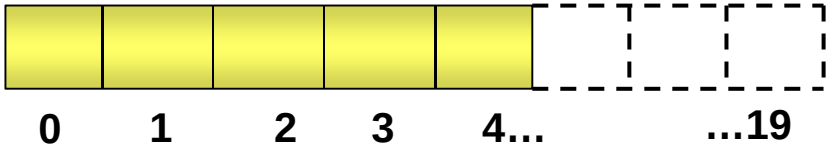


1. Set N = 0
2. Repeat until N = 20
   a. Accept marks.
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

- Let us perform some insert operations in the array by placing the elements at their appropriate positions to get a sorted list.

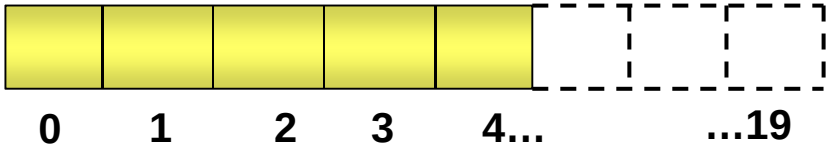
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1



N = 0

Inserting a Node in a Singly-Linked List (Contd.)

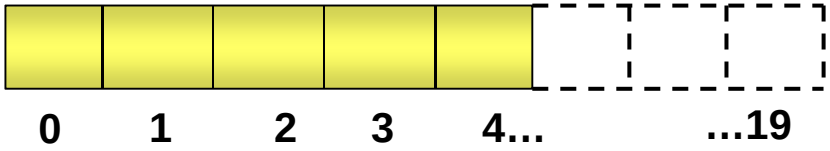
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1



N = 0

Inserting a Node in a Singly-Linked List (Contd.)

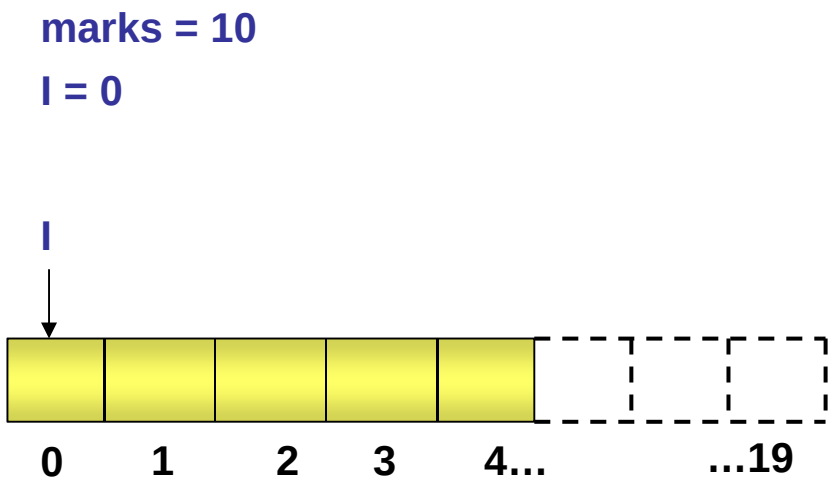
marks = 10



N = 0

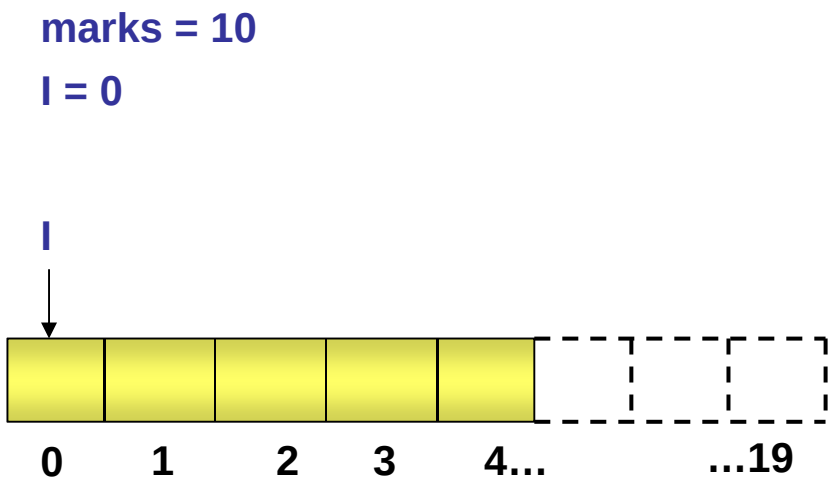
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



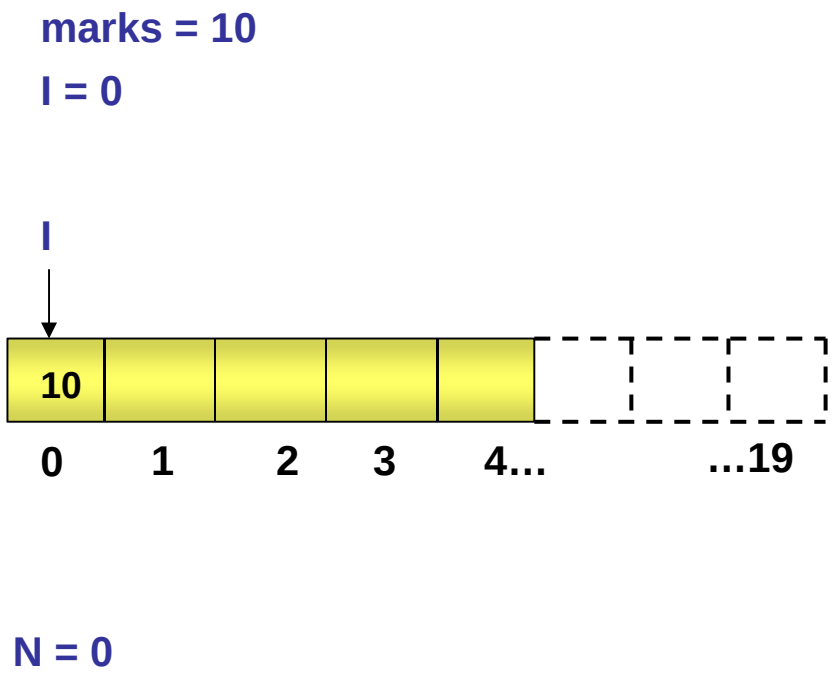
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



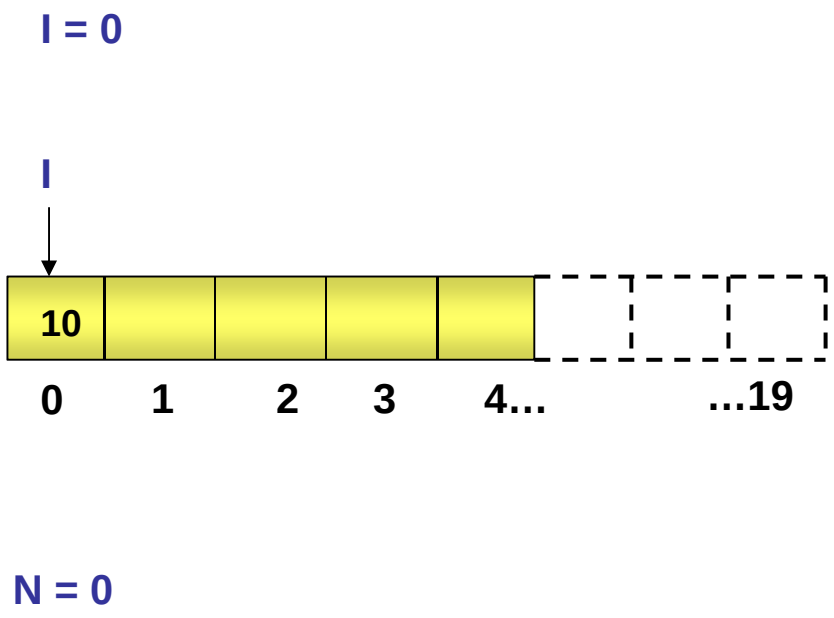
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



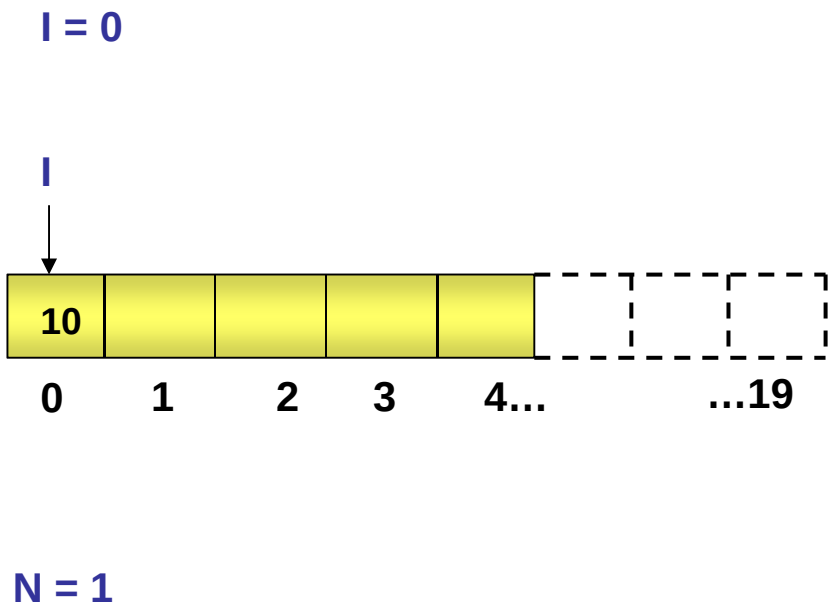
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

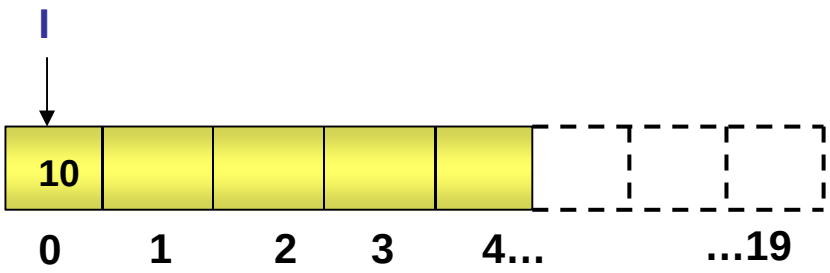


1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 20

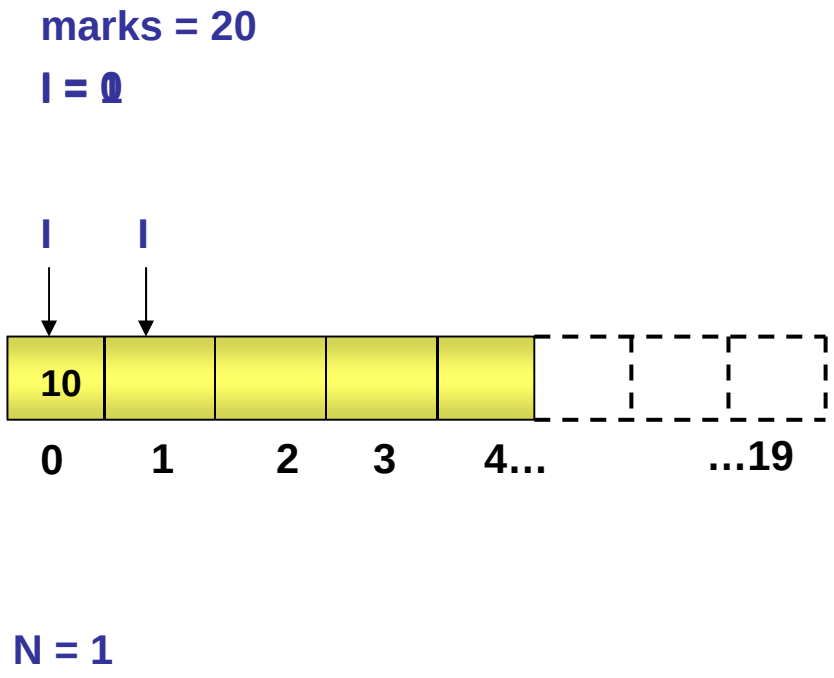
I = 0



N = 1

1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

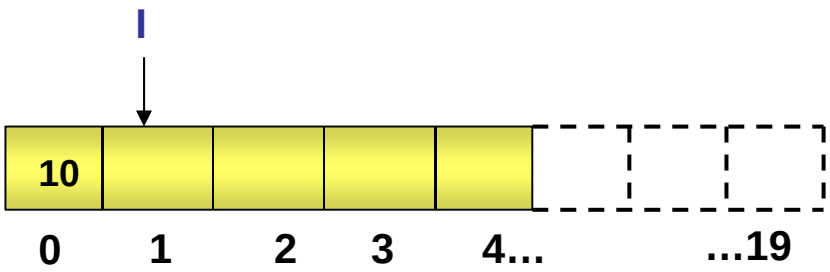


1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 20

I = 1



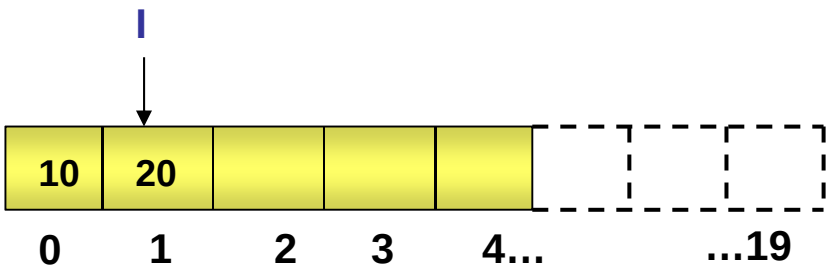
N = 1

1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 20

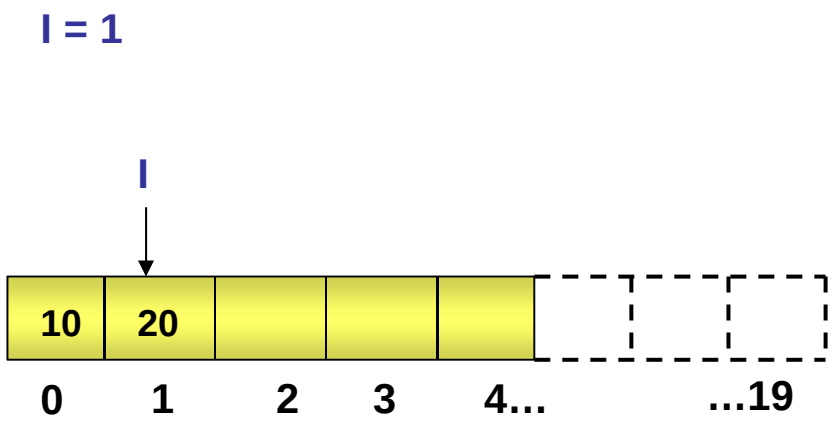
I = 1



N = 1

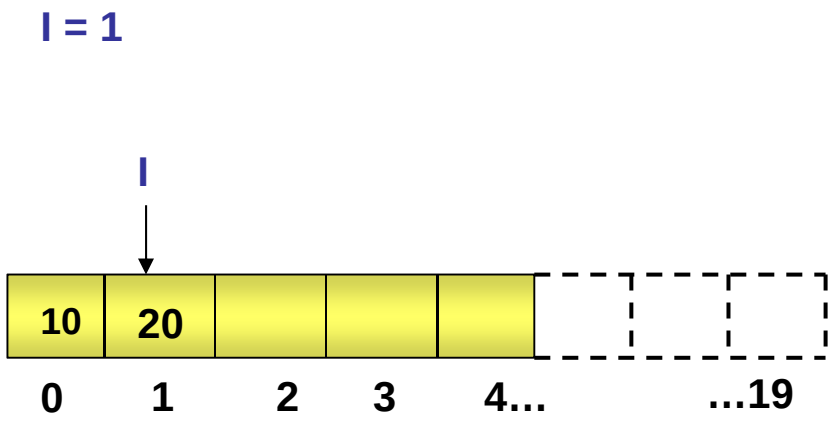
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

