Introduction to Doubly linked list Tutorial 03-Part I

A **Doubly Linked List (DLL)** is a type of linked list in which each node contains three components:

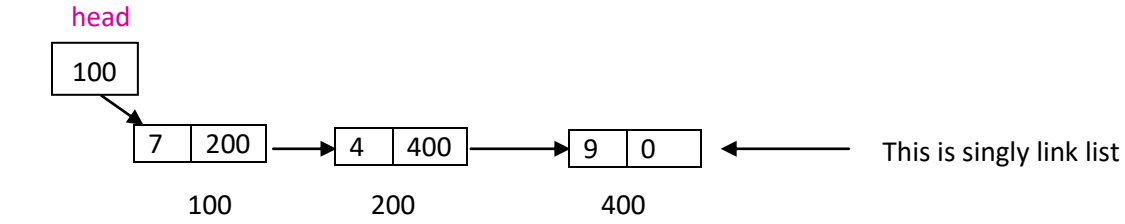
1. **Data** – The value stored in the node.
2. **Pointer to the Next Node** – Points to the next node in the list.
3. **Pointer to the Previous Node** – Points to the previous node in the list.

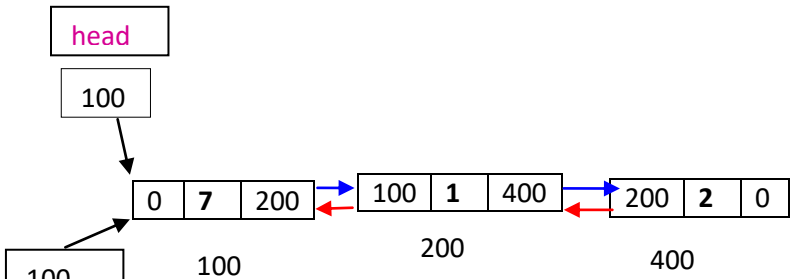
Advantages of Doubly Linked List

- Can be traversed in both directions (forward and backward).
- Easier to delete a node without needing extra traversal.
- More efficient than a singly linked list for certain operations.

Explanation of Operations

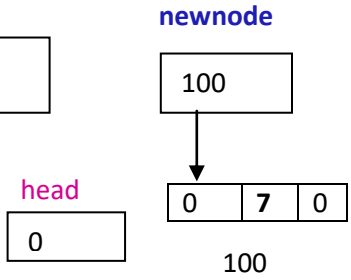
1. **Implementation of Doubly Linked List**
2. **Displaying the List**: Prints the list from head to tail.
3. **Insertion at the Beginning**: Adds a new node before the current head.
4. **Insertion at the End**: Adds a new node at the last position.
5. **Deletion of a Node**: Finds the node with a given value and removes it.






```
struct node
{
int data;
struct node*next;
struct node*prev;
};
struct node *head, *newnode;
```

User defined data type



Now you are going to define the **create()** function.

```
void create(){
head=0; struct node *temp;
newnode= (struct node*)malloc(sizeof(struct node));
printf("Enter Data: ");
scanf("%d", &newnode->data);
newnode->prev=0;
newnode->next=0;
if(head==0)
{
//head=newnode;
   head=temp=newnode;
}
else
{
//head->next=newnode;
//newnode->prev=head;
temp->next=newnode;
newnode->prev=temp;
temp=newnode;
}
```

When we assign the values for third node we need **temp**

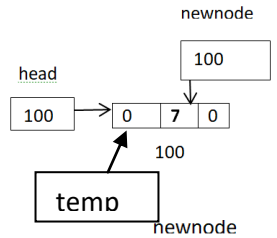
You are going to create the first node dynamically. We are going to assign the memory to this node dynamically. It can be done using malloc function. Node you want how much size is 12 bytes.

Malloc() function will return a pointer to the first bite of that allocated block.

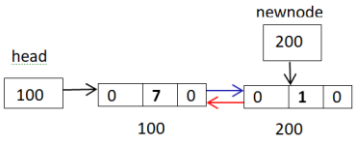
Now asking user enters data, starting head is null.

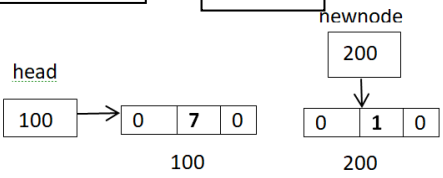
This is the node. Address of first byte is 100. Malloc is going to return what a pointer to the first byte. You can say that void pointer.

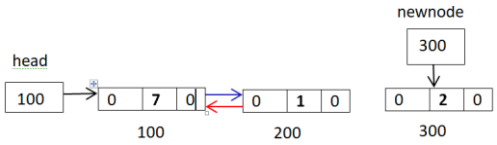
You have to store this address into some another pointer variable obviously.so we use a pointer variable now. We are going to declare another pointer variable. Suppose we declare a pointer as **newnode**.



