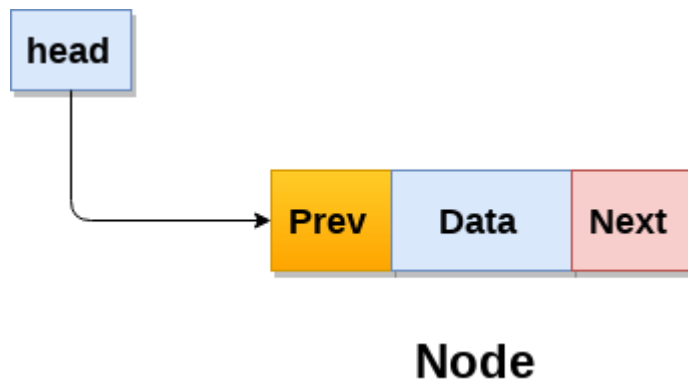# Doubly linked list

Doubly linked list is a complex type of linked list in which a node contains a pointer to the previous as well as the next node in the sequence. Therefore, in a doubly linked list, a node consists of three parts: node data, pointer to the next node in sequence (next pointer) , pointer to the previous node (previous pointer). A sample node in a doubly linked list is shown in the figure.

A doubly linked list containing three nodes having numbers from 1 to 3 in their data part, is shown in the following image.

Doubly Linked List

In C, structure of a node in doubly linked list can be given as :

```
struct node
{
    struct node *prev;
    int data;
    struct node *next;
}
```

The **prev** part of the first node and the **next** part of the last node will always contain null indicating end in each direction.

In a singly linked list, we could traverse only in one direction, because each node contains address of the next node and it doesn't have any record of its previous nodes. However, doubly linked list overcome this limitation of singly linked list. Due to the fact that, each node of the list contains the address of its previous node, we can find all the details about the previous node as well by using the previous address stored inside the previous part of each node.

Memory Representation of a doubly linked list

Memory Representation of a doubly linked list is shown in the following image. Generally, doubly linked list consumes more space for every node and therefore, causes more expansive basic operations such as insertion and deletion. However, we can easily manipulate the elements of the list since the list maintains pointers in both the directions (forward and backward).

In the following image, the first element of the list that is i.e. 13 stored at address 1. The head pointer points to the starting address 1. Since this is the first element being added to the list therefore the **prev** of the list **contains** null. The next node of the list resides at address 4 therefore the first node contains 4 in its next pointer.

We can traverse the list in this way until we find any node containing null or -1 in its next part.

Head

1

	Data	Prev	Next
1	13	-1	4
2			
3			
4	15	1	6
5			
6	19	4	8
7			
8	57	6	-1
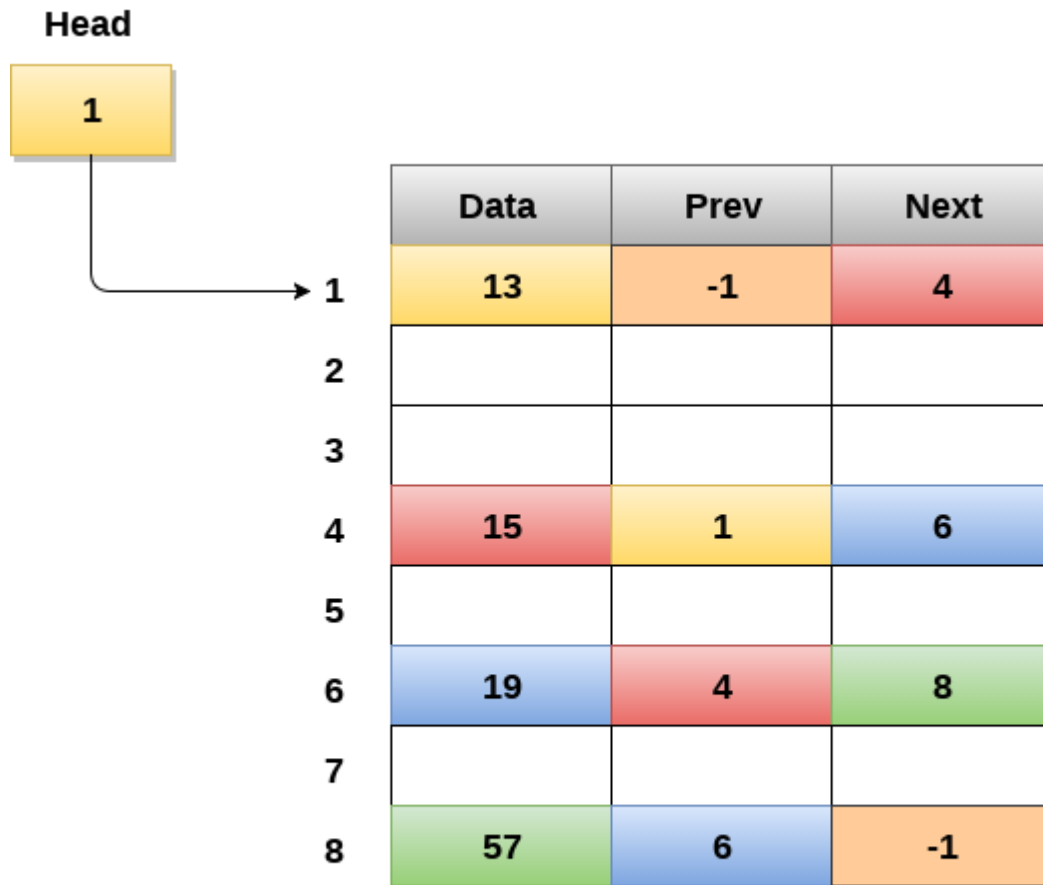
Memory Representation of a Doubly linked list