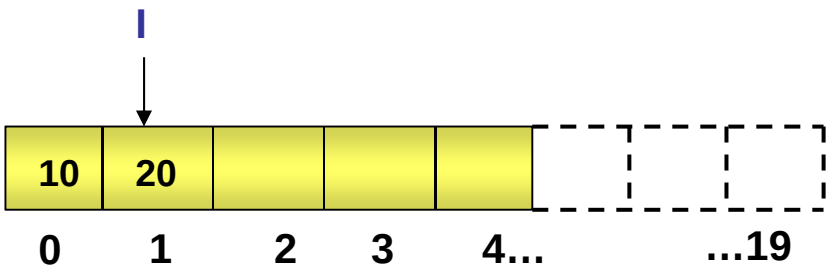


1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 17

I = 1



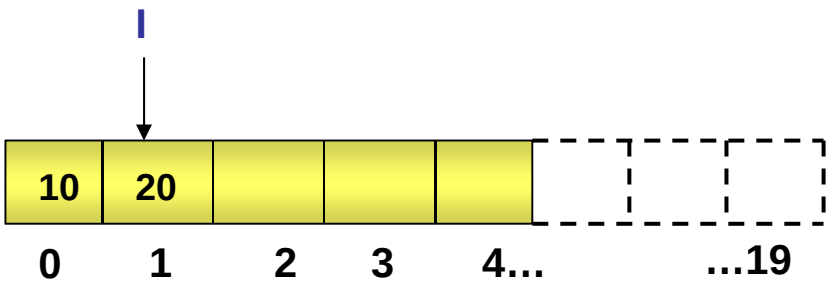
N = 2

1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 17

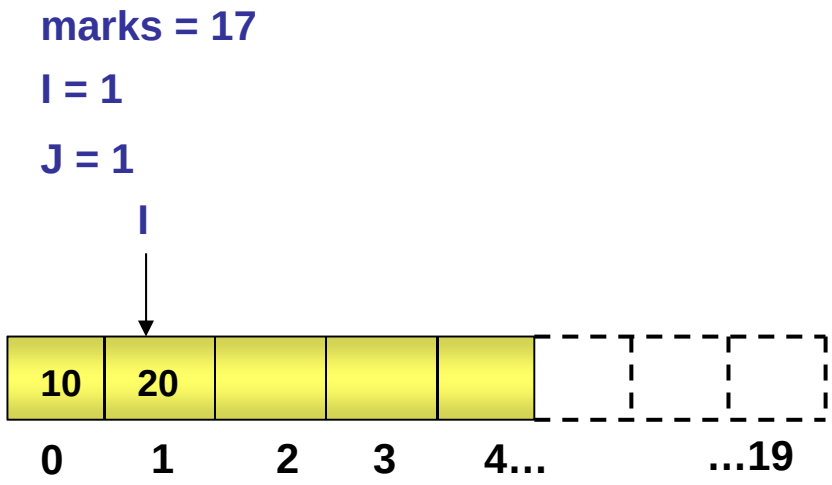
I = 1



N = 2

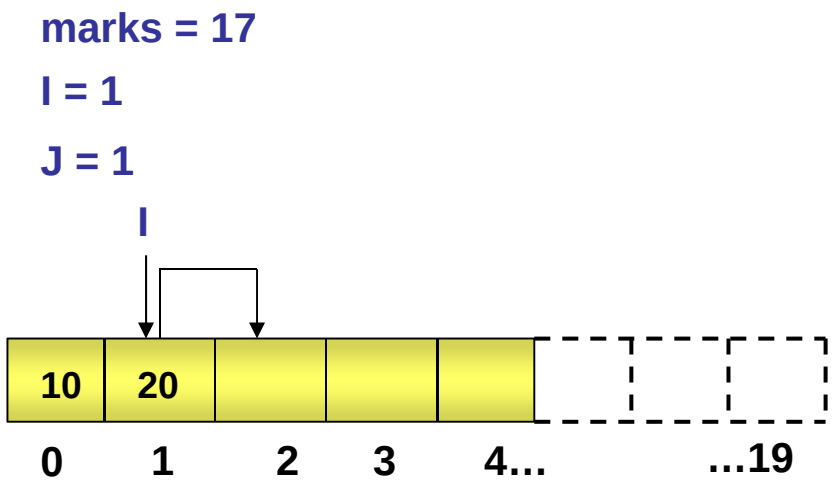
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



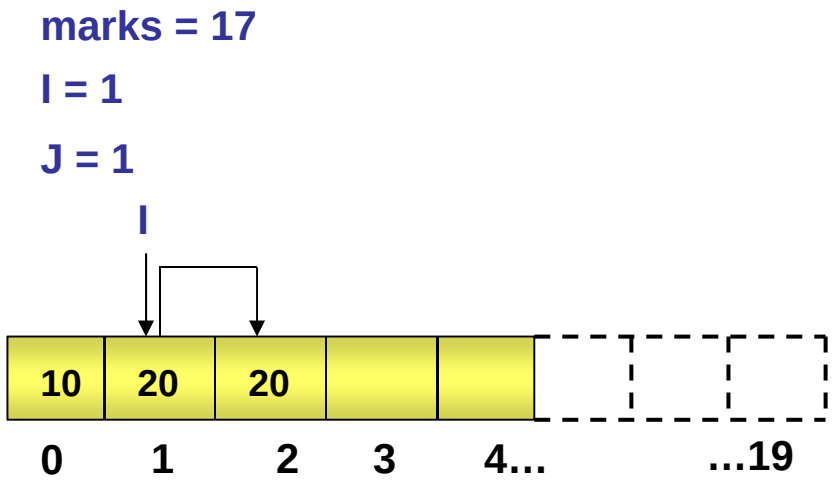
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



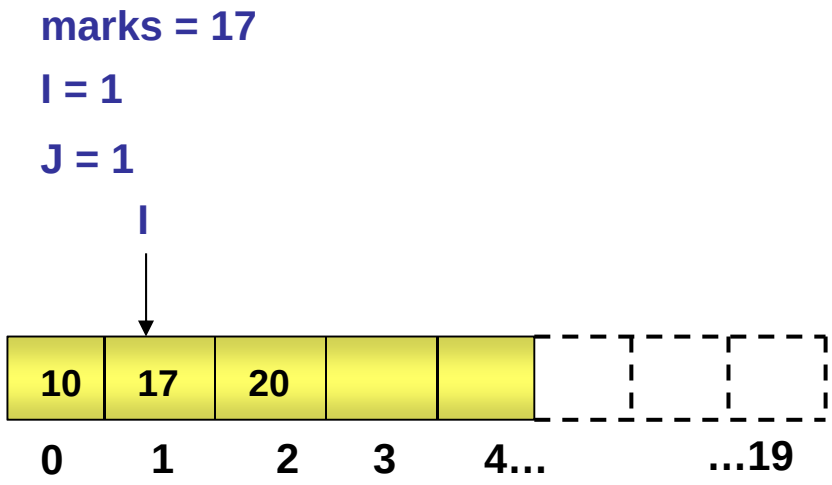
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



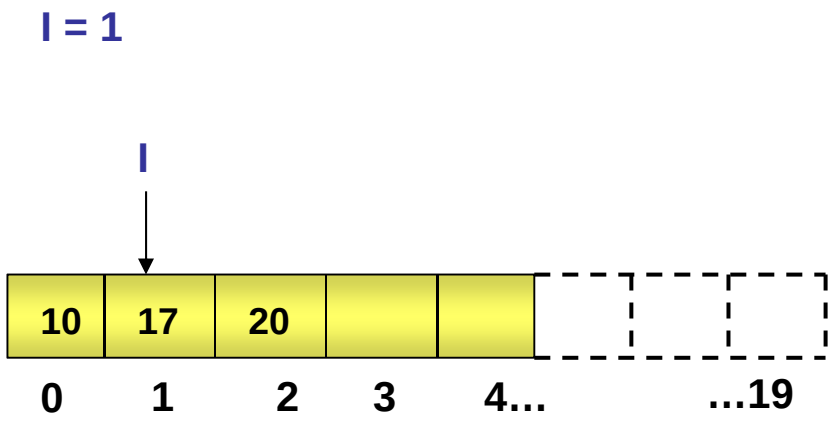
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

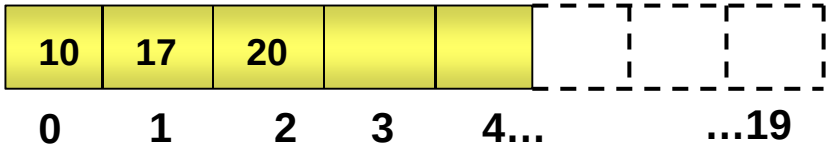
Inserting a Node in a Singly-Linked List (Contd.)



1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 15



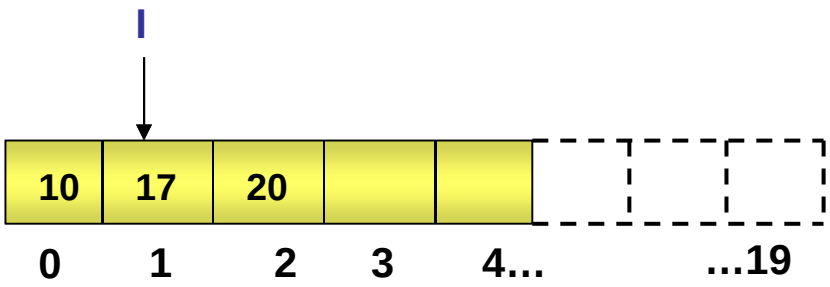
N = 3

1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)

marks = 15

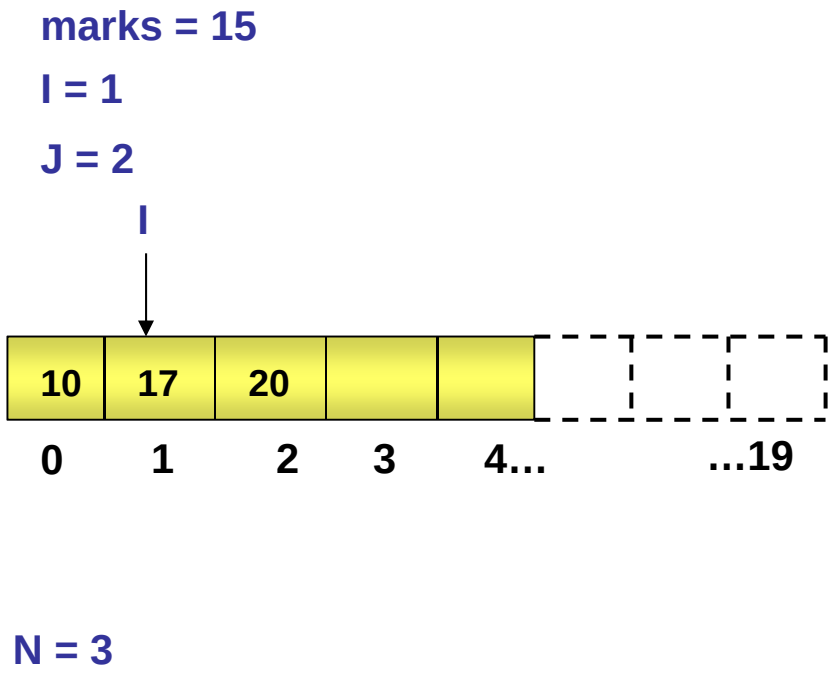
I = 1



N = 3

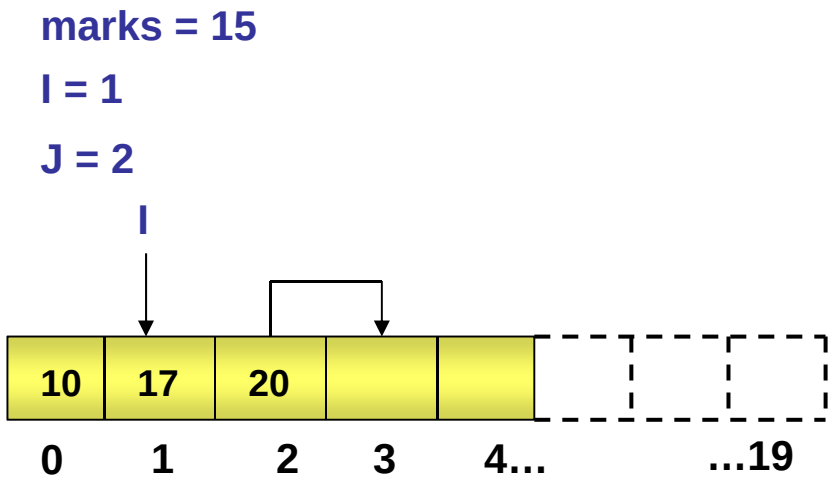
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



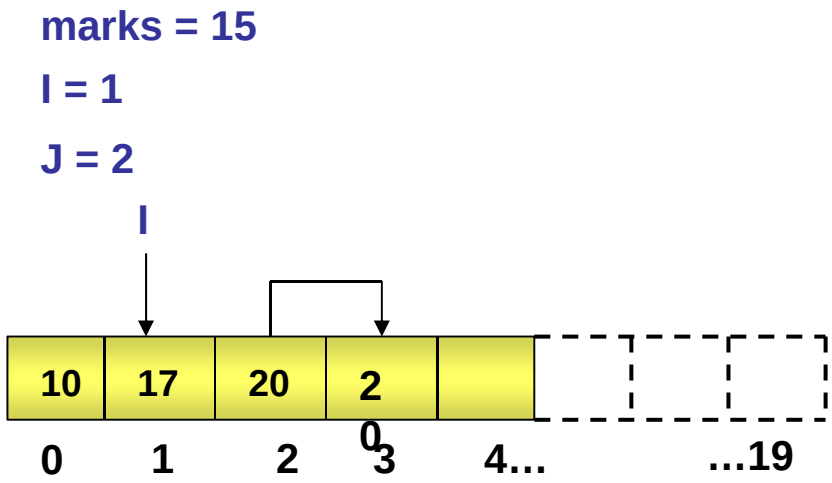
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



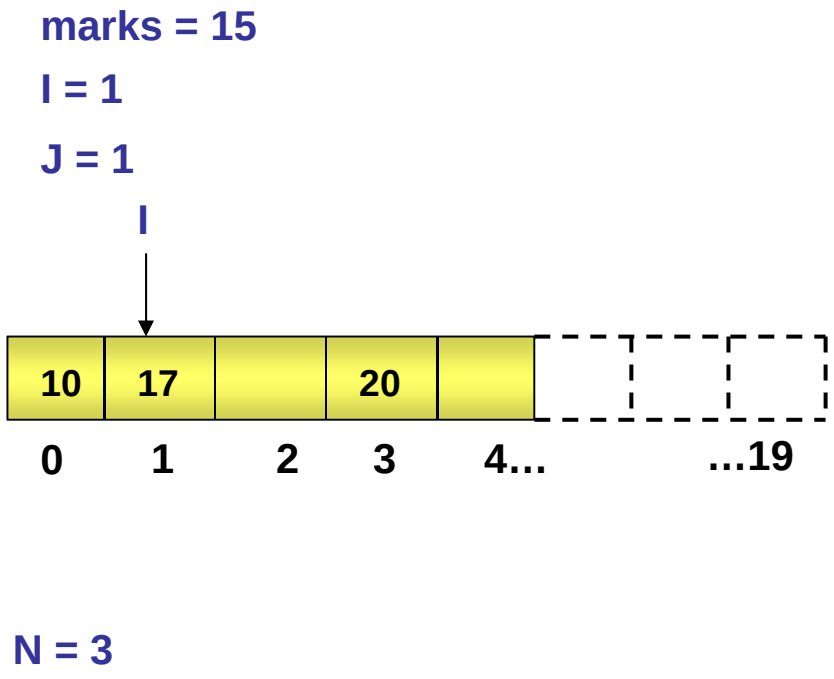
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



