Introduction to Linked List

Bharadwaj

IIT Madras

August 25, 2014

What is a Linked List?

- Linked List is a collection of *nodes*
- Each node contains a data and a pointer to the next node in the list.

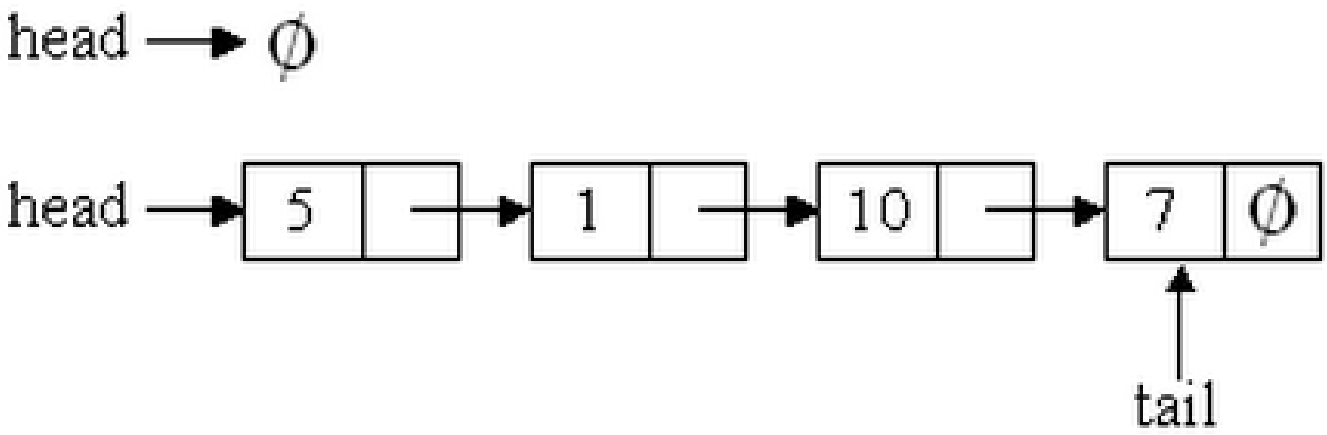

Figure 1 : Linked List

- *head* is a pointer to the first node in the linked list
- The last node has a pointer to NULL

Linked List Node structure

```
struct Node{
    int data;
    struct Node *next;
};
```

Operations on Linked Lists

- Create
- Insert
- Delete

Create

- You have the linked list structure.
- You have an unintialized pointer **head*
- You need to create a linked list with a single node with a value k.

Create

- You have the linked list structure.
- You have an unintialized pointer **head*
- You need to create a linked list with a single node with a value k.
- Use malloc to allocate memory for one node.
  $head = (struct\ Node*)malloc(sizeof(struct\ Node))$
- Initialize data to k

The $->$ operator

- *head* is a pointer to the node
- To access the node pointed to by head we need to dereference it using *
- Hence, we write:

$$(*head).data = k$$

- Or equvivalently, we can write:

$$head->data = k$$

Traversal

- **Traversal in array**

```
void traverse(int arr[], int n)
{   int i=0;
    while(i<n)
    {   printf("%d ",arr[i]);
        i++;
    }
}
```

- **Traversal in Linked List**

```
void traverse(struct Node* head)
{   struct Node *temp=head;
    while(temp!=NULL)
    {   printf("%d ",temp->data);
        temp = temp->next;
    }
}
```

