



Experiment 4.1

Student Name: Pranjal Kumar
Branch: CSE
Semester: 5th
Subject Name: DAA Lab

UID: 20BCS3504
Section/Group: 607 B
Date of Performance: 06/08/2022
Subject Code: 20CSP-312

1. Aim/Overview of the practical:-

Code to Insert and Delete an element at the beginning and at end in Doubly and Circular Linked List

2. Task to be done/ Which logistics used:-

Operation done in Insert and Delete an element at the beginning and at end in Doubly and Circular Linked List

3. Algorithm/Flowchart (For programming based labs):

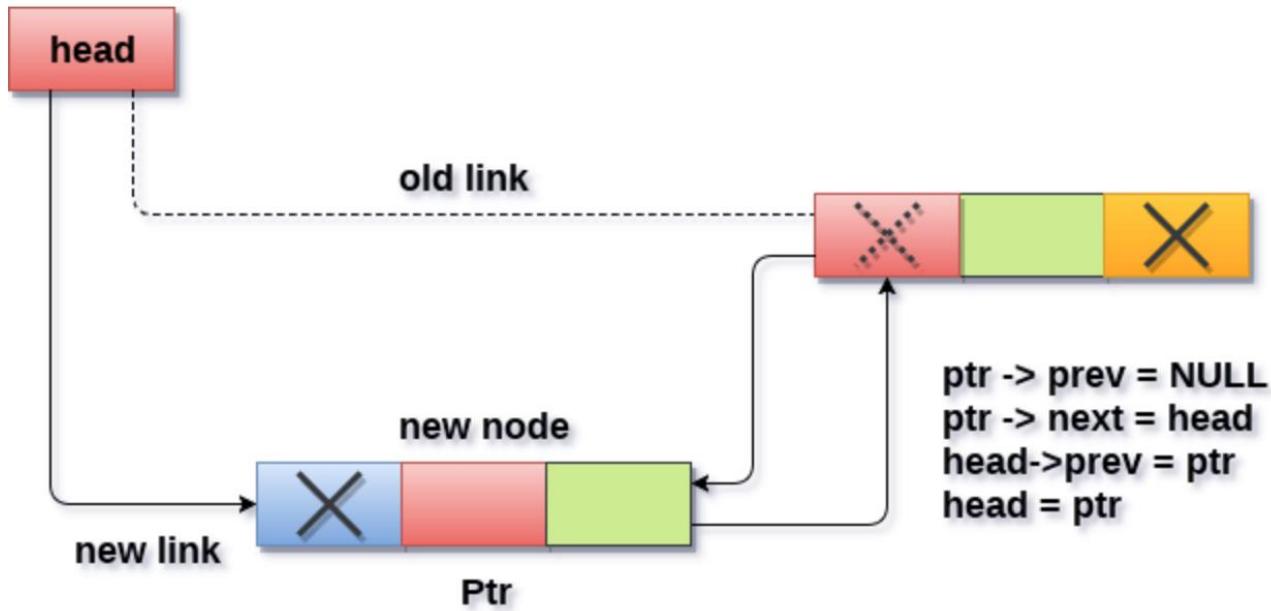
Doubly Linked List

INSERTION:-

A) Insertion in doubly linked list at beginning

- **Step 1:** IF ptr = NULL
 - Write OVERFLOW
 - Go to Step 9
 - [END OF IF]
- **Step 2:** SET NEW_NODE = ptr
- **Step 3:** SET ptr = ptr -> NEXT
- **Step 4:** SET NEW_NODE -> DATA = VAL
- **Step 5:** SET NEW_NODE -> PREV = NULL
- **Step 6:** SET NEW_NODE -> NEXT = START
- **Step 7:** SET head -> PREV = NEW_NODE
- **Step 8:** SET head = NEW_NODE

- **Step 9:** EXIT



B) Insertion in doubly linked list at the end

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

Write OVERFLOW

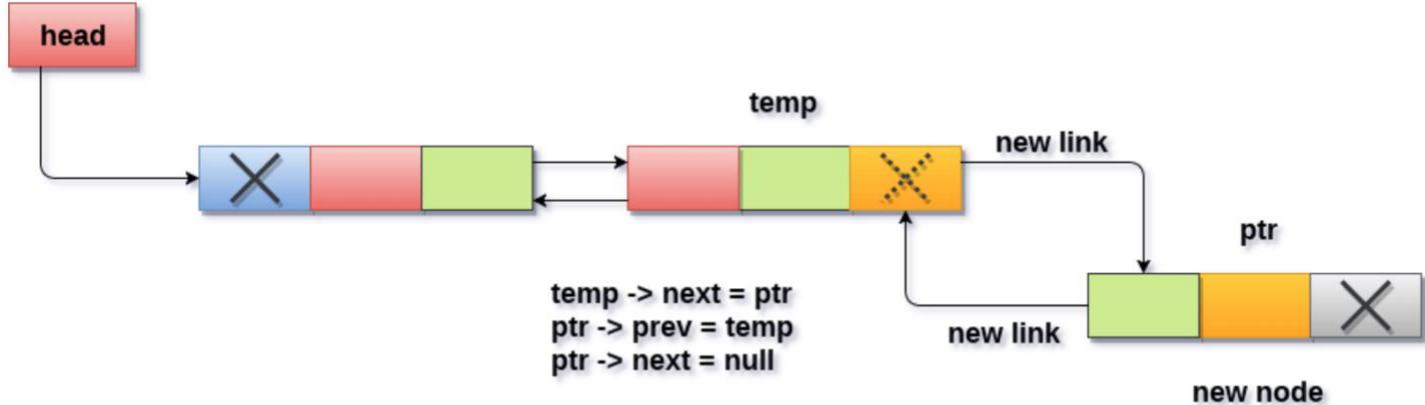
Go to Step 11

[END OF IF]

