

Traversing in doubly linked list

Traversing is the most common operation in case of each data structure. For this purpose, copy the head pointer in any of the temporary pointer **ptr**.

```
Ptr = head
```

then, traverse through the list by using while loop. Keep shifting value of pointer variable **ptr** until we find the last node. The last node contains **null** in its next part.

```
while(ptr != NULL)
{
    printf("%d\n",ptr->data);
    ptr=ptr->next;
}
```

Although, traversing means visiting each node of the list once to perform some specific operation. Here, we are printing the data associated with each node of the list.

Algorithm

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



```
WRITE "UNDERFLOW"  
GOTO STEP 6  
[END OF IF]
```

- **Step 2:** Set PTR = HEAD
- **Step 3:** Repeat step 4 and 5 while PTR != NULL
- **Step 4:** Write PTR → data
- **Step 5:** PTR = PTR → next
- **Step 6:** Exit

C Function

```
#include<stdio.h>  
#include<stdlib.h>  
void create(int);  
int traverse();  
struct node  
{  
    int data;  
    struct node *next;  
    struct node *prev;  
};  
struct node *head;  
void main ()  
{  
    int choice,item;  
    do  
    {  
        printf("1.Append List\n2.Traverse\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:
        traverse();
        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
```



```
{

if(head==NULL)
{
    ptr->next = NULL;
    ptr->prev=NULL;
    ptr->data=item;
    head=ptr;
}
else
{
    ptr->data=item;printf("\nPress 0 to insert more ?\n");
    ptr->prev=NULL;
    ptr->next = head;
    head->prev=ptr;
    head=ptr;
}
printf("\nNode Inserted\n");
}

}

int traverse()
{
    struct node *ptr;
    if(head == NULL)
    {
        printf("\nEmpty List\n");
    }
}
```



```
    else
    {
        ptr = head;
        while(ptr != NULL)
        {
            printf("%d\n",ptr->data);
            ptr=ptr->next;
        }
    }
}
```

Output

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

```
Enter the item
23
```

Node Inserted

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

```
Enter the item
23
```

```
Press 0 to insert more ?
```



Node Inserted

1.Append List

2.Traverse

3.Exit

4.Enter your choice?1

Enter the item

90

Press 0 to insert more ?

Node Inserted

1.Append List

2.Traverse

3.Exit

4.Enter your choice?2

90

23

23

← 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.