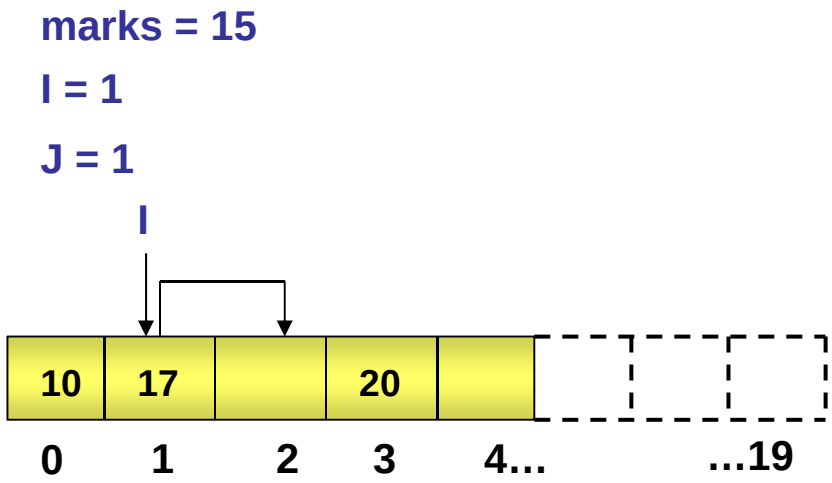
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



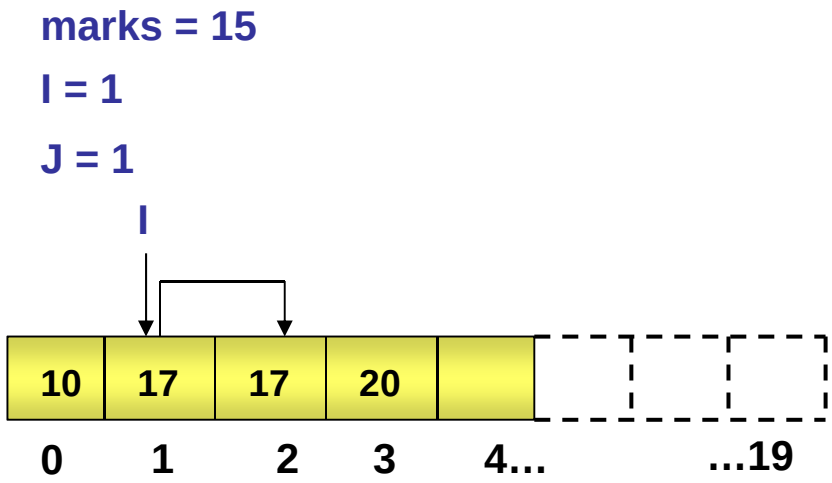
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



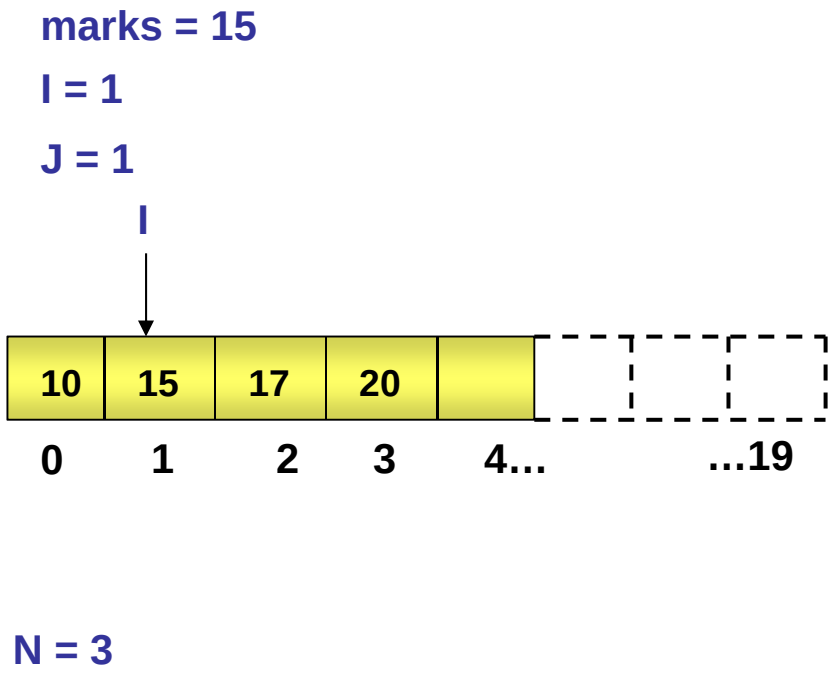
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



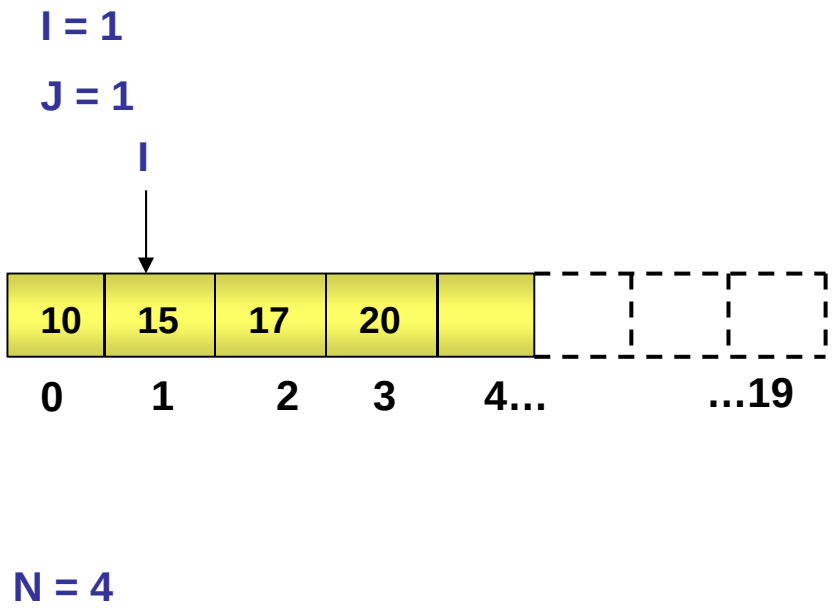
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

Inserting a Node in a Singly-Linked List (Contd.)



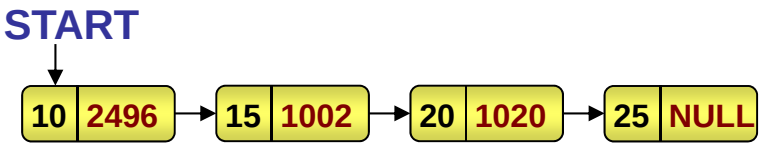
1. Set N = 0
2. Repeat until N = 20
   a. Accept marks
   b. Locate position I where the marks must be inserted
   c. For J = N-1 down to I
      Move A[J] to A[J+1]
   d. Set A[I] = marks
   e. N = N + 1

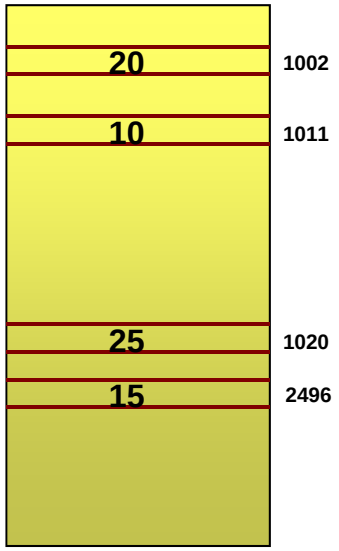
Inserting a Node in a Singly-Linked List (Contd.)

- What is the problem in this algorithm?
  - To insert an element at any position other than the end of the list, there is an additional overhead of shifting all succeeding elements one position forward.
  - Similarly, to delete an element at any position other than the end of the list, you need to shift the succeeding elements one position backwards.
  - When the list is very large, this would be very time consuming.
- From the preceding example we conclude that insertion and deletion at any position other than the end of an array is complex and inefficient.
- How can you overcome this limitation?
  - By **using a data structure that does not require shifting of data elements** after every insert / delete operation.
  - **A linked list offers this flexibility**.

Inserting a Node in a Singly-Linked List (Contd.)

- Consider a linked list that stores the marks of students in an ascending order.
- Insert marks (17).
- 17 should be inserted after 15.

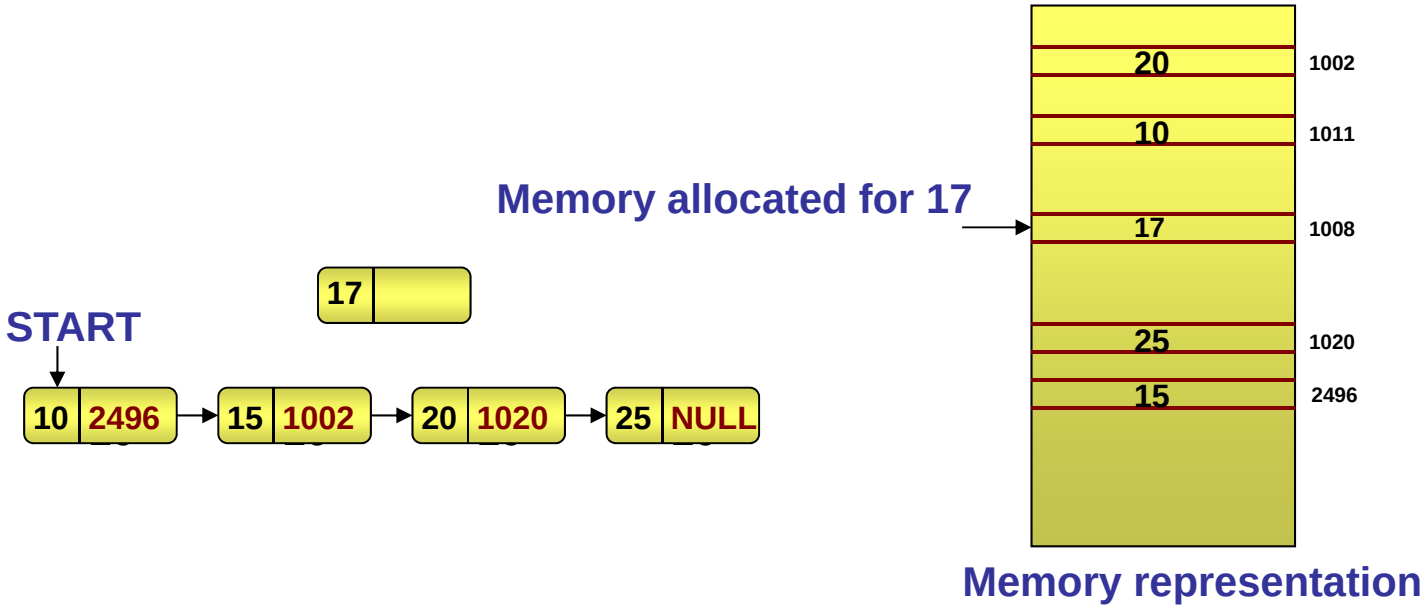




Memory representation

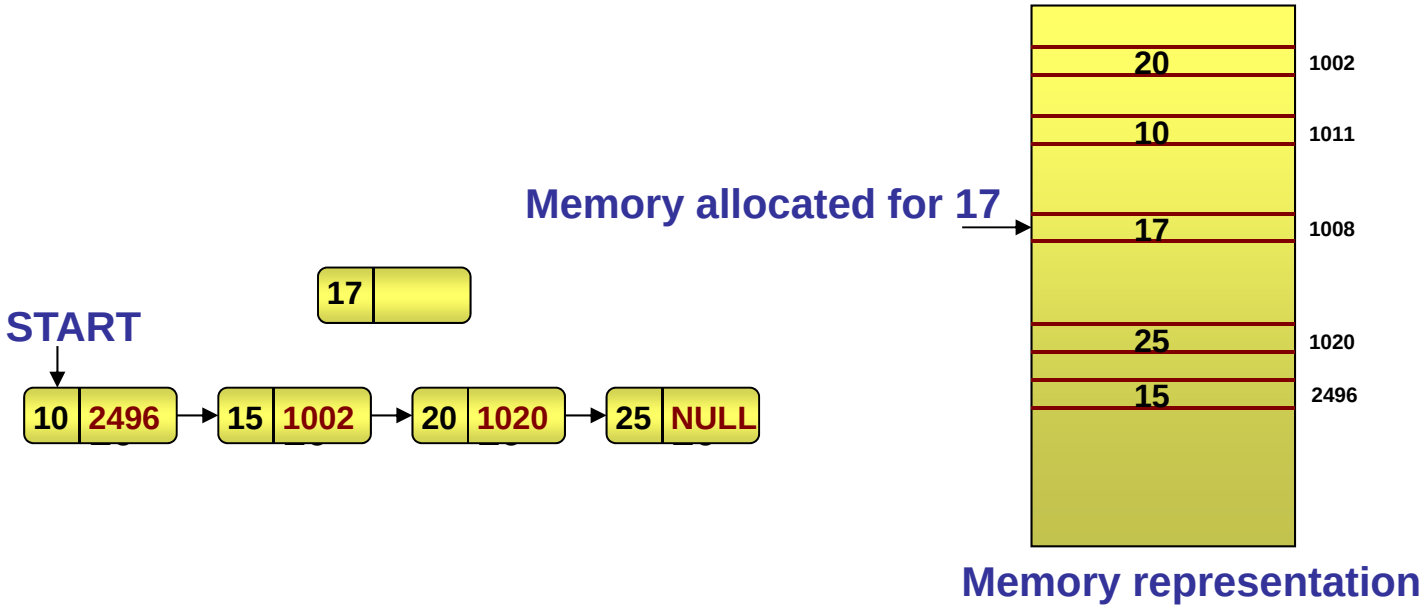
Inserting a Node in a Singly-Linked List (Contd.)

- Allocate memory for 17.



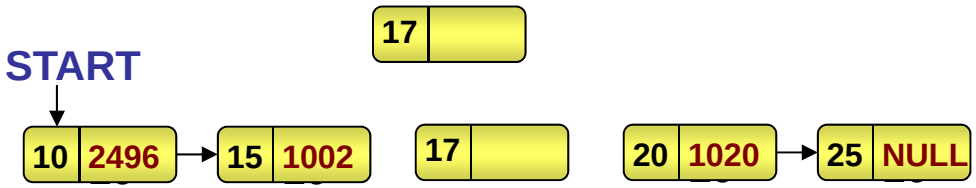
Inserting a Node in a Singly-Linked List (Contd.)