- **Step 2:** SET NEW_NODE = PTR
- **Step 3:** SET PTR = PTR -> NEXT
- **Step 4:** SET NEW_NODE -> DATA = VAL
- **Step 5:** SET NEW_NODE -> NEXT = NULL
- **Step 6:** SET TEMP = START
- **Step 7:** Repeat Step 8 while TEMP -> NEXT != NULL
- **Step 8:** SET TEMP = TEMP -> NEXT

[END OF LOOP]

- **Step 9:** SET TEMP -> NEXT = NEW_NODE
- **Step 10C:** SET NEW_NODE -> PREV = TEMP
- **Step 11:** EXIT



C) Insertion in doubly linked list after Specified node

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

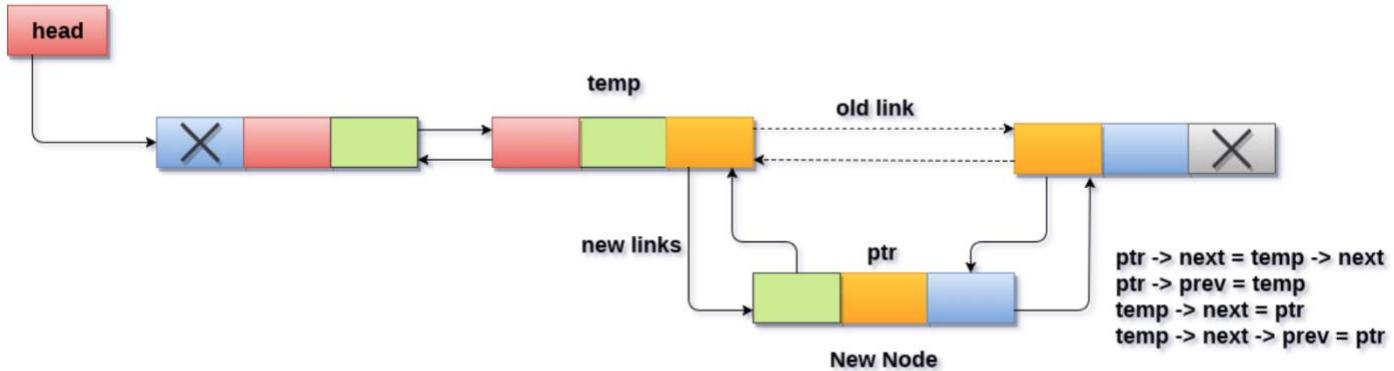
Write OVERFLOW
 Go to Step 15
 [END OF IF]
- **Step 2:** SET NEW_NODE = PTR
- **Step 3:** SET PTR = PTR -> NEXT
- **Step 4:** SET NEW_NODE -> DATA = VAL
- **Step 5:** SET TEMP = START
- **Step 6:** SET I = 0
- **Step 7:** REPEAT 8 to 10 until I<="" li=""">
- **Step 8:** SET TEMP = TEMP -> NEXT
- **STEP 9:** IF TEMP = NULL
- **STEP 10:** WRITE "LESS THAN DESIRED NO. OF ELEMENTS"

GOTO STEP 15

[END OF IF]

[END OF LOOP]

- **Step 11:** SET NEW_NODE -> NEXT = TEMP -> NEXT
- **Step 12:** SET NEW_NODE -> PREV = TEMP
- **Step 13 :** SET TEMP -> NEXT = NEW_NODE
- **Step 14:** SET TEMP -> NEXT -> PREV = NEW_NODE
- **Step 15:** EXIT