Insert

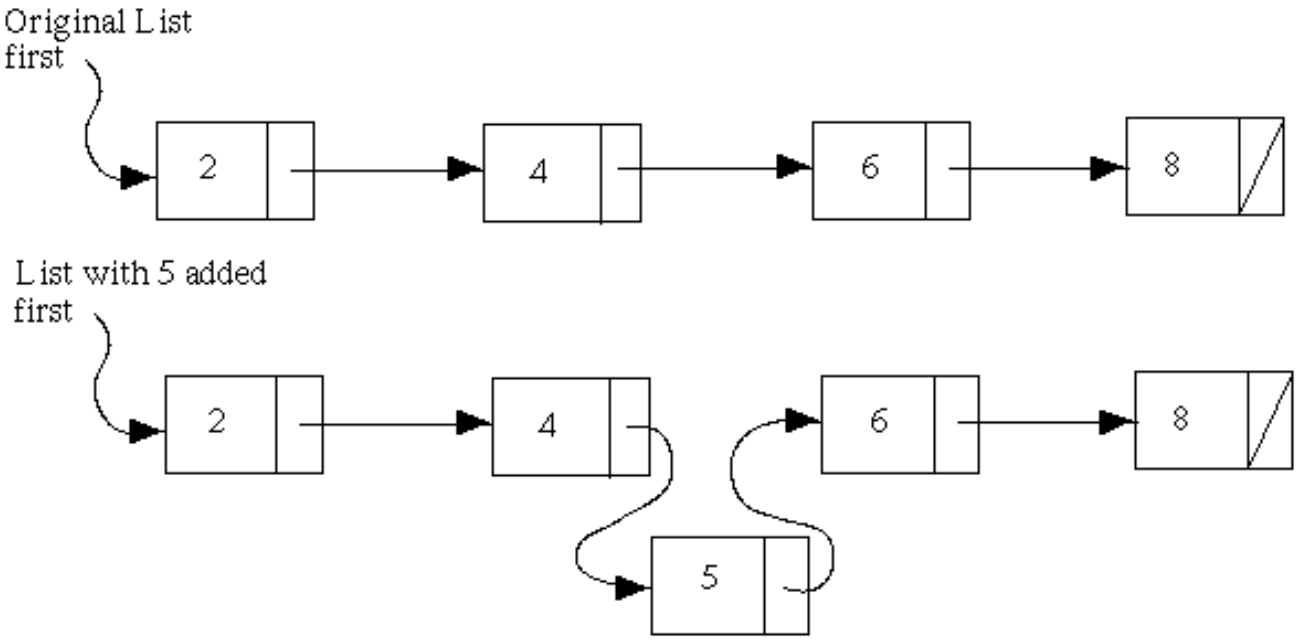

Figure 2 : Insertion in Linked List

Special Cases:

1. Insertion at the beginning
2. Insertion at the end

Delete

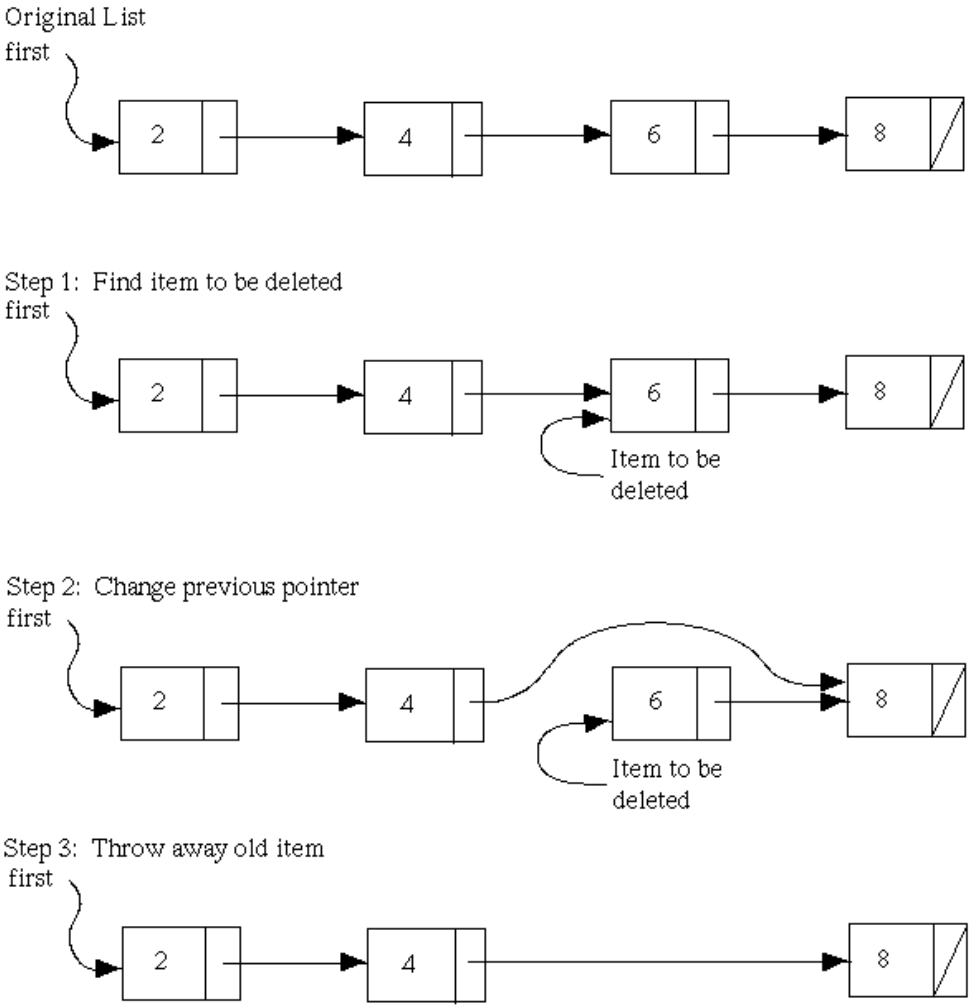

Figure 3 : Deletion in Linked List

Delete

Special Cases:

1. Deletion at the beginning
2. Deletion at the end

**** DO NOT FORGET TO FREE THE DELETED NODE!!!**

References

1. http://cslibrary.stanford.edu/103/LinkedListBasics.pdf