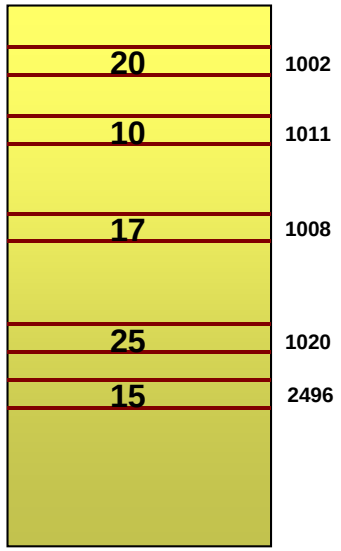
- In the given list, node 15 contains the address of node 20.
- To add node 17 after node 15, you need to update the address field of node 15 so that it now contains the address of node 17.



Inserting a Node in a Singly-Linked List (Contd.)

- Update the address field of node 15 to store the address of node 17.

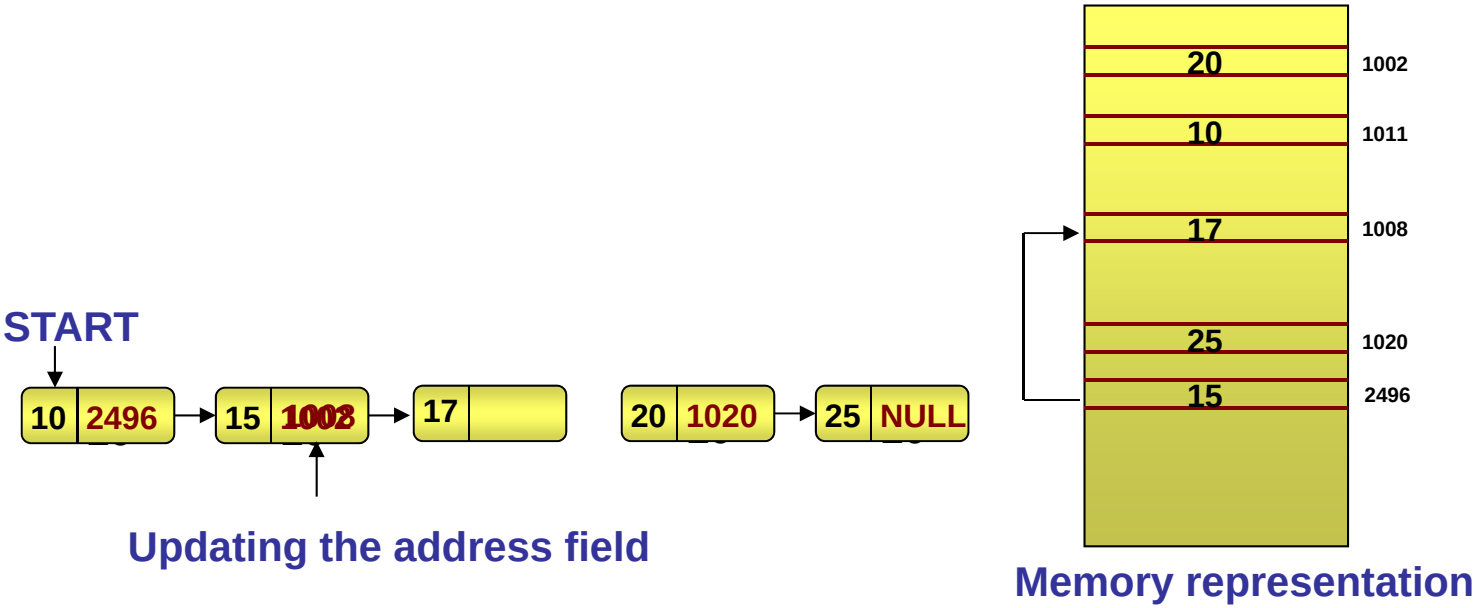




Memory representation

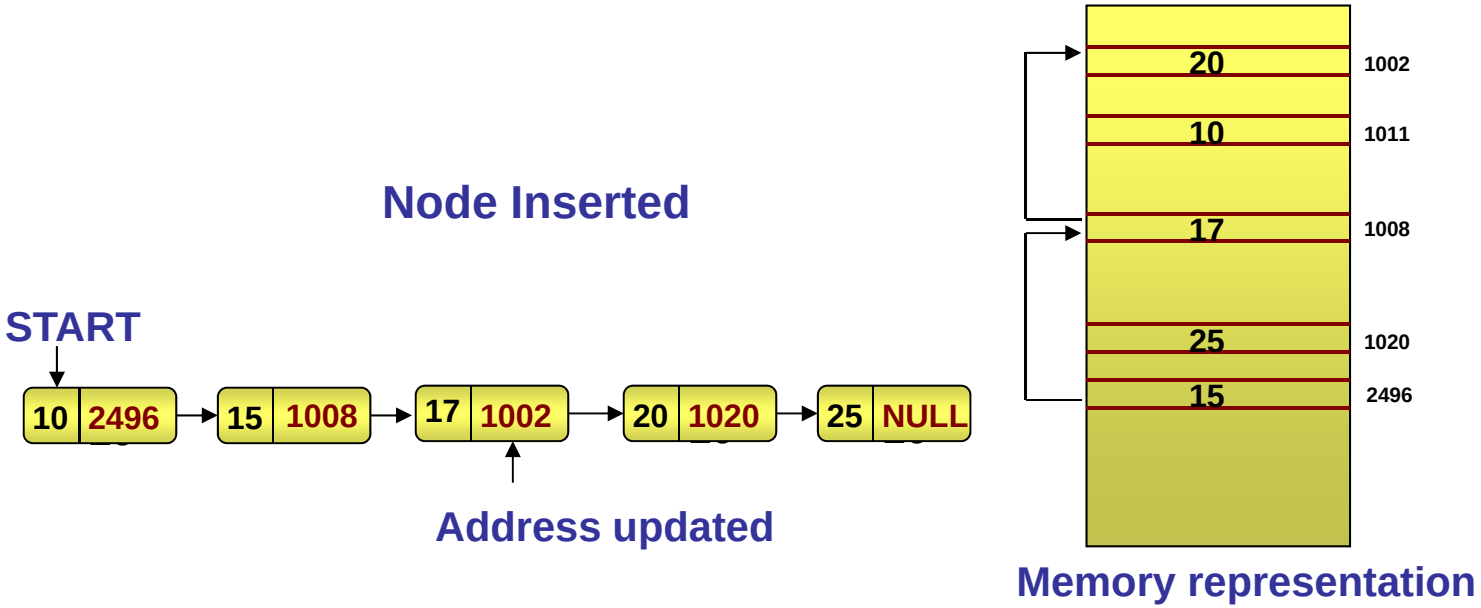
Inserting a Node in a Singly-Linked List (Contd.)

- Update the address field of node 15 to store the address of node 17.



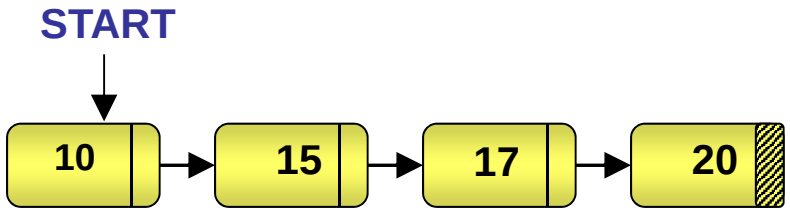
Inserting a Node in a Singly-Linked List (Contd.)

- Store the address of node 20 in the address field of node 17 to make a complete list.



Inserting a Node in a Singly-Linked List (Contd.)

- Now, let us solve the given problem using a linked list.
- Suppose the linked list currently contains the following elements.



- The linked list needs to be created in the ascending order of values.
- Therefore, the position of a new node in the list will be determined by the value contained in the new node.

Inserting a Node in a Singly-Linked List (Contd.)

- Write an algorithm to locate the position of the new node to be inserted in a linked list, where the nodes need to be stored in the increasing order of the values contained in them.

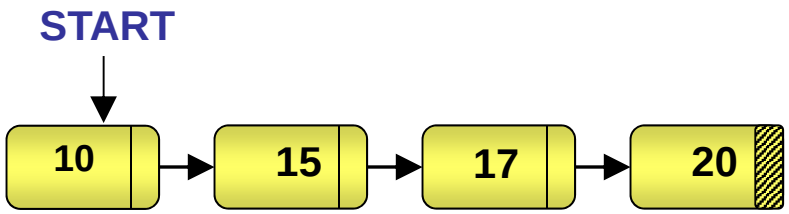
Inserting a Node in a Singly-Linked List (Contd.)

- Algorithm for locating the position of a new node in the linked list.
- After executing this algorithm the variables/pointers previous and current will be placed on the nodes between which the new node is to be inserted.

1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)

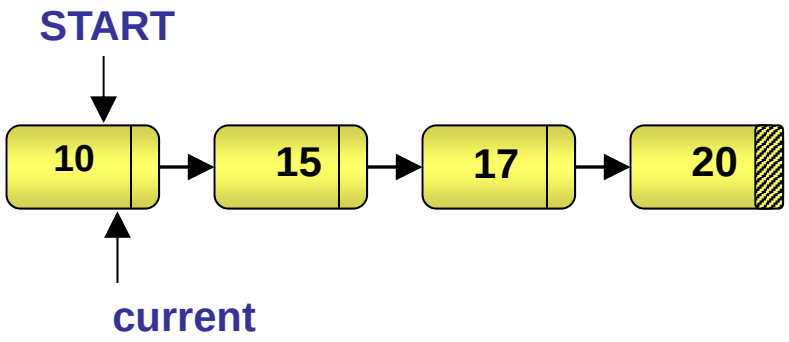
- Insert 16 in the given list.



1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

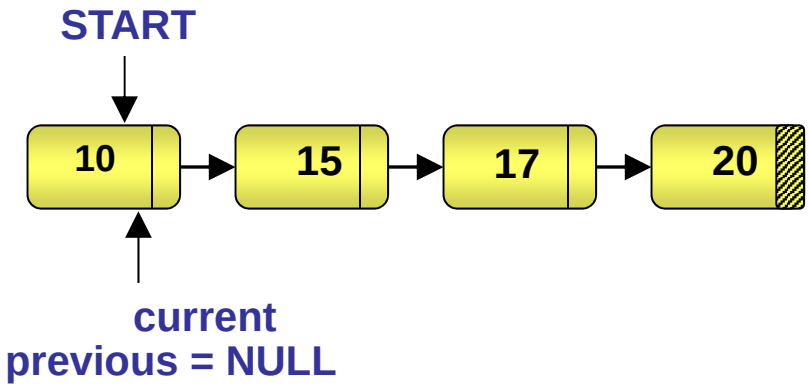
Inserting a Node in a Singly-Linked List (Contd.)

- Insert 16 in the given list.



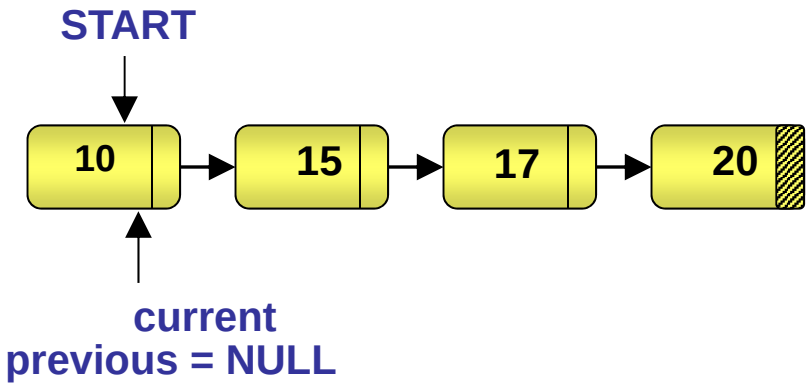
1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)



1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

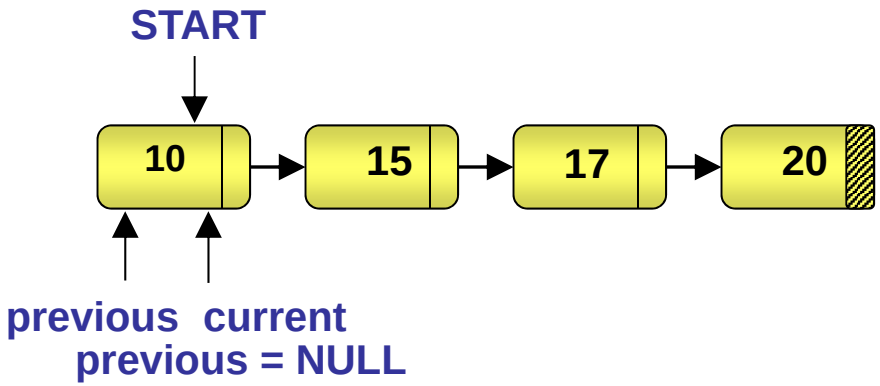
Inserting a Node in a Singly-Linked List (Contd.)



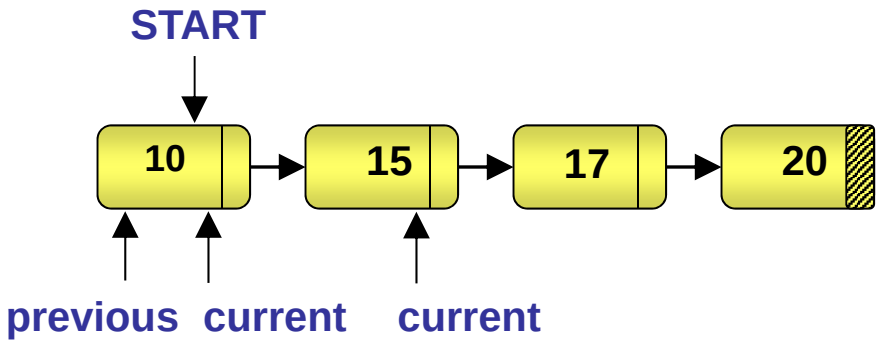
1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)

1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

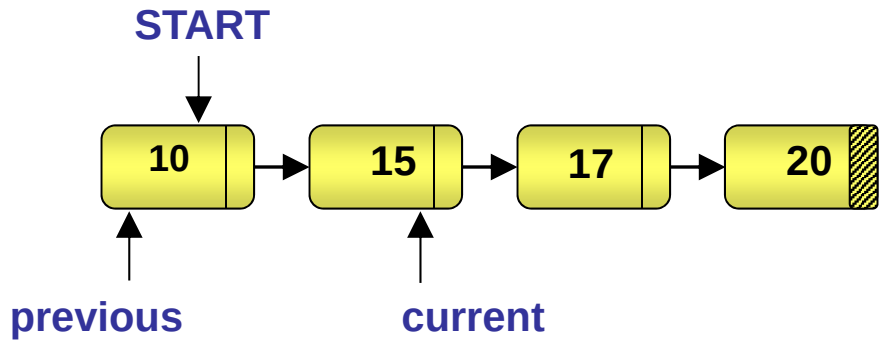


Inserting a Node in a Singly-Linked List (Contd.)