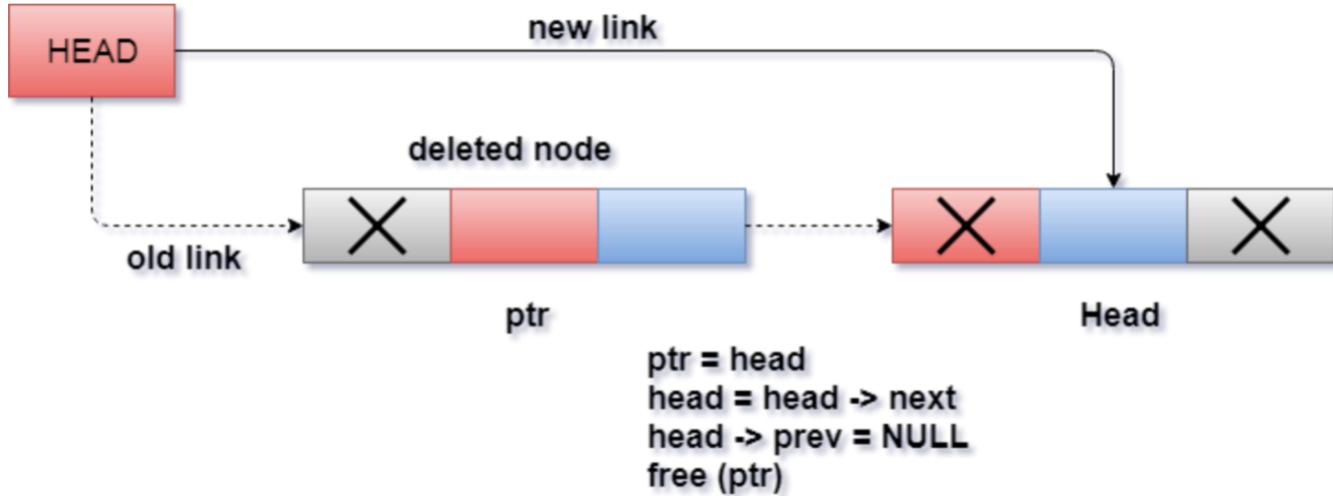
DELETION :

A) Deletion at beginning

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

WRITE UNDERFLOW
GOTO STEP 6

- **STEP 2:** SET PTR = HEAD
- **STEP 3:** SET HEAD = HEAD → NEXT
- **STEP 4:** SET HEAD → PREV = NULL
- **STEP 5:** FREE PTR
- **STEP 6:** EXIT



B) Deletion in doubly linked list at the end

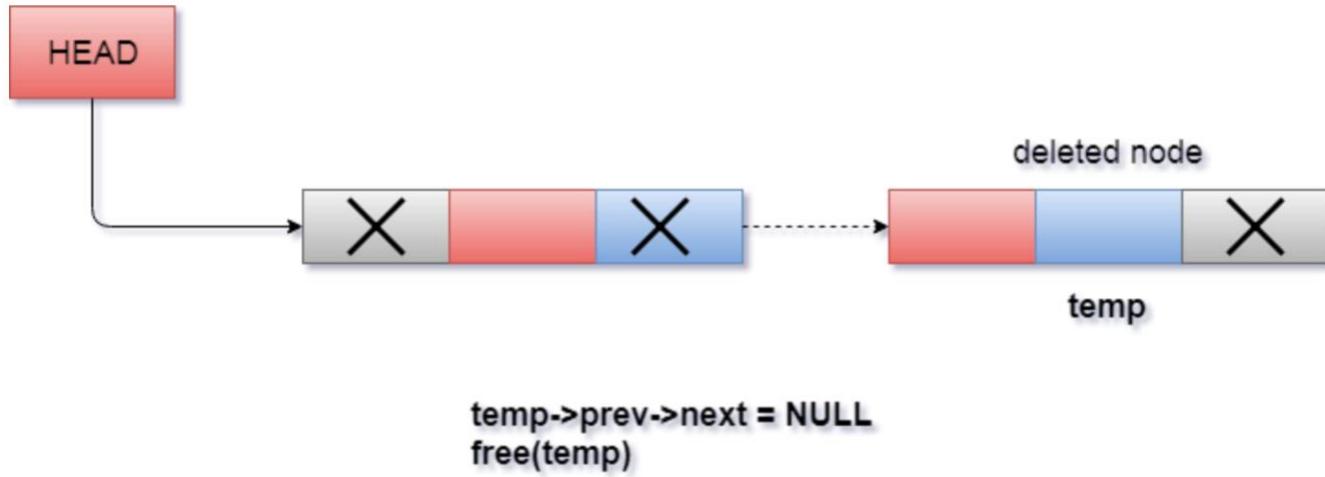
- ### o **Step 1:** IF HEAD = NULL

Write UNDERFLOW
Go to Step 7
[END OF IF]

- **Step 2:** SET TEMP = HEAD
 - **Step 3:** REPEAT STEP 4 WHILE TEMP->NEXT != NULL
 - **Step 4:** SET TEMP = TEMP->NEXT

[END OF LOOP]

- **Step 5:** SET TEMP ->PREV-> NEXT = NULL
 - **Step 6:** FREE TEMP
 - **Step 7:** EXIT