Operations on doubly linked list

Node Creation

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

All the remaining operations regarding doubly linked list are described in the following table.

SN	Operation	Description
1	Insertion at beginning	Adding the node into the linked list at beginning.
2	Insertion at end	Adding the node into the linked list to the end.

3	Insertion after specified node	Adding the node into the linked list after the specified node.
4	Deletion at beginning	Removing the node from beginning of the list
5	Deletion at the end	Removing the node from end of the list.
6	Deletion of the node having given data	Removing the node which is present just after the node containing the given data.
7	Searching	Comparing each node data with the item to be searched and return the location of the item in the list if the item found else return null.
8	Traversing	Visiting each node of the list at least once in order to perform some specific operation like searching, sorting, display, etc.

Menu Driven Program in C to implement all the operations of doubly linked list

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
    struct node *prev;
    struct node *next;
    int data;
};
struct node *head;
void insertion_beginning();
void insertion_last();
```

```c
void insertion_specified();
void deletion_beginning();
void deletion_last();
void deletion_specified();
void display();
void search();
void main ()
{
int choice =0;
    while(choice != 9)
    {
        printf("\n*********Main Menu*********\n");
        printf("\nChoose one option from the following list ...\n");
        printf("\n===============================================\n");
        printf("\n1.Insert in begining\n2.Insert at last\n3.Insert at any random location\n4.Delete from Beginning\n
        5.Delete from last\n6.Delete the node after the given data\n7.Search\n8.Show\n9.Exit\n");
        printf("\nEnter your choice?\n");
        scanf("\n%d",&choice);
        switch(choice)
        {
            case 1:
            insertion_beginning();
            break;
            case 2:
                insertion_last();
            break;
            case 3:
            insertion_specified();
            break;
```

```c
        case 4:
        deletion_beginning();
        break;
        case 5:
        deletion_last();
        break;
        case 6:
        deletion_specified();
        break;
        case 7:
        search();
        break;
        case 8:
        display();
        break;
        case 9:
        exit(0);
        break;
        default:
        printf("Please enter valid choice..");
      }
    }
}
void insertion_beginning()
{
   struct node *ptr;
   int item;
   ptr = (struct node *)malloc(sizeof(struct node));
   if(ptr == NULL)
```

```
    {
        printf("\nOVERFLOW");
    }
    else
    {
     printf("\nEnter Item value");
     scanf("%d",&item);

     if(head==NULL)
     {
        ptr->next = NULL;
        ptr->prev=NULL;
        ptr->data=item;
        head=ptr;
     }
     else
     {
        ptr->data=item;
        ptr->prev=NULL;
        ptr->next = head;
        head->prev=ptr;
        head=ptr;
     }
     printf("\nNode inserted\n");
    }

}
void insertion_last()
{
```

```c
    struct node *ptr,*temp;
    int item;
    ptr = (struct node *) malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter value");
        scanf("%d",&item);
        ptr->data=item;
        if(head == NULL)
        {
            ptr->next = NULL;
            ptr->prev = NULL;
            head = ptr;
        }
        else
        {
            temp = head;
            while(temp->next!=NULL)
            {
                temp = temp->next;
            }
            temp->next = ptr;
            ptr ->prev=temp;
            ptr->next = NULL;
        }
```

```c
    }
     printf("\nnode inserted\n");
   }
void insertion_specified()
{
   struct node *ptr,*temp;
   int item,loc,i;
   ptr = (struct node *)malloc(sizeof(struct node));
   if(ptr == NULL)
   {
      printf("\n OVERFLOW");
   }
   else
   {
      temp=head;
      printf("Enter the location");
      scanf("%d",&loc);
      for(i=0;i<loc;i++)
      {
         temp = temp->next;
         if(temp == NULL)
         {
            printf("\n There are less than %d elements", loc);
            return;
         }
      }
      printf("Enter value");
      scanf("%d",&item);
```