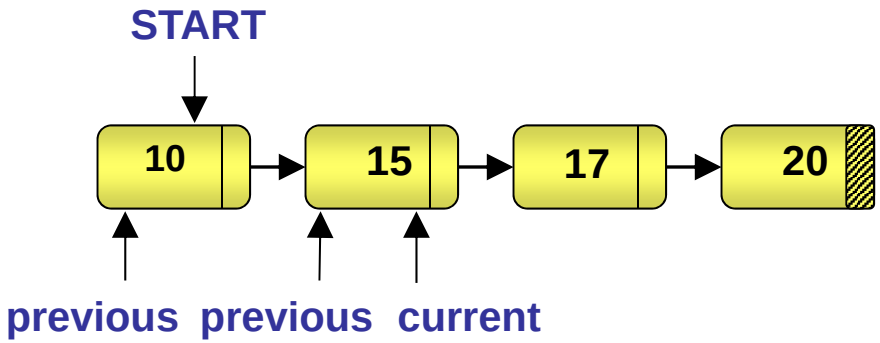
1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)



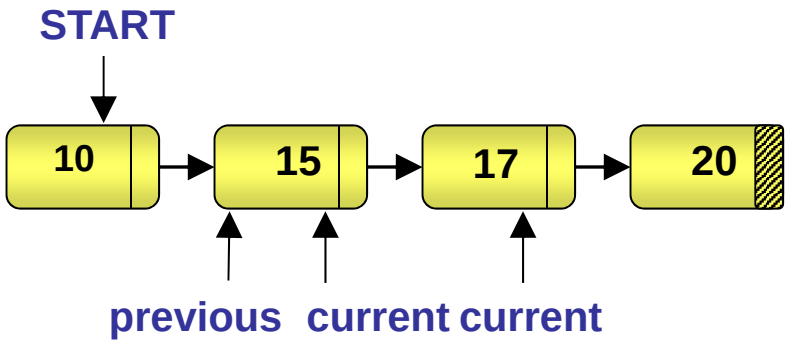
1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)



1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

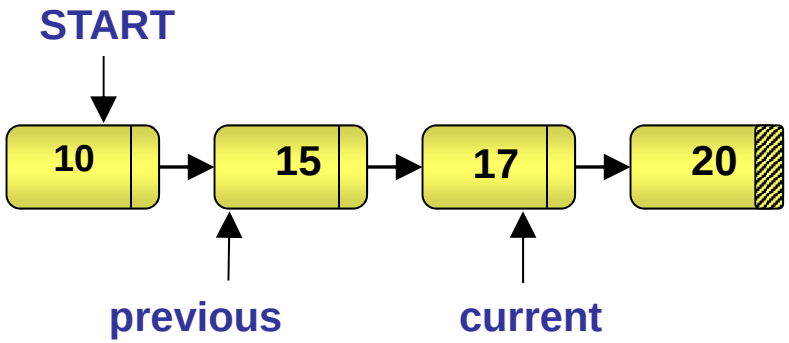
Inserting a Node in a Singly-Linked List (Contd.)



1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.

Inserting a Node in a Singly-Linked List (Contd.)

1. Make current point to the first node.
2. Make previous point to NULL.
3. Repeat step 4 and step 5 until current.info becomes greater than the newnode.info or current becomes equal to NULL.
4. Make previous point to current.
5. Make current point to the next node in sequence.



Node located

Inserting a Node in a Singly-Linked List (Contd.)

- Once the position of the new node has been determined, the new node can be inserted in the linked list.
- A new node can be inserted at any of the following positions in the list:
  - Beginning of the list
  - End of the list
  - Between two nodes in the list

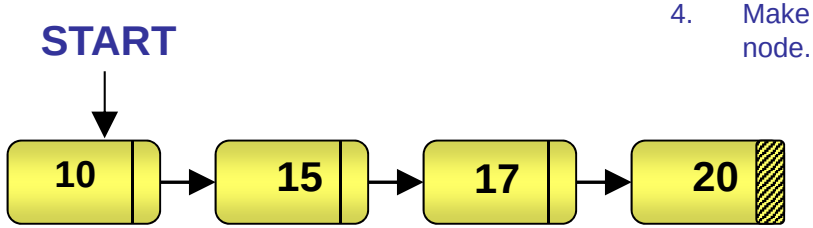
Inserting a Node in a Singly-Linked List (Contd.)

- Write an algorithm to insert a node in the beginning of a linked list.

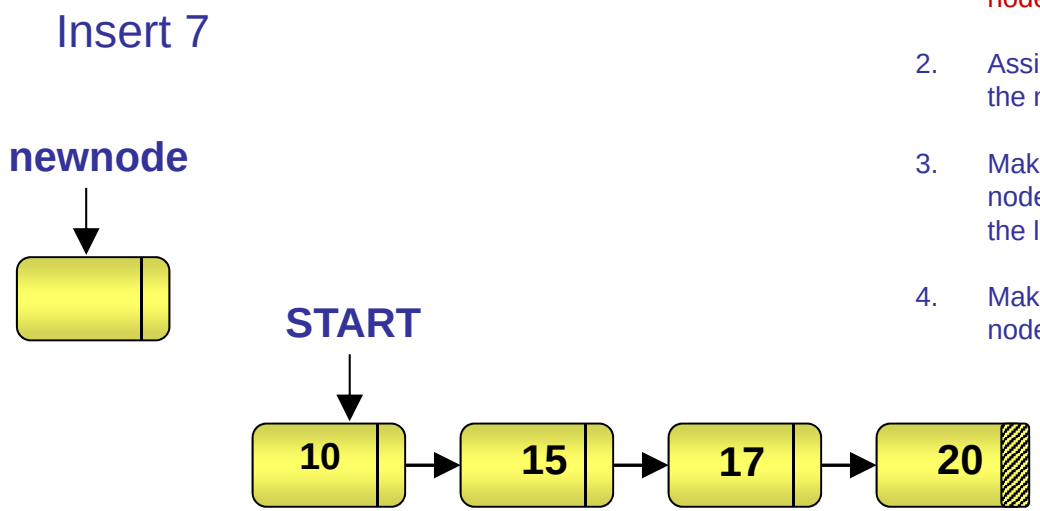
Inserting a Node in a Singly-Linked List (Contd.)

- Algorithm to insert a node in the beginning of a linked list

1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. Make the next field of the new node point to the first node in the list.
4. Make START, point to the new node.

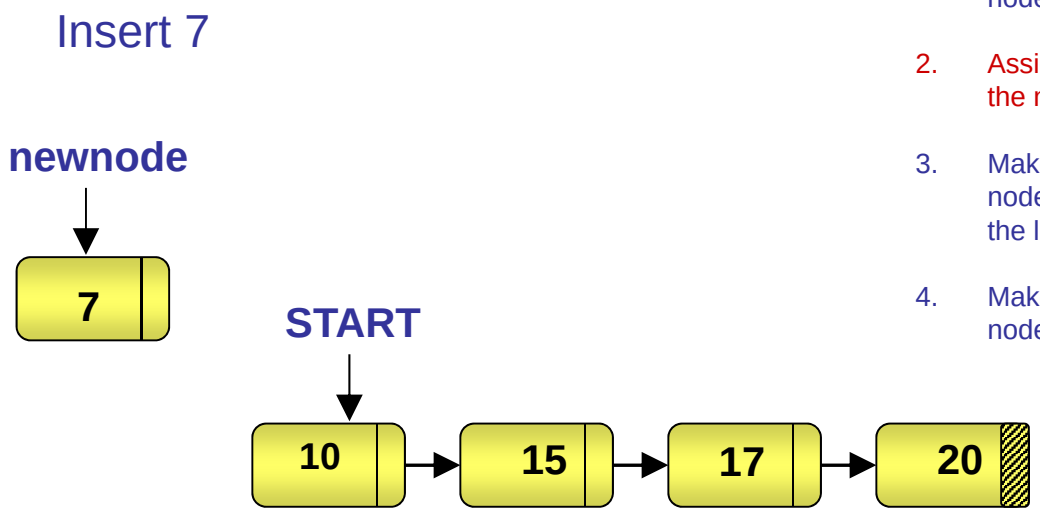


Inserting a Node in a Singly-Linked List (Contd.)



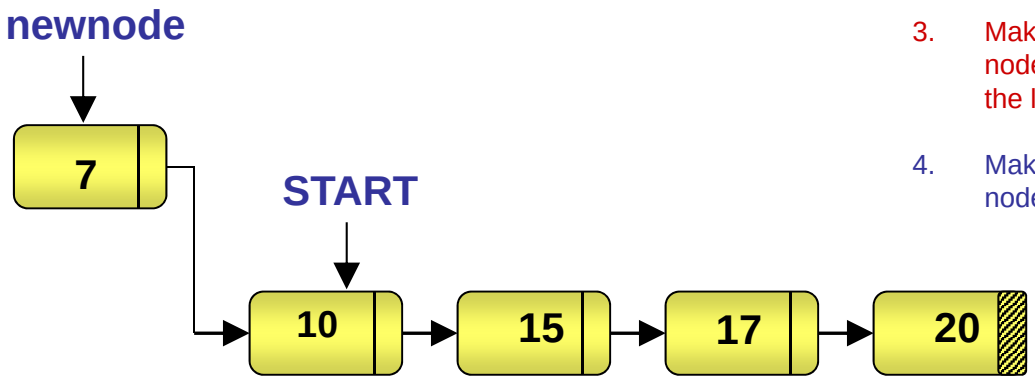
1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. Make the next field of the new node point to the first node in the list.
4. Make START, point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. Make the next field of the new node point to the first node in the list.
4. Make START, point to the new node.

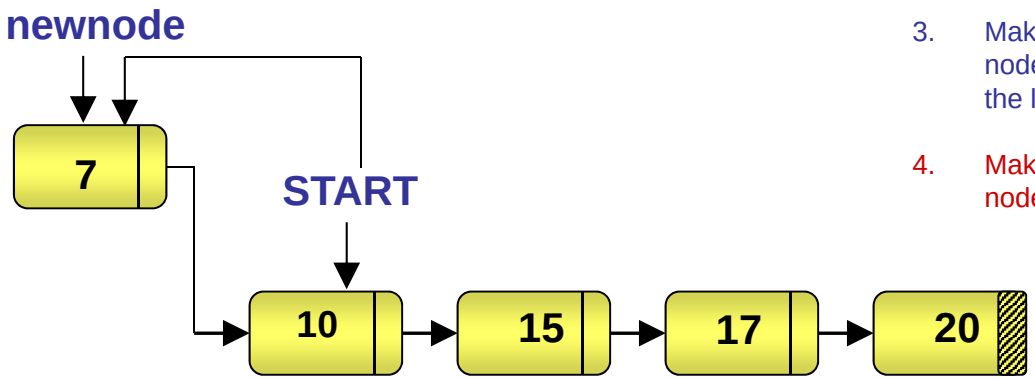
Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. Make the next field of the new node point to the first node in the list.
4. Make START, point to the new node.

newnode. next = START

Inserting a Node in a Singly-Linked List (Contd.)



1. Allocate memory for the new node.
2. Assign value to the data field of the new node.
3. Make the next field of the new node point to the first node in the list.
4. Make START, point to the new node.

Insertion complete

newnode. next = START

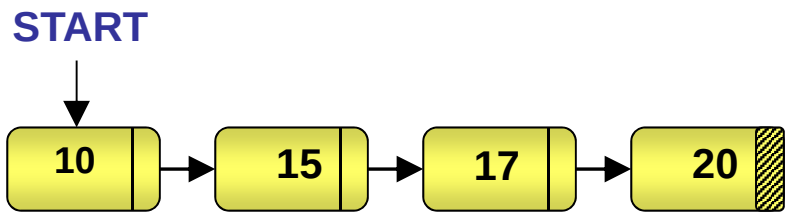
START = newnode

Inserting a Node in a Singly-Linked List (Contd.)

- Write an algorithm to insert a node between two nodes in a linked list.

Inserting a Node in a Singly-Linked List (Contd.)

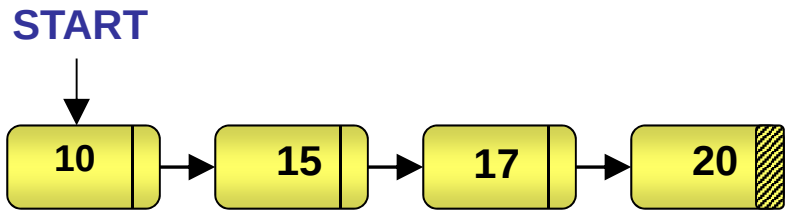
- Algorithm to insert a node between two nodes in a linked list.



1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

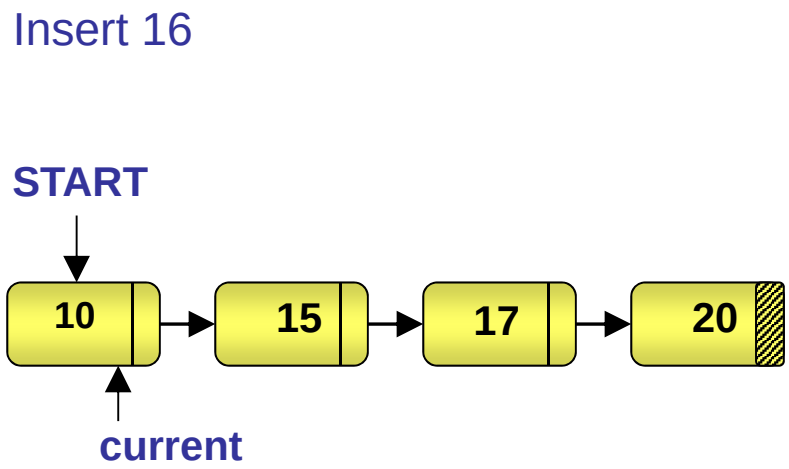
Inserting a Node in a Singly-Linked List (Contd.)