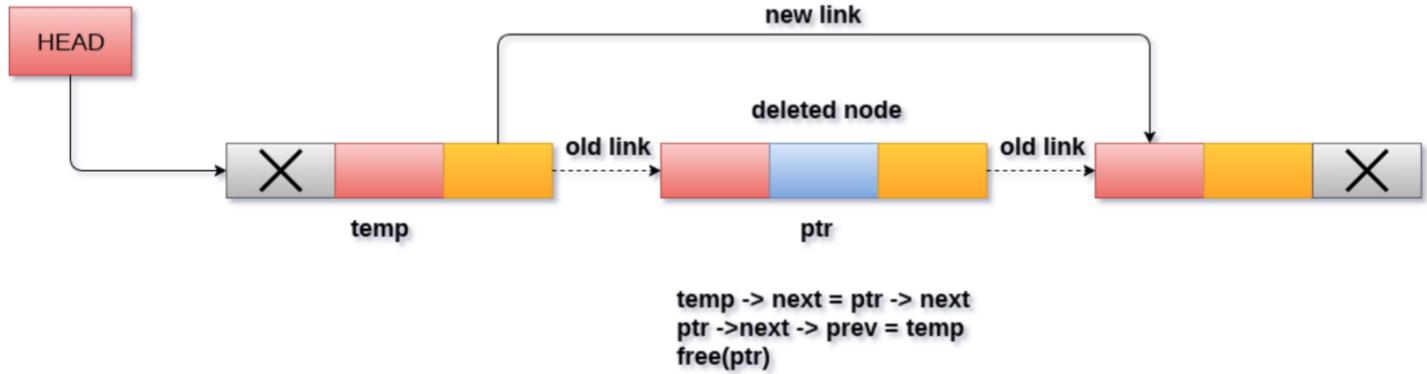


C) Deletion in doubly linked list after the specified node

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

Write UNDERFLOW
Go to Step 9
[END OF IF]

- o **Step 2:** SET TEMP = HEAD
- o **Step 3:** Repeat Step 4 while TEMP -> DATA != ITEM
- o **Step 4:** SET TEMP = TEMP -> NEXT
- [END OF LOOP]
- o **Step 5:** SET PTR = TEMP -> NEXT
- o **Step 6:** SET TEMP -> NEXT = PTR -> NEXT
- o **Step 7:** SET PTR -> NEXT -> PREV = TEMP
- o **Step 8:** FREE PTR
- o **Step 9:** EXIT



Circular Linked List

Insertion:-

A) Insertion into circular singly linked list at beginning

- o **Step 1:** IF PTR = NULL

 Write OVERFLOW

 Go to Step 11

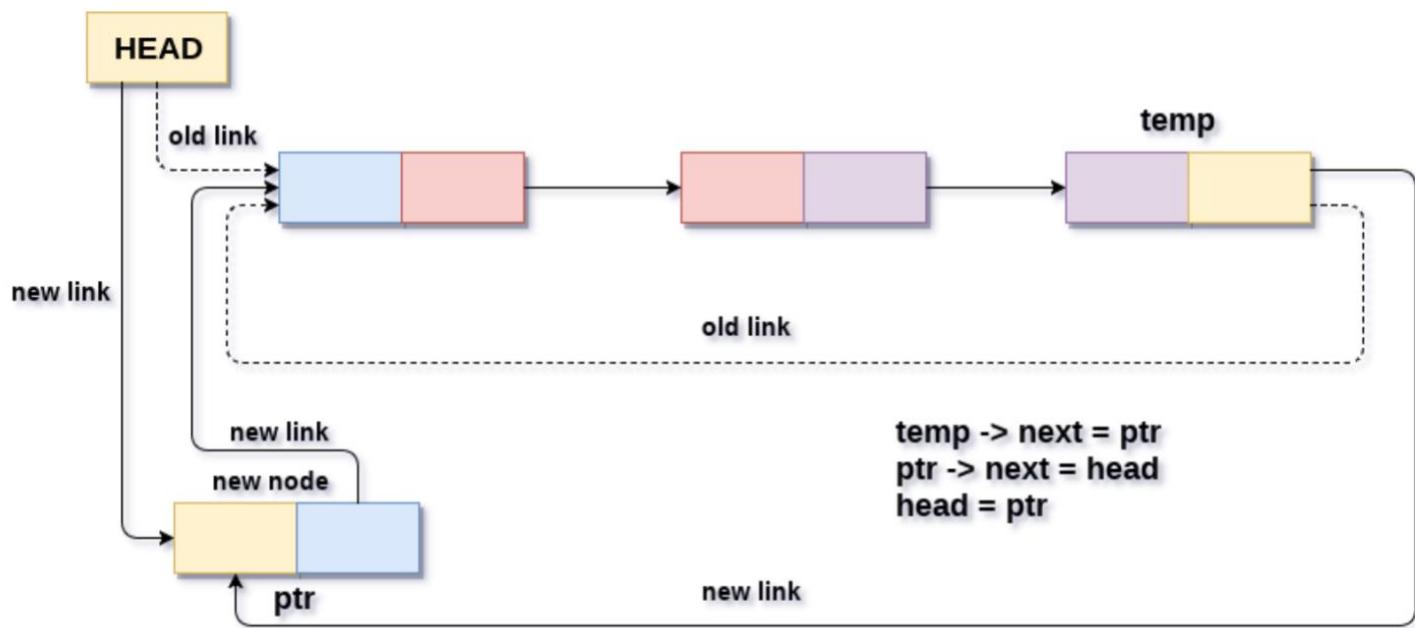
 [END OF IF]

- o **Step 2:** SET NEW_NODE = PTR
- o **Step 3:** SET PTR = PTR -> NEXT
- o **Step 4:** SET NEW_NODE -> DATA = VAL
- o **Step 5:** SET TEMP = HEAD
- o **Step 6:** Repeat Step 8 while TEMP -> NEXT != HEAD
- o **Step 7:** SET TEMP = TEMP -> NEXT