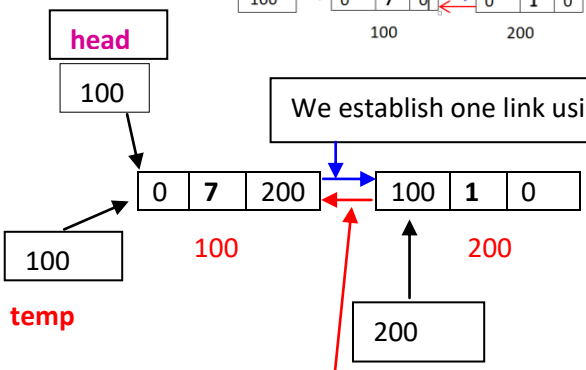
How to insert second node

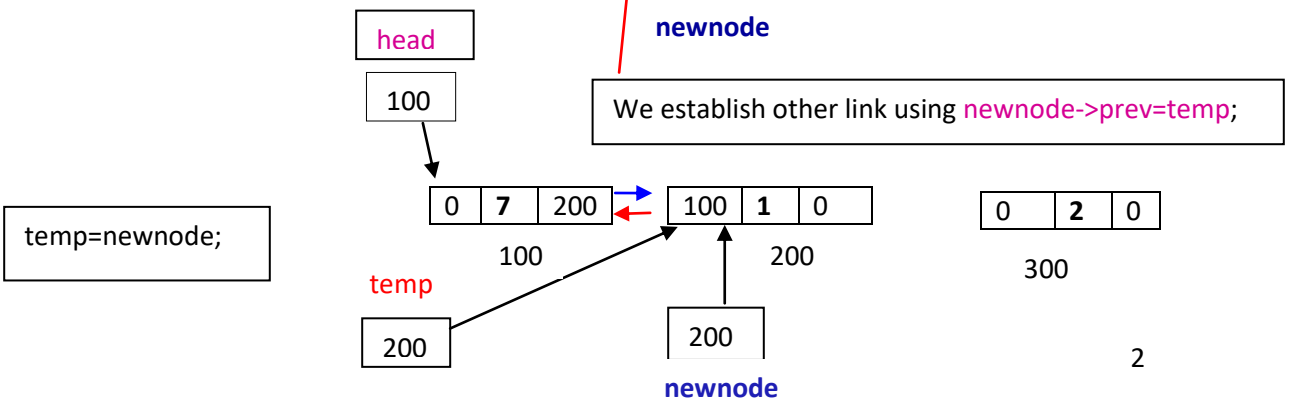


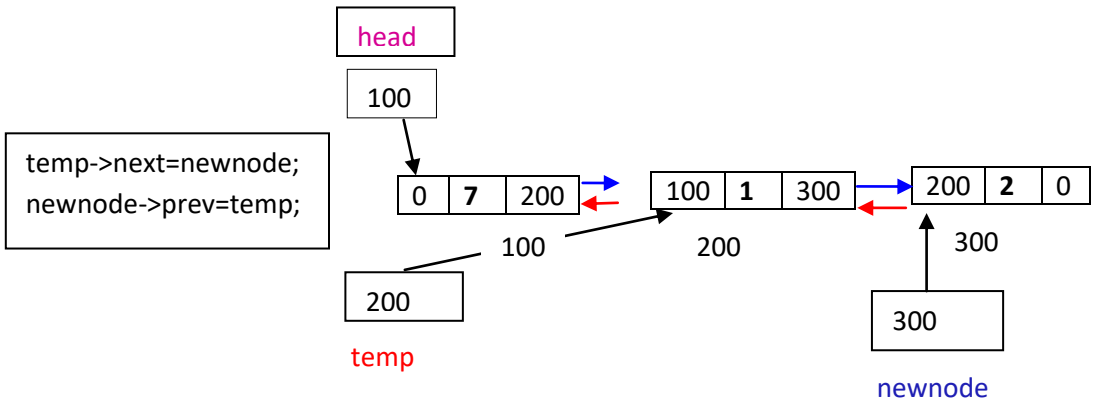


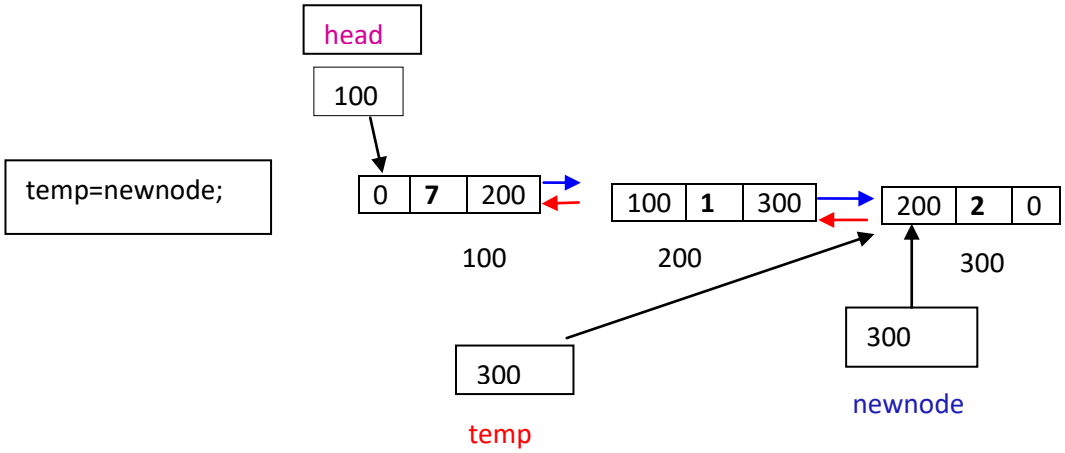


But problem comes when we insert third node









display() function

```
void display()
{
struct node *temp;
temp=head;
while(temp!=0)
{
printf("%d",temp->data);
temp=temp->next;
}
}
```

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
int data;
struct node*next;
struct node*prev;
};
struct node *head, *newnode;

void create(){
   head=0; struct node *temp;
   int choice=1;
   while(choice){
newnode= (struct node*)malloc(sizeof(struct node));
printf("Enter Data:");
scanf("%d", &newnode->data);

newnode->prev=0;
newnode->next=0;
              if(head==0)
              {
              head=temp=newnode;
             }
else
{
temp->next=newnode;
newnode->prev=temp;
temp=newnode;
}
printf("Do you want to continue:");
scanf("%d",&choice);
}
}
void display()
{
struct node *temp;
temp=head;
while(temp!=0)
{
printf("%d",temp->data);
temp=temp->next;
}
}

int main() {
   create();
   display();
    return 0;
}
```