Insert 16



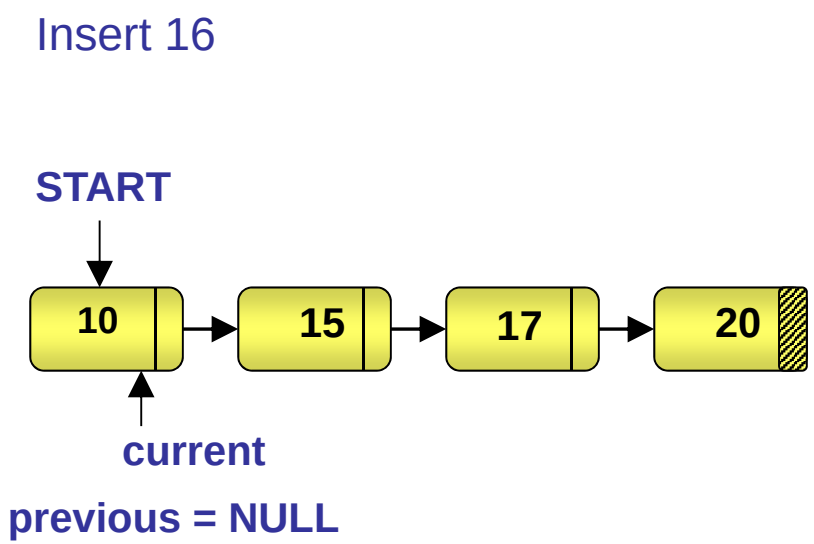
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



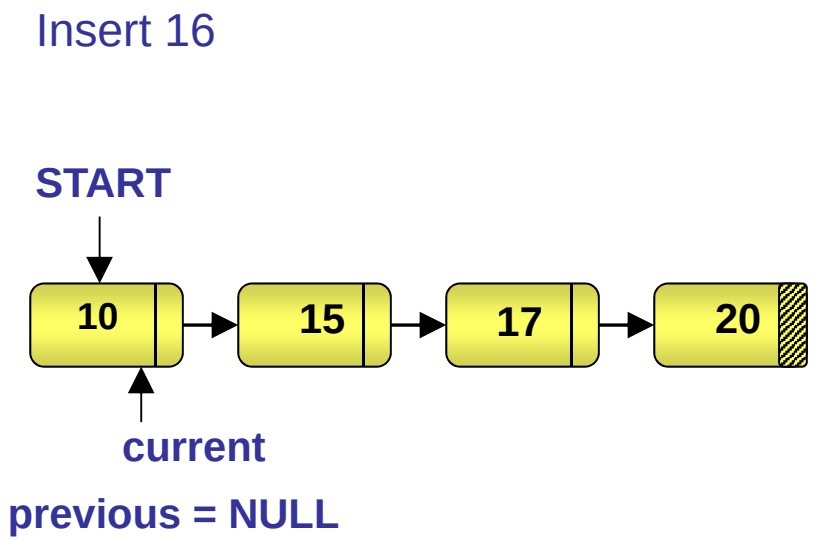
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



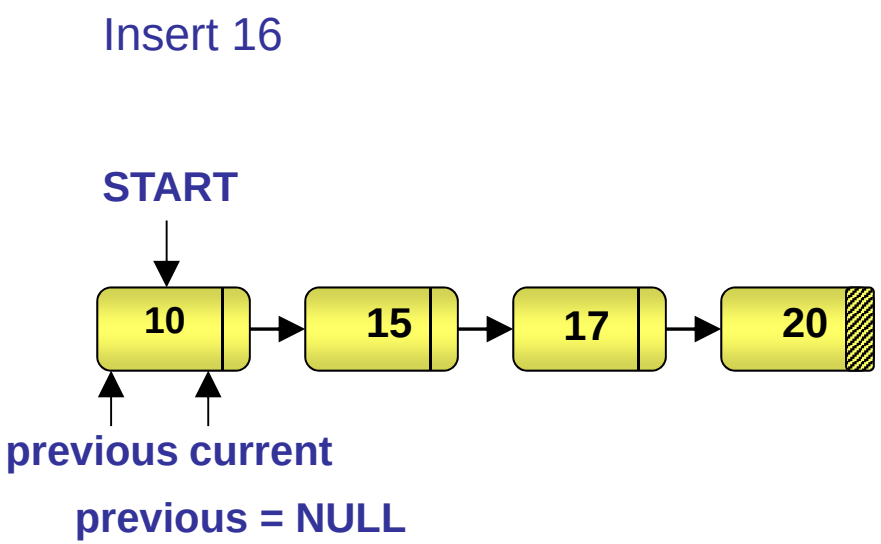
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



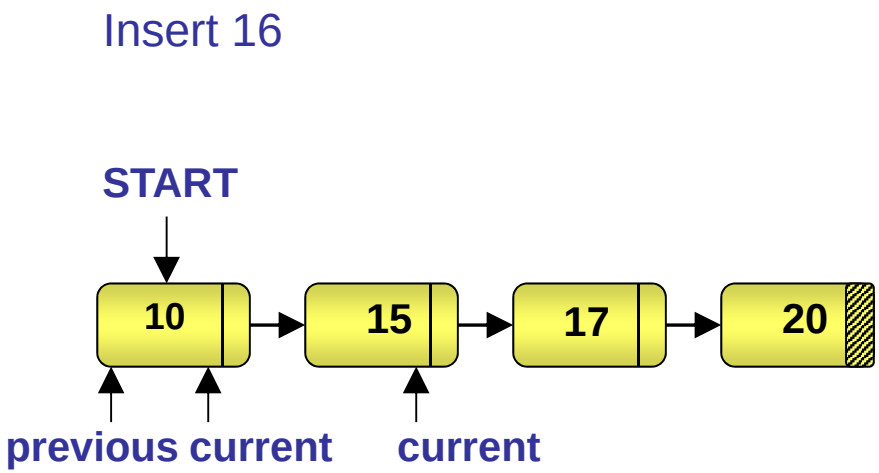
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



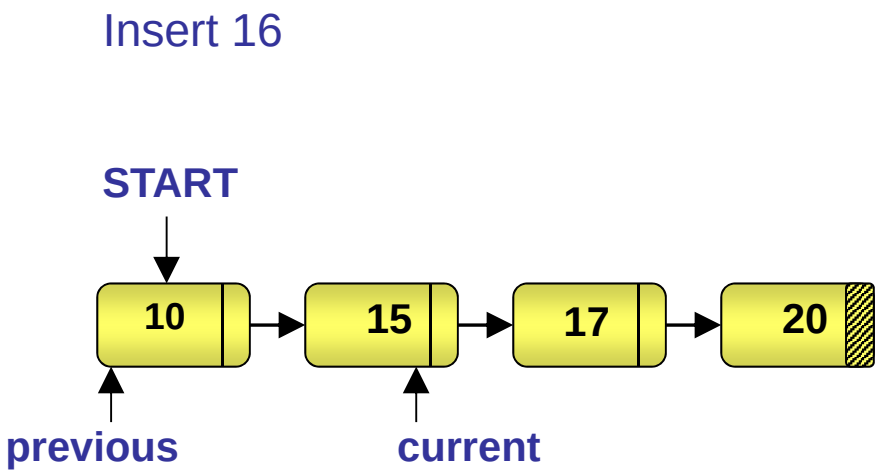
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



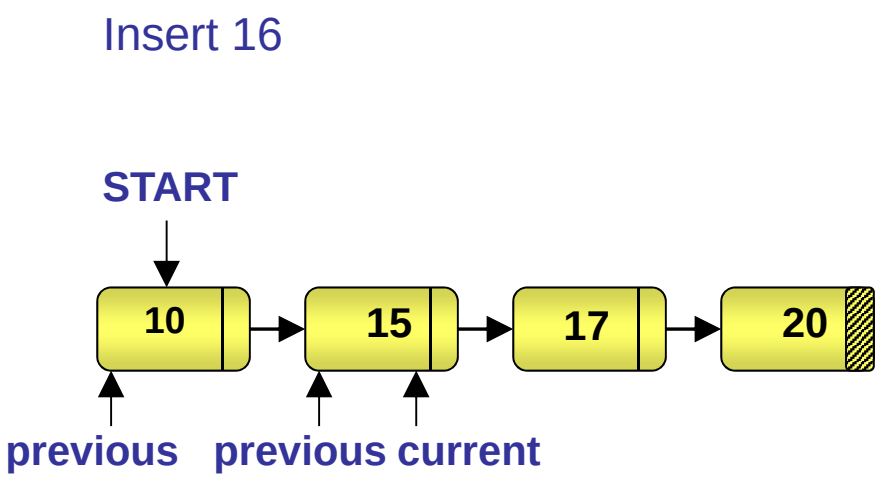
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

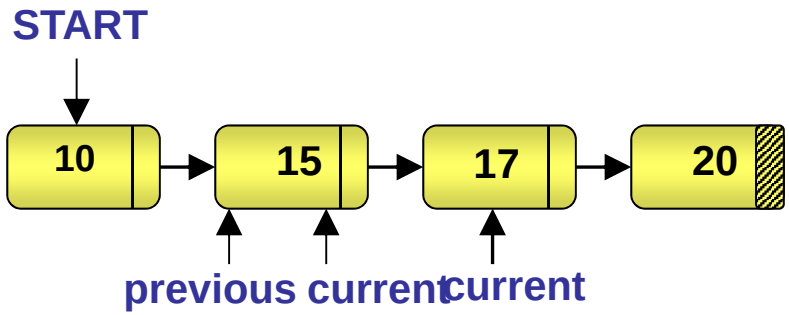
Inserting a Node in a Singly-Linked List (Contd.)



1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

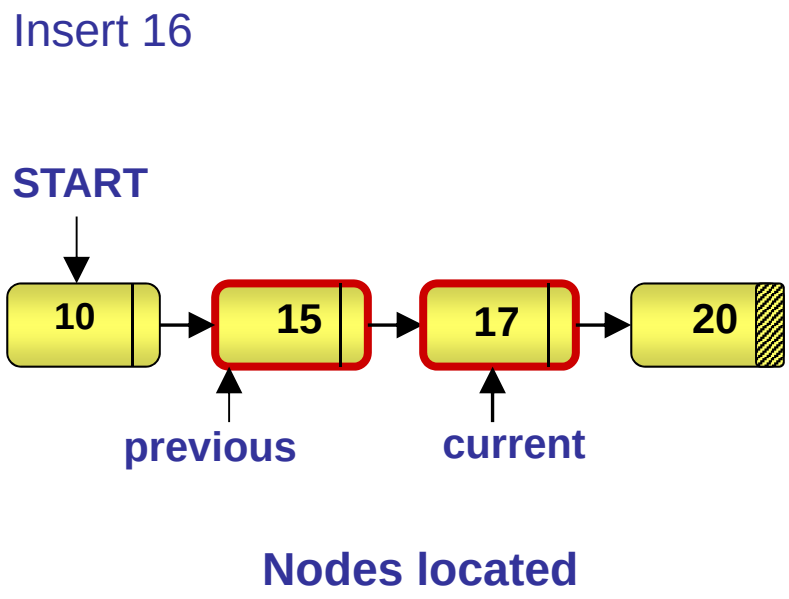
Inserting a Node in a Singly-Linked List (Contd.)

Insert 16



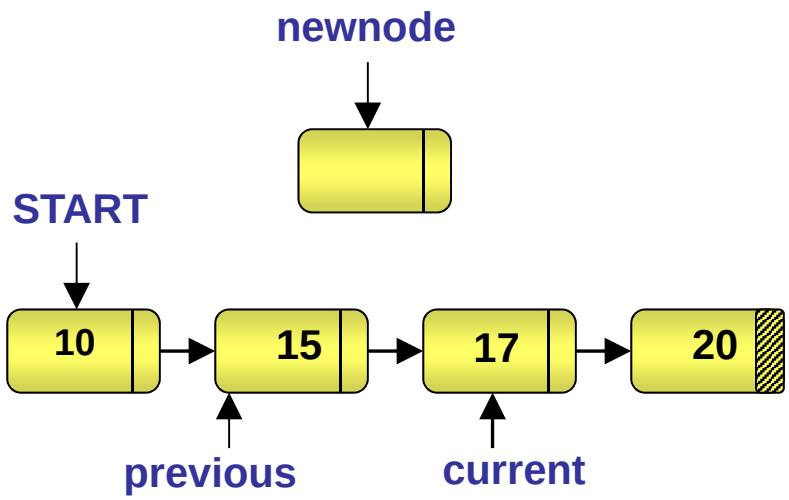
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

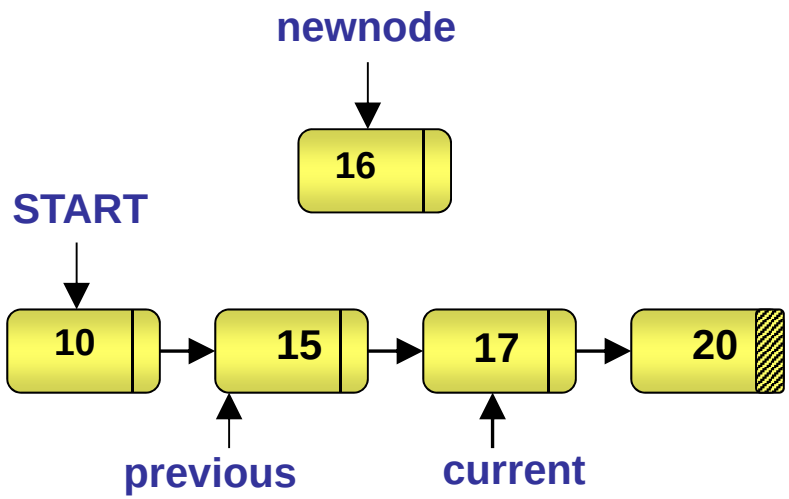
Inserting a Node in a Singly-Linked List (Contd.)



Nodes located

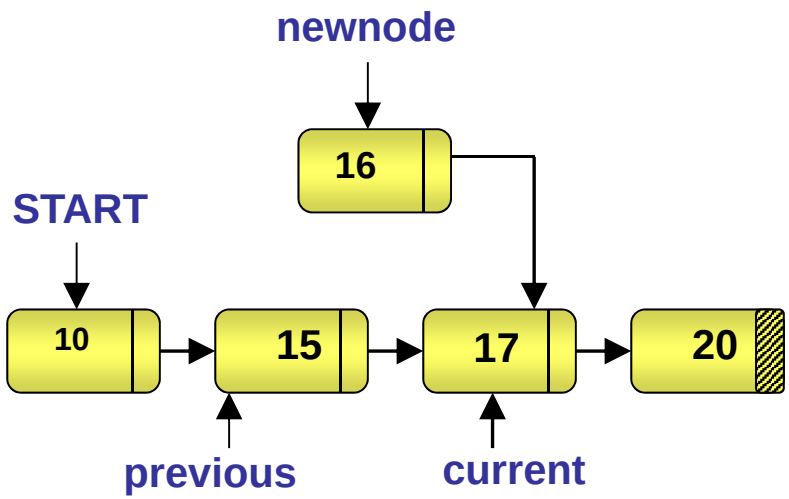
1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

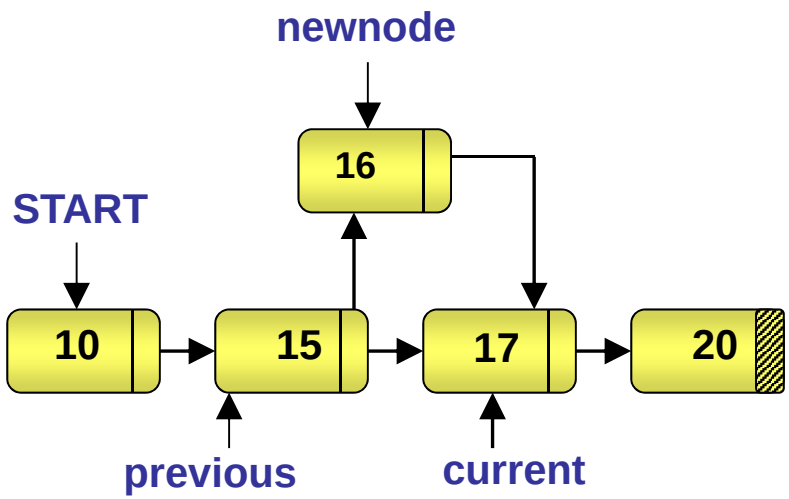
Inserting a Node in a Singly-Linked List (Contd.)



newnode.next = current

1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Inserting a Node in a Singly-Linked List (Contd.)



newnode.next = current

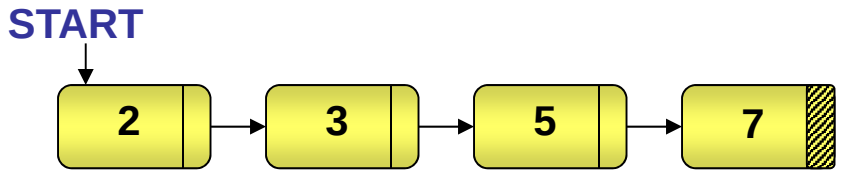
previous.next = newnode

Insertion complete

1. Identify the nodes between which the new node is to be inserted. Mark them as previous and current. To locate previous and current, execute the following steps:
   a. Make current point to the first node.
   b. Make previous point to NULL.
   c. Repeat step d and step e until current.info becomes greater than newnode.info or current becomes equal to NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Allocate memory for the new node.
3. Assign value to the data field of the new node.
4. Make the next field of the new node point to current.
5. Make the next field of previous point to the new node.