[END OF LOOP]

- o **Step 8:** SET NEW_NODE -> NEXT = HEAD
- o **Step 9:** SET TEMP -> NEXT = NEW_NODE
- o **Step 10:** SET HEAD = NEW_NODE

- **Step 11: EXIT**



B) Insertion into circular singly linked list at the end

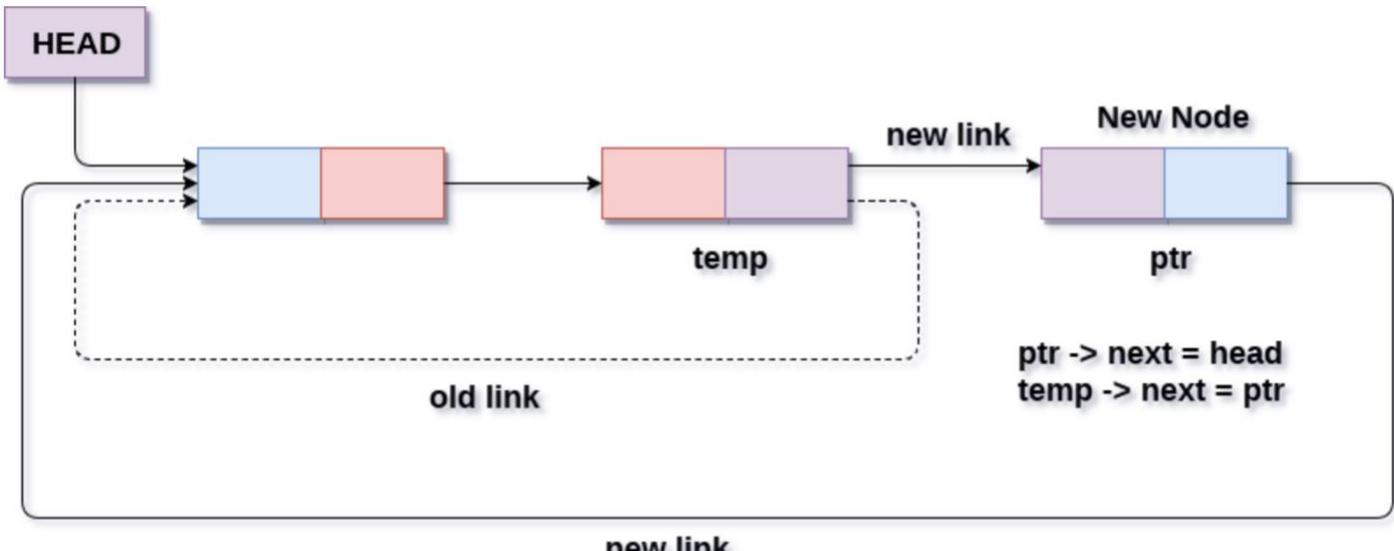
- **Step 1:** IF PTR = NULL

 Write OVERFLOW
 Go to Step 1
 [END OF IF]

- **Step 2:** SET NEW_NODE = PTR
- **Step 3:** SET PTR = PTR -> NEXT
- **Step 4:** SET NEW_NODE -> DATA = VAL
- **Step 5:** SET NEW_NODE -> NEXT = HEAD
- **Step 6:** SET TEMP = HEAD
- **Step 7:** Repeat Step 8 while TEMP -> NEXT != HEAD
- **Step 8:** SET TEMP = TEMP -> NEXT

[END OF LOOP]