```c
        ptr->data = item;
        ptr->next = temp->next;
        ptr -> prev = temp;
        temp->next = ptr;
        temp->next->prev=ptr;
        printf("\nnode inserted\n");
    }
}
void deletion_beginning()
{
    struct node *ptr;
    if(head == NULL)
    {
        printf("\n UNDERFLOW");
    }
    else if(head->next == NULL)
    {
        head = NULL;
        free(head);
        printf("\nnode deleted\n");
    }
    else
    {
        ptr = head;
        head = head -> next;
        head -> prev = NULL;
        free(ptr);
        printf("\nnode deleted\n");
    }
```

```c
    }
    void deletion_last()
    {
        struct node *ptr;
        if(head == NULL)
        {
            printf("\n UNDERFLOW");
        }
        else if(head->next == NULL)
        {
            head = NULL;
            free(head);
            printf("\nnode deleted\n");
        }
        else
        {
            ptr = head;
            if(ptr->next != NULL)
            {
                ptr = ptr -> next;
            }
            ptr -> prev -> next = NULL;
            free(ptr);
            printf("\nnode deleted\n");
        }
    }
    void deletion_specified()
    {
```

```c
    struct node *ptr, *temp;
    int val;
    printf("\n Enter the data after which the node is to be deleted : ");
    scanf("%d", &val);
    ptr = head;
    while(ptr -> data != val)
    ptr = ptr -> next;
    if(ptr -> next == NULL)
    {
        printf("\nCan't delete\n");
    }
    else if(ptr -> next -> next == NULL)
    {
        ptr ->next = NULL;
    }
    else
    {
        temp = ptr -> next;
        ptr -> next = temp -> next;
        temp -> next -> prev = ptr;
        free(temp);
        printf("\nnode deleted\n");
    }
}
void display()
{
    struct node *ptr;
    printf("\n printing values...\n");
    ptr = head;
```

```c
        while(ptr != NULL)
        {
            printf("%d\n",ptr->data);
            ptr=ptr->next;
        }
    }
    void search()
    {
        struct node *ptr;
        int item,i=0,flag;
        ptr = head;
        if(ptr == NULL)
        {
            printf("\nEmpty List\n");
        }
        else
        {
            printf("\nEnter item which you want to search?\n");
            scanf("%d",&item);
            while (ptr!=NULL)
            {
                if(ptr->data == item)
                {
                    printf("\nitem found at location %d ",i+1);
                    flag=0;
                    break;
                }
                else
                {
```

```
            flag=1;

        }

        i++;

        ptr = ptr -> next;

    }

    if(flag==1)

    {

        printf("\nItem not found\n");

    }

  }


}
```

Output

```
*********Main Menu*********


Choose one option from the following list ...


==============================================


1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit
```

```
Enter your choice?

8


 printing values...


*********Main Menu*********


Choose one option from the following list ...


===========================================


1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit


Enter your choice?

1


Enter Item value12


Node inserted


*********Main Menu*********


Choose one option from the following list ...
```

```
=================================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit


Enter your choice?

1


Enter Item value123


Node inserted


*********Main Menu*********


Choose one option from the following list ...


=================================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search
```

8.Show

9.Exit

Enter your choice?

1

Enter Item value1234

Node inserted

*********Main Menu*********

Choose one option from the following list ...

===

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

 printing values...

1234

123

```
12


*********Main Menu*********


Choose one option from the following list ...


=============================================


1.Insert in begining
2.Insert at last
3.Insert at any random location
4.Delete from Beginning
5.Delete from last
6.Delete the node after the given data
7.Search
8.Show
9.Exit


Enter your choice?
2


Enter value89


node inserted


*********Main Menu*********


Choose one option from the following list ...


=============================================


1.Insert in begining
```

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

3

Enter the location1

Enter value12345

node inserted

*********Main Menu*********

Choose one option from the following list ...

==

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

 printing values...

1234

123

12345

12

89

*********Main Menu*********

Choose one option from the following list ...

==

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

4

node deleted

*********Main Menu*********

Choose one option from the following list ...

==

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

5

node deleted

*********Main Menu*********

Choose one option from the following list ...

==

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

```
7.Search

8.Show

9.Exit


Enter your choice?

8


 printing values...

123

12345


*********Main Menu*********


Choose one option from the following list ...


=============================================


1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit


Enter your choice?

6


 Enter the data after which the node is to be deleted : 123
```

```
*********Main Menu*********

Choose one option from the following list ...