Traversing a Singly-Linked List

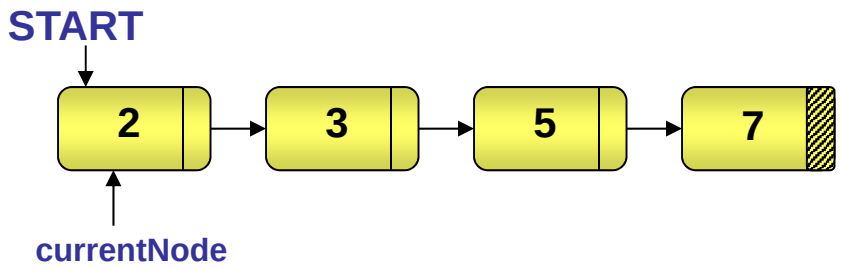
- Algorithm for traversing a linked list.
- Write an algorithm to traverse a singly-linked list.



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

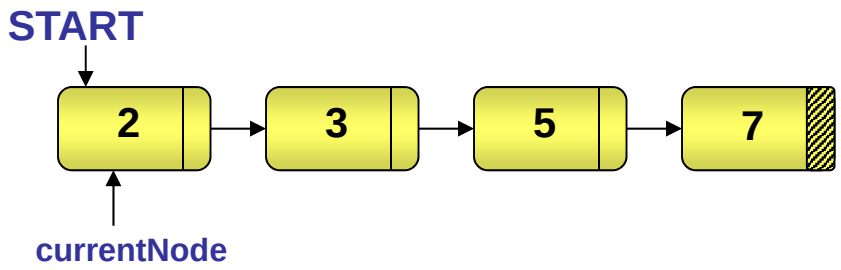
Traversing a Singly-Linked List (Contd.)

- Refer to the algorithm to display the elements in the linked list.



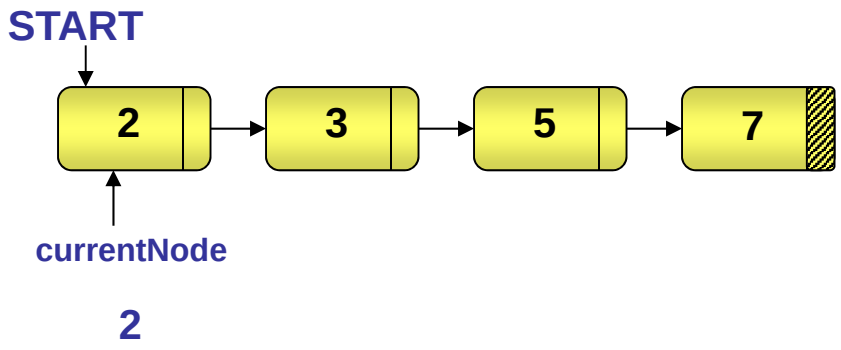
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



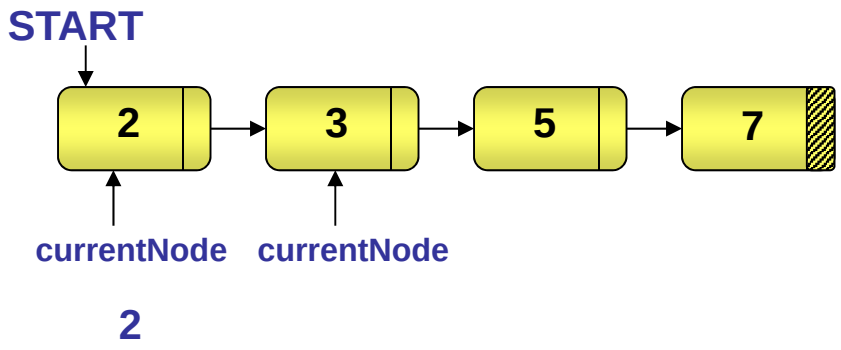
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



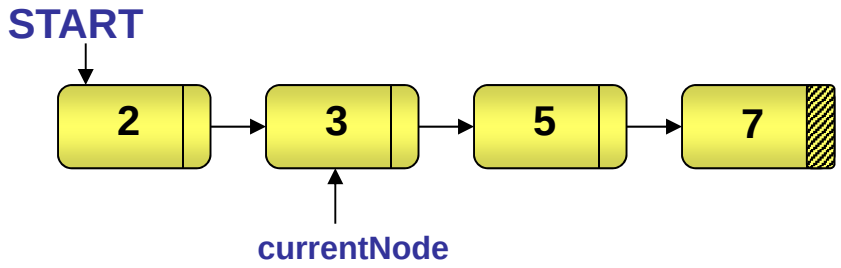
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

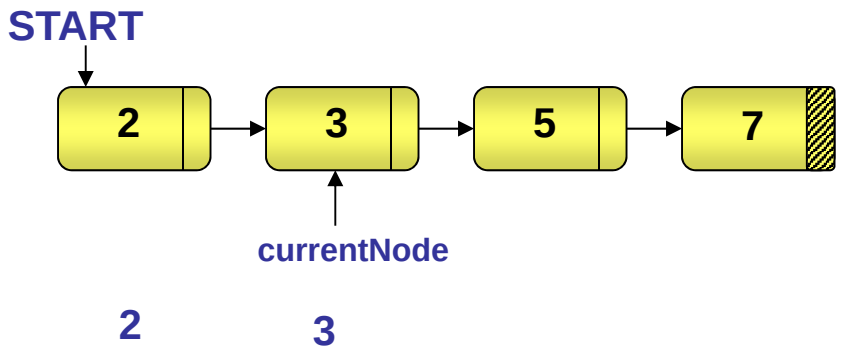
Traversing a Singly-Linked List (Contd.)



2

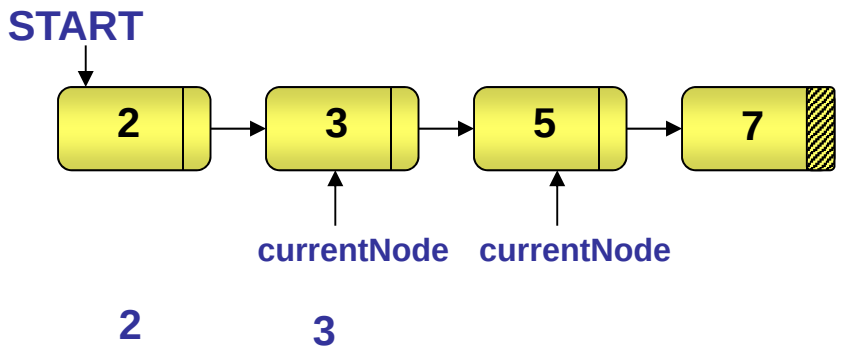
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



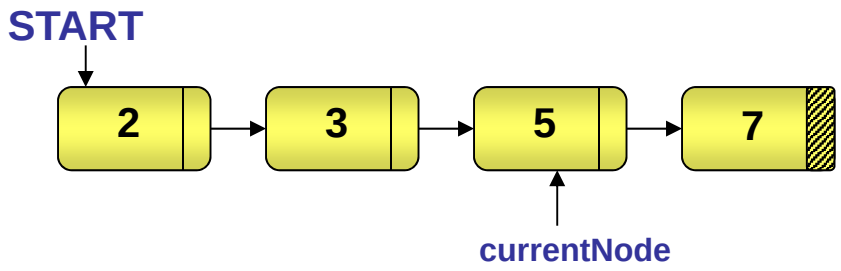
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

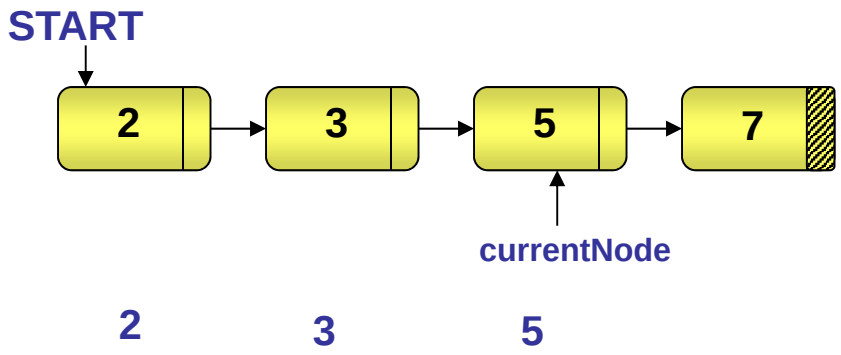
Traversing a Singly-Linked List (Contd.)



2 3

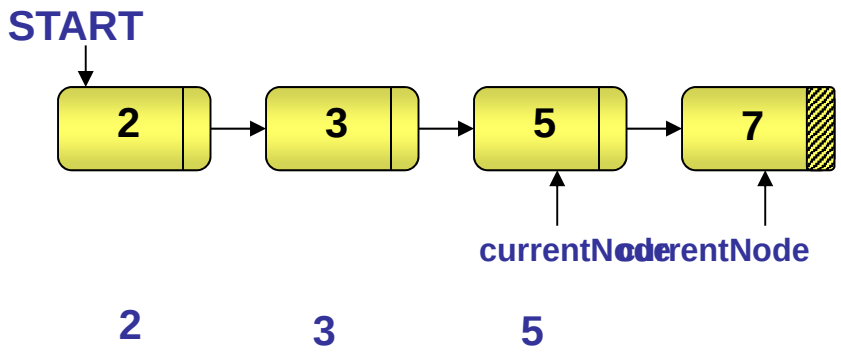
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



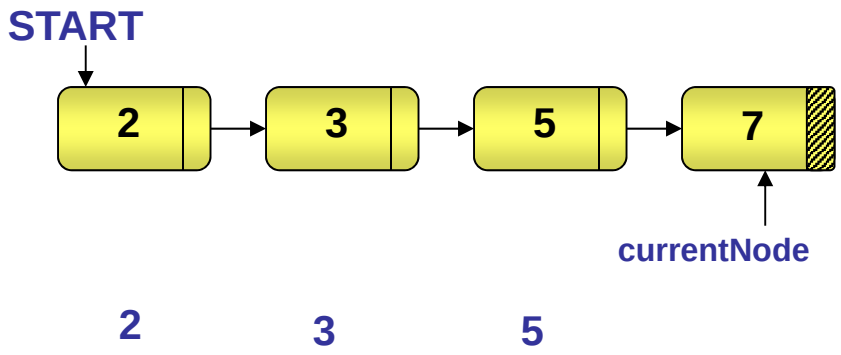
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



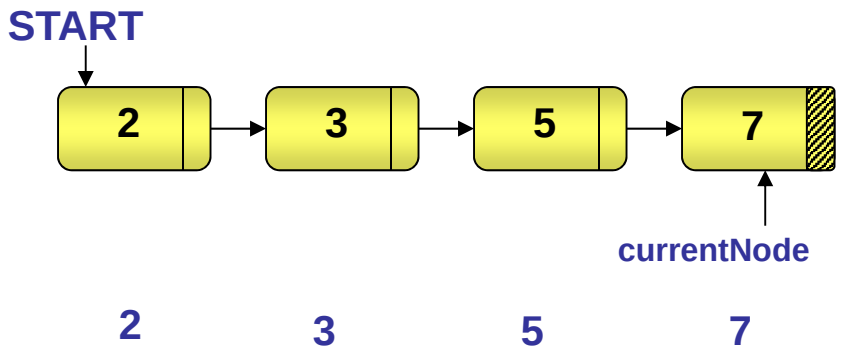
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



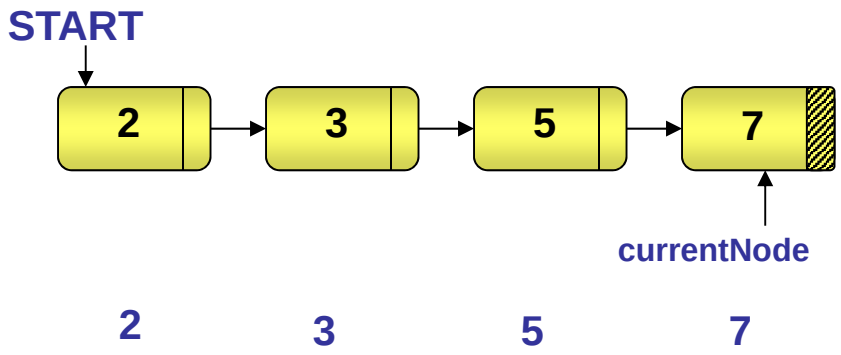
1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

Traversing a Singly-Linked List (Contd.)



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

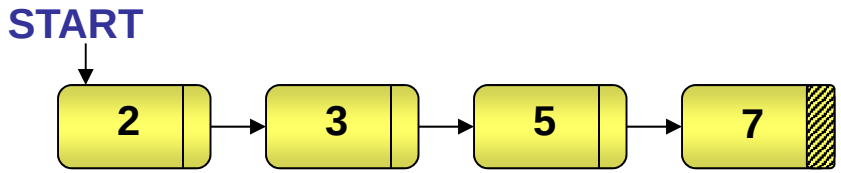
Traversing a Singly-Linked List (Contd.)



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

currentNode = NULL

Traversing a Singly-Linked List (Contd.)



1. Make currentNode point to the first node in the list.
2. Repeat step 3 and 4 until currentNode becomes NULL.
3. Display the information contained in the node marked as currentNode.
4. Make currentNode point to the next node in sequence.

currentNode = NULL

2 3 5 7

Traversal complete

Deleting a Node from Singly-Linked List

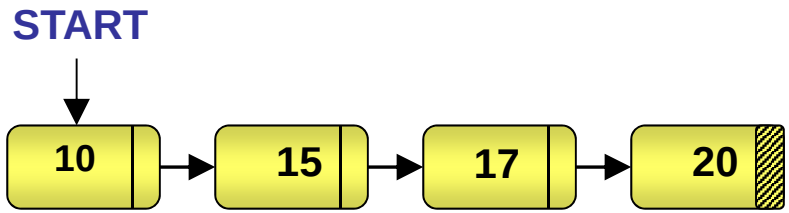
- Delete operation in a linked list refers to the process of removing a specified node from the list.
- You can delete a node from the following places in a linked list:
  - Beginning of the list
  - Between two nodes in the list
  - End of the list

- Write an algorithm to delete the first node in a linked list.

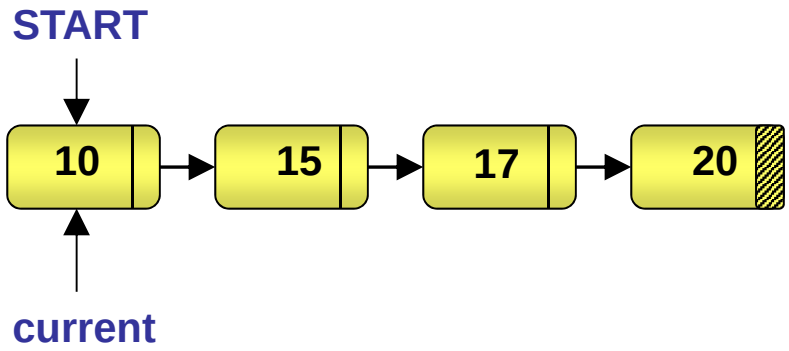
Deleting a Node from the Beginning of the List

- Algorithm to delete a node from the beginning of a linked list.

1. Mark the first node in the list as current.
2. Make START point to the next node in its sequence.
3. Release the memory for the node marked as current.