- **Step 9:** SET TEMP -> NEXT = NEW_NODE
- **Step 10:** EXIT

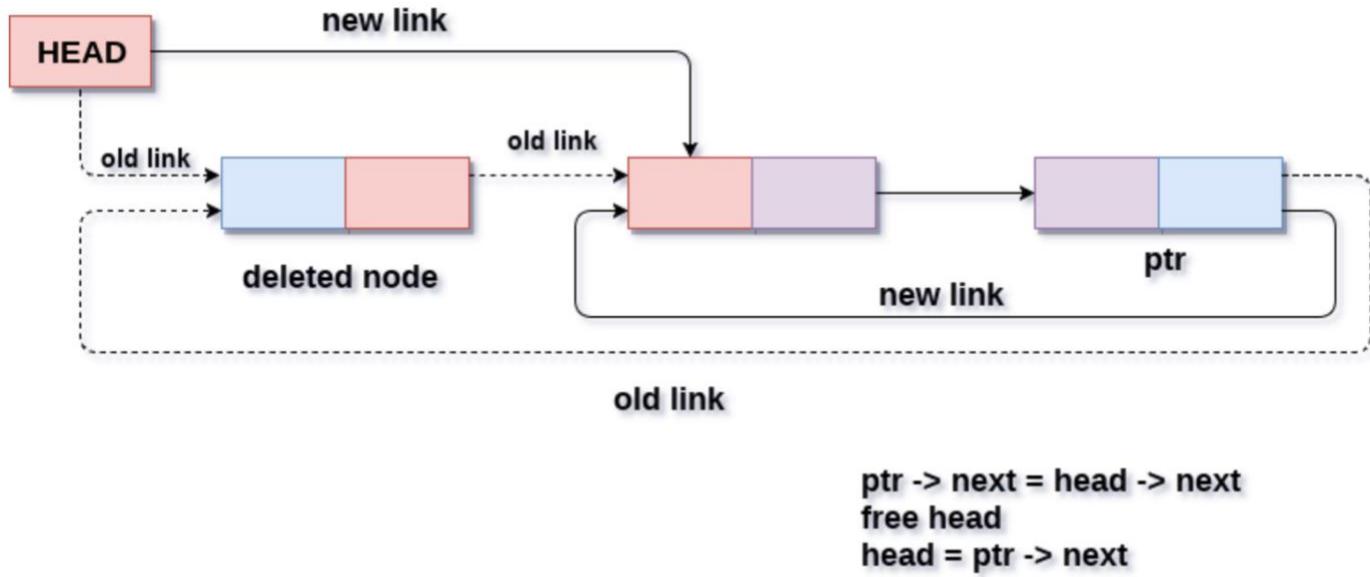


DELETION:-

A) Deletion in circular singly linked list at beginning

- **Step 1:** IF HEAD = NULL
 - Write UNDERFLOW
 - Go to Step 8
 - [END OF IF]
- **Step 2:** SET PTR = HEAD
- **Step 3:** Repeat Step 4 while PTR → NEXT != HEAD
- **Step 4:** SET PTR = PTR → next
 - [END OF LOOP]
- **Step 5:** SET PTR → NEXT = HEAD → NEXT
- **Step 6:** FREE HEAD

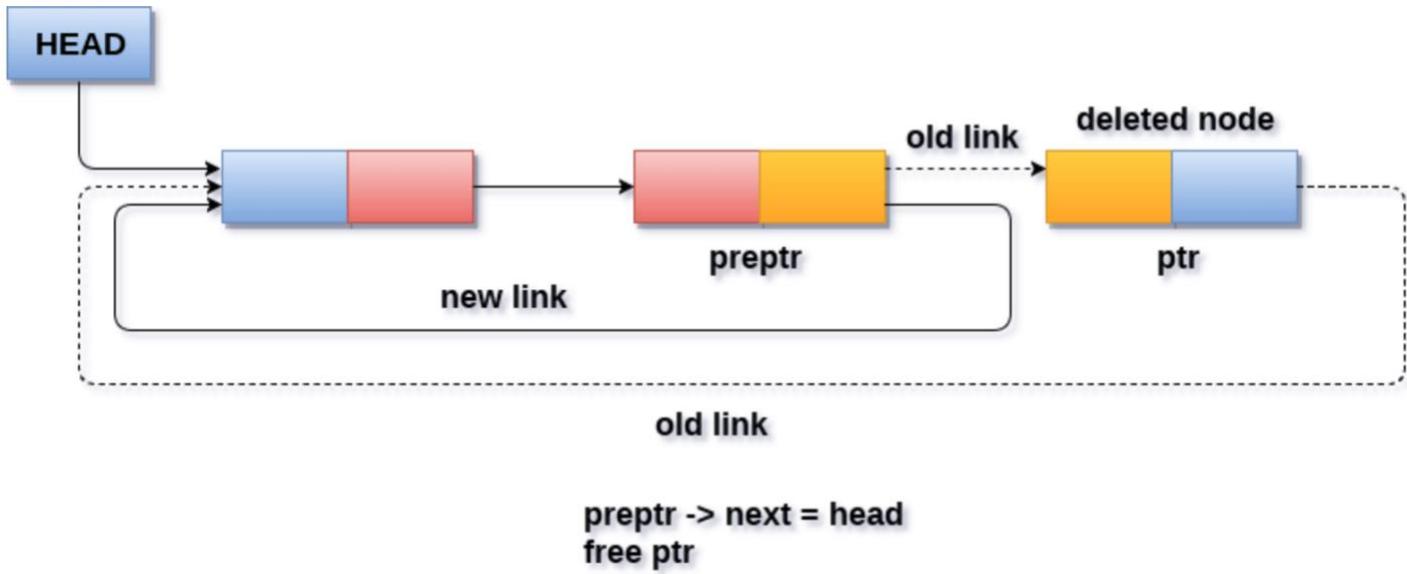
- **Step 7:** SET HEAD = PTR → NEXT
- **Step 8:** EXIT



B) Deletion in Circular singly linked list at the end

- **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 != HEAD
- **Step 4:** SET PREPTR = PTR
- **Step 5:** SET PTR = PTR → NEXT
- [END OF LOOP]
- **Step 6:** SET PREPTR → NEXT = HEAD
- **Step 7:** FREE PTR

