

## Deletion in singly linked list at the end

There are two scenarios in which, a node is deleted from the end of the linked list.

1. There is only one node in the list and that needs to be deleted.
2. There are more than one node in the list and the last node of the list will be deleted.

### In the first scenario,

the condition `head → next = NULL` will survive and therefore, the only node head of the list will be assigned to null. This will be done by using the following statements.

```
ptr = head  
    head = NULL  
    free(ptr)
```

### In the second scenario,

The condition `head → next = NULL` would fail and therefore, we have to traverse the node in order to reach the last node of the list.



For this purpose, just declare a temporary pointer temp and assign it to head of the list. We also need to keep track of the second last node of the list. For this purpose, two pointers ptr and ptr1 will be used where ptr will point to the last node and ptr1 will point to the second last node of the list.

this all will be done by using the following statements.

```
ptr = head;
    while(ptr->next != NULL)
    {
        ptr1 = ptr;
        ptr = ptr ->next;
    }
```

Now, we just need to make the pointer ptr1 point to the NULL and the last node of the list that is pointed by ptr will become free. It will be done by using the following statements.

```
ptr1->next = NULL;
    free(ptr);
```

## Algorithm

- **Step 1:** IF HEAD = NULL

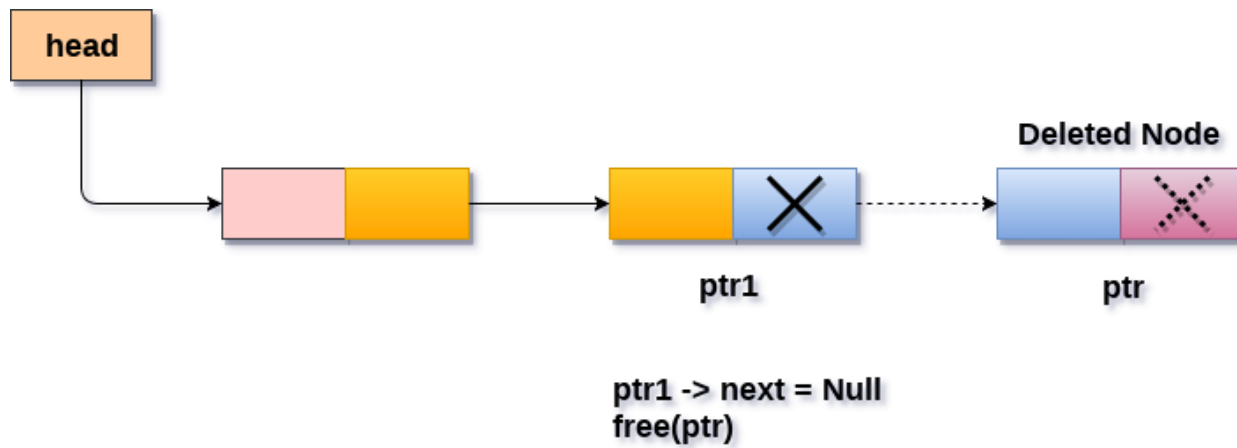
Write UNDERFLOW  
Go to Step 8  
[END OF IF]

- **Step 2:** SET PTR = HEAD
- **Step 3:** Repeat Steps 4 and 5 while PTR -> NEXT!= NULL
- **Step 4:** SET PREPTR = PTR
- **Step 5:** SET PTR = PTR -> NEXT



[END OF LOOP]

- **Step 6:** SET PREPTR -> NEXT = NULL
- **Step 7:** FREE PTR
- **Step 8:** EXIT



**Deleting a node from the last**

C Function :

```
#include<stdio.h>
#include<stdlib.h>
void create(int);
void end_delete();
struct node
{
    int data;
    struct node *next;
};
struct node *head;
void main ()
{
    int choice,item;
    do
    {
        printf("\n1.Append List\n2.Delete node\n3.Exit\n4.Enter your choice?");
        scanf("%d",&choice);
        switch(choice)
        {
```



```
    case 1:
        printf("\nEnter the item\n");
        scanf("%d",&item);
        create(item);
    break;
    case 2:
        end_delete();
    break;
    case 3:
        exit(0);
    break;
    default:
        printf("\nPlease enter valid choice\n");
}

} while(choice != 3);
}

void create(int item)
{
    struct node *ptr = (struct node *)malloc(sizeof(struct node *));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW\n");
    }
    else
    {
        ptr->data = item;
        ptr->next = head;
        head = ptr;
    }
}
```



```
        printf("\nNode inserted\n");
    }

}

void end_delete()
{
    struct node *ptr,*ptr1;
    if(head == NULL)
    {
        printf("\nlist is empty");
    }
    else if(head -> next == NULL)
    {
        head = NULL;
        free(head);
        printf("\nOnly node of the list deleted ...");
    }

    else
    {
        ptr = head;
        while(ptr->next != NULL)
        {
            ptr1 = ptr;
            ptr = ptr ->next;
        }
        ptr1->next = NULL;
        free(ptr);
        printf("\n Deleted Node from the last ...");
    }
}
```



```
}  
}
```

## Output

```
1.Append List  
2.Delete node  
3.Exit  
4.Enter your choice?1
```

```
Enter the item  
12
```

```
Node inserted
```

```
1.Append List  
2.Delete node  
3.Exit  
4.Enter your choice?2
```

```
Only node of the list deleted ...
```

← prev

next →





# linode

Root Access, 1GB RAM for Only \$5  
7 Day Money Back Guarantee.

Ad Root Access, 1GB RAM for Only \$5/m  
Money Back Guarantee.

Linode

[Learn more](#)

Please Share



Learn Latest Tutorials



Swift



Pig



Flask



C. Graphics



Automata



Testing

Preparation







Aptitude



Reasoning



Verbal A.



Interview

## B.Tech / MCA



DBMS



DS



DAA



OS



C. Network



Compiler D.



COA



D. Math.



Web Tech.



Cyber Sec.



C



C++



Java



.Net



Python



Programs





Control S.