==============================================

1.Insert in begining
2.Insert at last
3.Insert at any random location
4.Delete from Beginning
5.Delete from last
6.Delete the node after the given data
7.Search
8.Show
9.Exit

Enter your choice?
8

 printing values...
123

*********Main Menu*********

Choose one option from the following list ...

==============================================

1.Insert in begining
2.Insert at last
3.Insert at any random location
4.Delete from Beginning
```

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

7

Enter item which you want to search?

123

item found at location 1

*********Main Menu*********

Choose one option from the following list ...

==

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

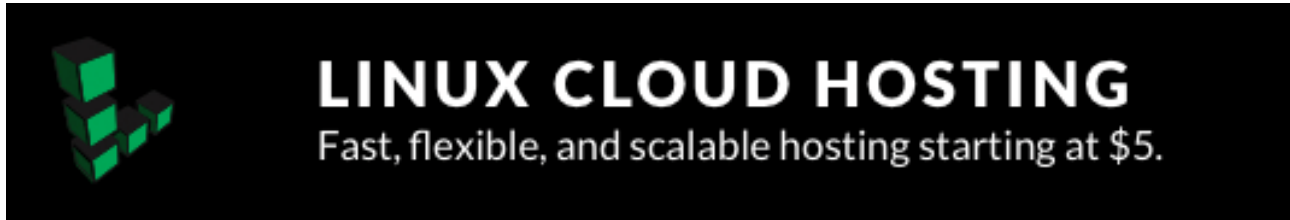Enter your choice?

6

```
 Enter the data after which the node is to be deleted : 123


Can't delete


*********Main Menu*********


Choose one option from the following list ...


=============================================


1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit


Enter your choice?

9


Exited..
```
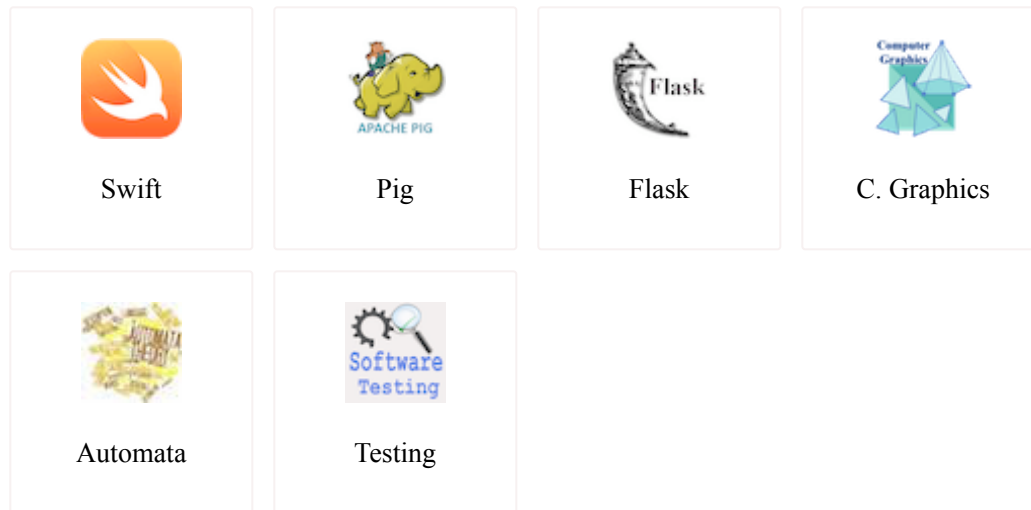
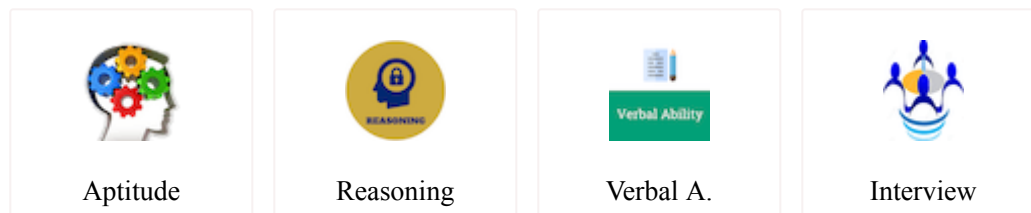← prev   next →

Please Share

Learn Latest Tutorials

Swift	Pig	Flask	C. Graphics

Automata	Testing

Preparation

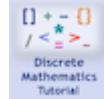|			
Aptitude	Reasoning	Verbal A.	Interview

B.Tech / MCA

DBMS

DS

DAA

OS

C. Network

Compiler D.

COA

D. Math.

E. Hacking

Web Tech.

Cyber Sec.

C

C++

Java

.Net

Python

Programs

Control S.