- o **Step 8: EXIT**



4. Steps for experiment/practical/Code:-

Doubly Linked List

INSERTION:-

A) Insertion in doubly linked list at beginning

```
#include<stdio.h>
#include<stdlib.h>
void insertbeginning(int);
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};
struct node *head;
void main ()
{
    int choice,item;
    do
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
{  
    printf("\nEnter the item which you want to insert?\n");  
    scanf("%d",&item);  
    insertbeginning(item);  
    printf("\nPress 0 to insert more ?\n");  
    scanf("%d",&choice);  
}while(choice == 0);  
}  
void insertbeginning(int item)  
{  
  
    struct node *ptr = (struct node *)malloc(sizeof(struct node));  
    if(ptr == NULL)  
    {  
        printf("\nOVERFLOW");  
    }  
    else  
    {  
        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;  
        }  
    }  
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void insertbeginning(int);
4 struct node
5 {
6     int data;
7     struct node *next;
8     struct node *prev;
9 };
10 struct node *head;
11 void main ()
12 {
13     int choice,item;
14     do
15     {
16         printf("\nEnter the item which you want to insert?\n");
17         scanf("%d",&item);
18         insertbeginning(item);
19         printf("\nPress 0 to insert more ?\n");
20         scanf("%d",&choice);
21     }while(choice == 0);
22 }
23 void insertbeginning(int item)
24 {
25
26     struct node *ptr = (struct node *)malloc(sizeof(struct node));
27     if(ptr == NULL)
28     {
29         printf("\nOVERFLOW");
30     }
31     else
32     {
33         if(head==NULL)
34         {
35             ptr->next = NULL;
36             ptr->prev=NULL;
37             ptr->data=item;
38             head=ptr;
39         }
40     else
41     {
42         ptr->data=item;
43         ptr->prev=NULL;
44         ptr->next = head;
45         head->prev=ptr;
46         head=ptr;
47     }
48 }
49 }
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

B) Insertion in doubly linked list at the end

```
#include<stdio.h>
#include<stdlib.h>
void insertlast(int);
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};
struct node *head;
void main ()
{
    int choice,item;
    do
    {
        printf("\nEnter the item which you want to insert?\n");
        scanf("%d",&item);
        insertlast(item);
        printf("\nPress 0 to insert more ?\n");
        scanf("%d",&choice);
    }while(choice == 0);
}
void insertlast(int item)
{

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



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
ptr->prev = NULL;
head = ptr;
}
else
{
    temp = head;
    while(temp->next!=NULL)
    {
        temp = temp->next;
    }
    temp->next = ptr;
    ptr ->prev=temp;
    ptr->next = NULL;
}
printf("\nNode Inserted\n");
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void insertlast(int);
4 struct node
5 {
6     int data;
7     struct node *next;
8     struct node *prev;
9 };
10 struct node *head;
11 void main ()
12 {
13     int choice,item;
14     do
15     {
16         printf("\nEnter the item which you want to insert?\n");
17         scanf("%d",&item);
18         insertlast(item);
19         printf("\nPress 0 to insert more ?\n");
20         scanf("%d",&choice);
21     }while(choice == 0);
22 }
23 void insertlast(int item)
24 {
25
26     struct node *ptr = (struct node *) malloc(sizeof(struct node));
27     struct node *temp;
28     if(ptr == NULL)
29     {
30         printf("\nOVERFLOW");
31     }
32 }
33 else
```



```
34    {
35        ptr->data=item;
36        if(head == NULL)
37        {
38            ptr->next = NULL;
39            ptr->prev = NULL;
40            head = ptr;
41        }
42        else
43        {
44            temp = head;
45            while(temp->next!=NULL)
46            {
47                temp = temp->next;
48            }
49            temp->next = ptr;
50            ptr ->prev=temp;
51            ptr->next = NULL;
52        }
53        printf("\nNode Inserted\n");
54    }
55 }
```

C) Insertion in doubly linked list after Specified node