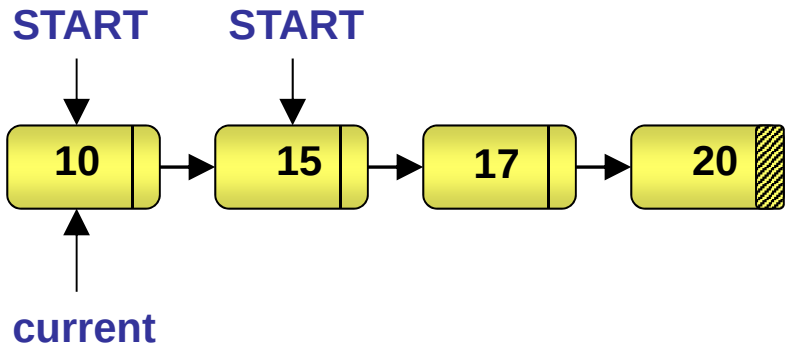
Deleting a Node from the Beginning of the List (Contd.)



1. Mark the first node in the list as current.
2. Make START point to the next node in its sequence.
3. Release the memory for the node marked as current.

current = START

Deleting a Node from the Beginning of the List (Contd.)

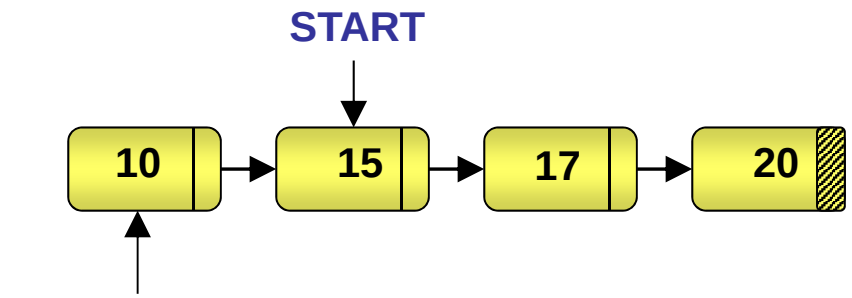


1. Mark the first node in the list as current.
2. Make START point to the next node in its sequence.
3. Release the memory for the node marked as current.

current = START
START = START. next

Deleting a Node from the Beginning of the List (Contd.)

1. Mark the first node in the list as current.
2. Make START point to the next node in its sequence.
3. Release the memory for the node marked as current.



Memory released

current = START

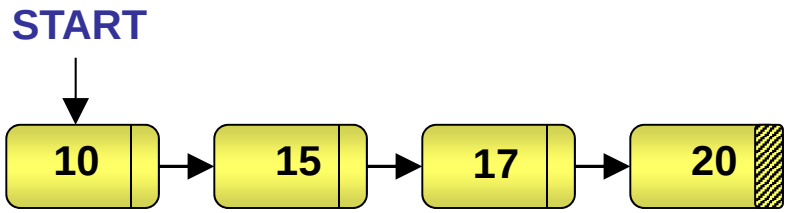
START = START. next

Delete operation complete

Deleting a Node Between two Nodes in the List

- Write an algorithm to delete a node between two nodes in a linked list.

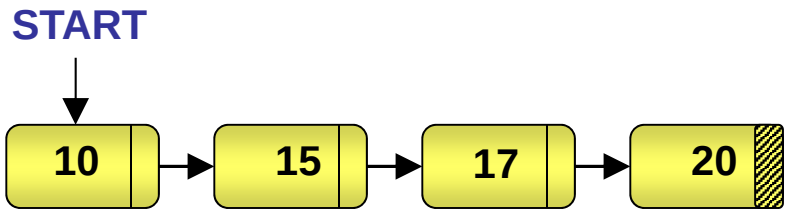
- Algorithm to delete a node between two nodes in the list.



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
    a. Set previous = START
    b. Set current = START
    c. Repeat step d and e until either the node is found or current becomes NULL.
    d. Make previous point to current .
    e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

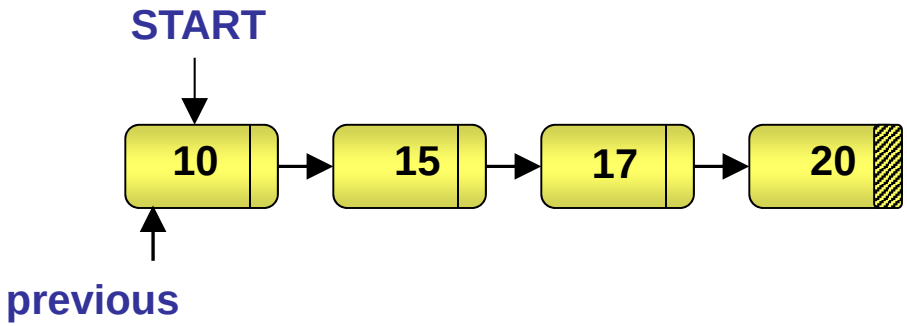
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

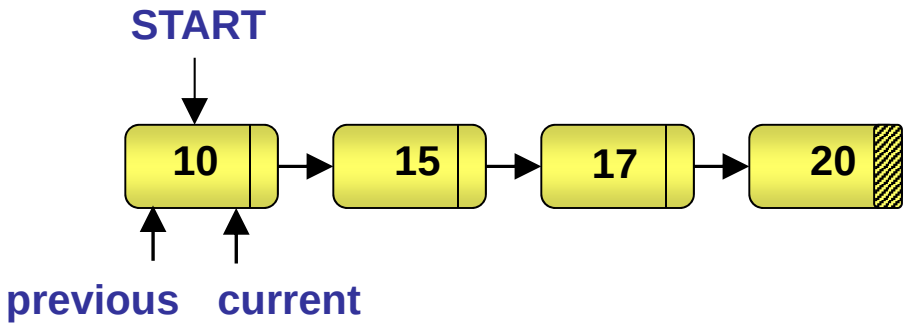
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

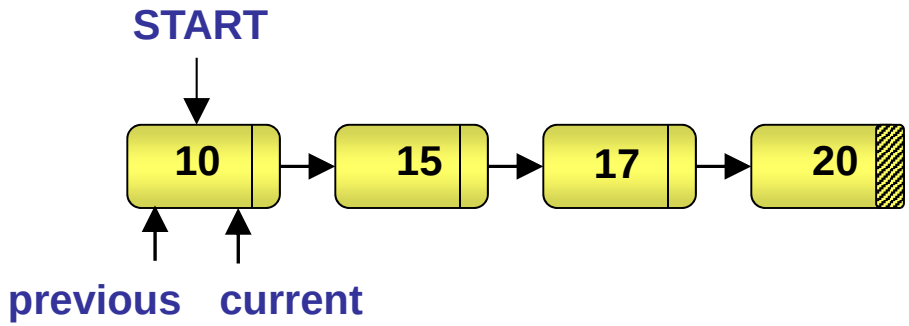
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

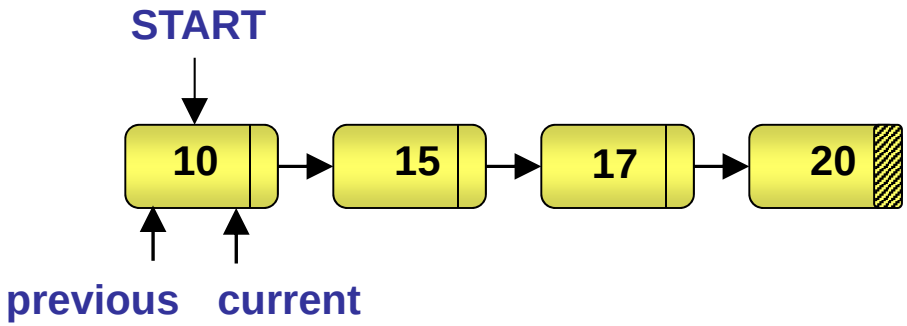
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

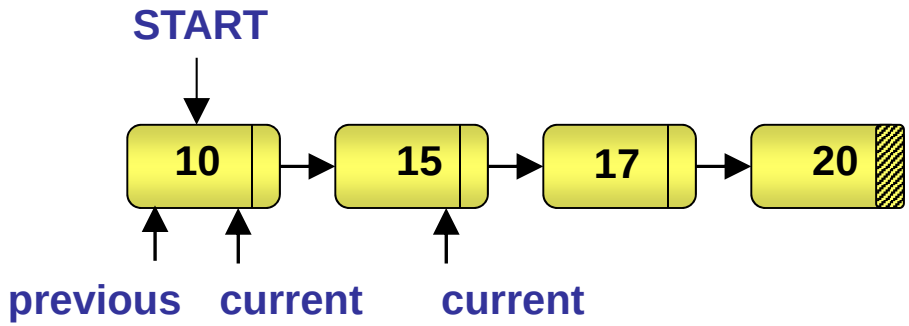
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

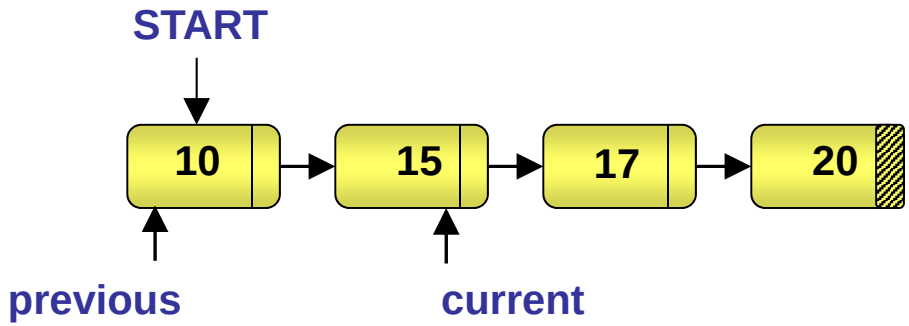
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

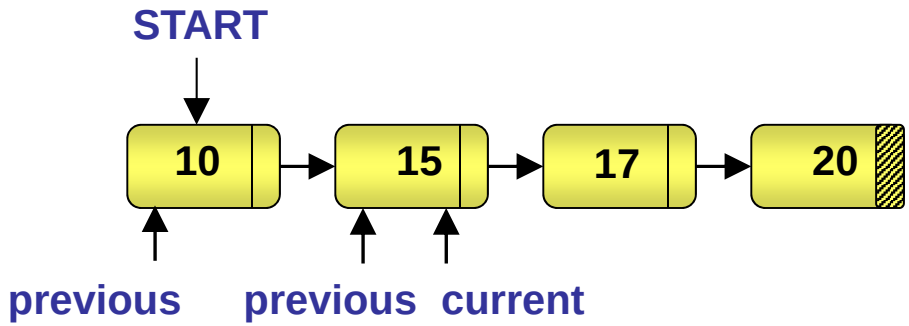
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

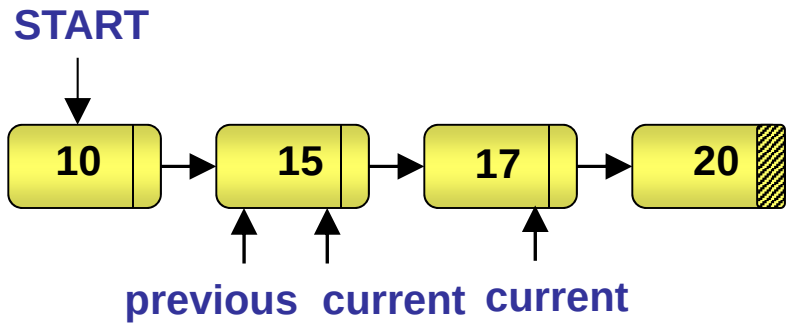
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

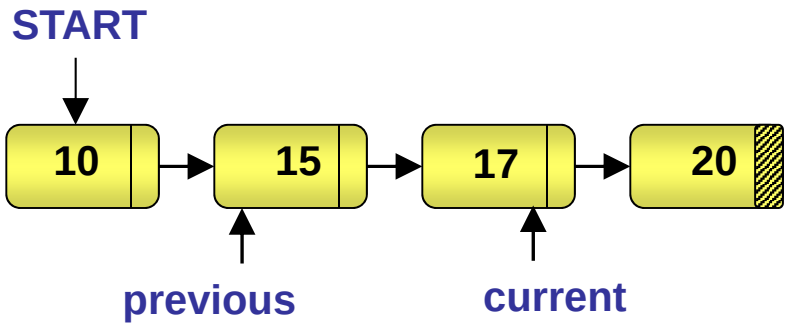
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

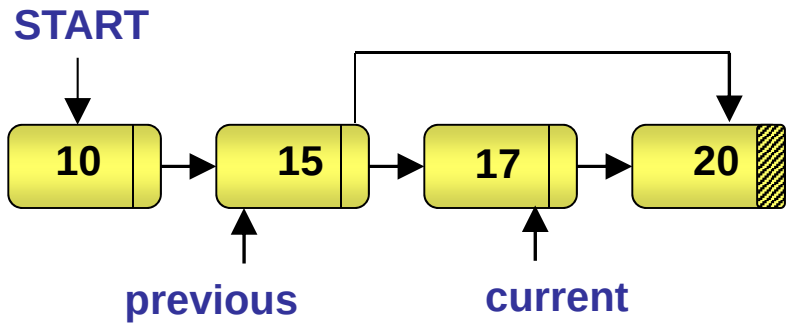
- Delete 17



1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

- Delete 17



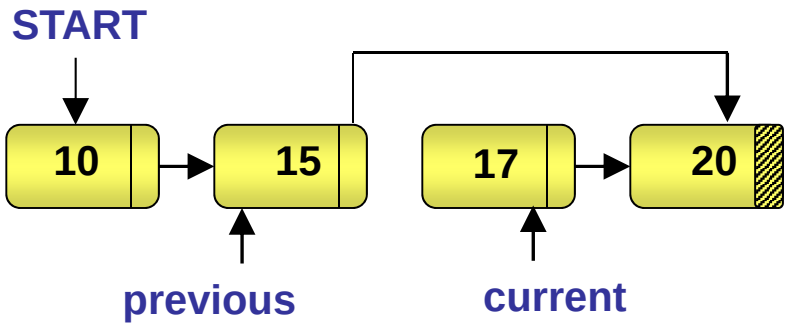
previous.next = current.next

1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Deleting a Node Between two Nodes in the List (Contd.)

- Delete 17

Delete operation complete



previous.next = current.next

1. Locate the node to be deleted. Mark the node to be deleted as current and its predecessor as previous. To locate current and previous, execute the following steps:
   a. Set previous = START
   b. Set current = START
   c. Repeat step d and e until either the node is found or current becomes NULL.
   d. Make previous point to current.
   e. Make current point to the next node in sequence.
2. Make the next field of previous point to the successor of current.
3. Release the memory for the node marked as current.

Representing a Singly-Linked List

- A Linked list is represented in a program by defining two classes:
  - Node class: This class contains the data members of varying data types, which represent data to be stored in a linked list. It also contains the reference of the class type (Node) to hold the reference of the next node in sequence.

```
// Code in C++
class Node
{
          public:
          int data;
          Node *next;
};
```

```
class List
 {
  Node * START;
  public:
  List()
  {
     START = NULL;
  }
 void addNode(int element);
 bool search(int element, Node *previous, Node *current);
 bool delNode(int element);
 void traverse();
 };
```

Summary

- In this session, you learned that:
  - In a singly-linked list, each node contains:
    - The information
    - The address of the next node in the list
- Singly-linked list can be traversed only in a single direction.
- Insertion and deletion in a linked list is fast as compared to arrays. However, accessing elements is faster in arrays as compared to linked lists.

Self Review Questions

1. Discuss the advantages and disadvantages of linked lists.
2. Discuss the differences between arrays and linked lists.
3. Write a program to implement insert, search, delete, and traverse operations on a singly-linked list that stores the records of the students in a class. Each record holds the following information:
    - Roll number of the student
    - Name of the student