```
#include<stdio.h>
#include<stdlib.h>
void insert_specified(int);
void create(int);
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};
struct node *head;
void main ()
{
    int choice,item,loc;
    do
    {
        printf("\nEnter the item which you want to insert?\n");

```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
scanf("%d",&item);
if(head == NULL)
{
    create(item);
}
else
{
    insert_specified(item);
}
printf("\nPress 0 to insert more ?\n");
scanf("%d",&choice);
}while(choice == 0);
}

void create(int item)
{
    struct node *ptr = (struct node *)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    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");
    }
}

void insert_specified(int item)
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
{  
    struct node *ptr = (struct node *)malloc(sizeof(struct node));  
    struct node *temp;  
    int i, loc;  
    if(ptr == NULL)  
    {  
        printf("\n OVERFLOW");  
    }  
    else  
    {  
        printf("\nEnter the location\n");  
        scanf("%d",&loc);  
        temp=head;  
        for(i=0;i<loc;i++)  
        {  
            temp = temp->next;  
            if(temp == NULL)  
            {  
                printf("\ncan't insert\n");  
                return;  
            }  
        }  
        ptr->data = item;  
        ptr->next = temp->next;  
        ptr -> prev = temp;  
        temp->next = ptr;  
        temp->next->prev=ptr;  
        printf("Node Inserted\n");  
    }  
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void insert_specified(int);
4 void create(int);
5 struct node
6 {
7     int data;
8     struct node *next;
9     struct node *prev;
10 };
11 struct node *head;
12 void main ()
13 {
14     int choice,item,loc;
15     do
16     {
17         printf("\nEnter the item which you want to insert?\n");
18         scanf("%d",&item);
19         if(head == NULL)
20         {
21             create(item);
22         }
23         else
24         {
25             insert_specified(item);
26         }
27         printf("\nPress 0 to insert more ?\n");
28         scanf("%d",&choice);
29     }while(choice == 0);
30 }
31 void create(int item)
32 {
33     struct node *ptr = (struct node *)malloc(sizeof(struct node));
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
34     if(ptr == NULL)
35     {
36         printf("\nOVERFLOW");
37     }
38     else
39     {
40
41
42     if(head==NULL)
43     {
44         ptr->next = NULL;
45         ptr->prev=NULL;
46         ptr->data=item;
47         head=ptr;
48     }
49     else
50     {
51         ptr->data=item;printf("\nPress 0 to insert more ?\n");
52         ptr->prev=NULL;
53         ptr->next = head;
54         head->prev=ptr;
55         head=ptr;
56     }
57     printf("\nNode Inserted\n");
58 }
59
60 }
61 void insert_specified(int item)
62 {
63     struct node *ptr = (struct node *)malloc(sizeof(struct node));
64     struct node *temp;
65     int i, loc;
66     if(ptr == NULL)
```



```
67      {
68          printf("\n OVERFLOW");
69      }
70  else
71  {
72      printf("\nEnter the location\n");
73      scanf("%d",&loc);
74      temp=head;
75      for(i=0;i<loc;i++)
76      {
77          temp = temp->next;
78          if(temp == NULL)
79          {
80              printf("\ncan't insert\n");
81              return;
82          }
83      }
84      ptr->data = item;
85      ptr->next = temp->next;
86      ptr -> prev = temp;
87      temp->next = ptr;
88      temp->next->prev=ptr;
89      printf("Node Inserted\n");
90  }
91 }
```

DELETION :

A) Deletion at beginning

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



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
void main ()
{
    int choice,item;
    do
    {
        printf("1.Append List\n2.Delete node from beginning\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:
                beginning_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
    {

        if(head==NULL)
        {
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
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");
}
}

void beginning_delete()
{
    struct node *ptr;
    if(head == NULL)
    {
        printf("\n UNDERFLOW\n");
    }
    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");
    }
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void create(int);
4 void beginning_delete();
5 struct node
6 {
7     int data;
8     struct node *next;
9     struct node *prev;
10 };
11 struct node *head;
12 void main ()
13 {
14     int choice,item;
15     do
16     {
17         printf("1.Append List\n2.Delete node from beginning\n3.Exit\n4.Enter your choice?");
18         scanf("%d",&choice);
19         switch(choice)
20         {
21             case 1:
22                 printf("\nEnter the item\n");
23                 scanf("%d",&item);
24                 create(item);
25                 break;
26             case 2:
27                 beginning_delete();
28                 break;
29             case 3:
30                 exit(0);
31                 break;
32             default:
33                 printf("\nPlease enter valid choice\n");
34         }
35     }
36     }while(choice != 3);
37 }
38 void create(int item)
39 {
40
41     struct node *ptr = (struct node *)malloc(sizeof(struct node));
42     if(ptr == NULL)
43     {
44         printf("\nOVERFLOW\n");
45     }
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
46     else
47     {
48
49
50     if(head==NULL)
51     {
52         ptr->next = NULL;
53         ptr->prev=NULL;
54         ptr->data=item;
55         head=ptr;
56     }
57     else
58     {
59         ptr->data=item;printf("\nPress 0 to insert more ?\n");
60         ptr->prev=NULL;
61         ptr->next = head;
62         head->prev=ptr;
63         head=ptr;
64     }
65     printf("\nNode Inserted\n");
66 }
67 }
68 void beginning_delete()
69 {
70     struct node *ptr;
71     if(head == NULL)
72     {
73         printf("\n UNDERFLOW\n");
74     }
75     else if(head->next == NULL)
76     {
77         head = NULL;
78         free(head);
79         printf("\nNode Deleted\n");
80     }
81     else
82     {
83         ptr = head;
84         head = head -> next;
85         head -> prev = NULL;
86         free(ptr);
87         printf("\nNode Deleted\n");
88     }
89 }
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

B) Deletion in doubly linked list at the end

```
#include<stdio.h>
#include<stdlib.h>
void create(int);
void last_delete();
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};
struct node *head;
void main ()
{
    int choice,item;
    do
    {
        printf("1.Append List\n2.Delete node from end\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:
                last_delete();
                break;
            case 3:
                exit(0);
                break;
            default:
                printf("\nPlease enter valid choice\n");
        }
    }while(choice != 3);
}
void create(int item)
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

{

```
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;
        ptr->prev=NULL;
        ptr->next = head;
        head->prev=ptr;
        head=ptr;
    }
    printf("\nNode Inserted\n");
}
}
void last_delete()
{
    struct node *ptr;
    if(head == NULL)
    {
        printf("\n UNDERFLOW\n");
    }
    else if(head->next == NULL)
    {
        head = NULL;
        free(head);
    }
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
    printf("\nNode Deleted\n");
}
else
{
    ptr = head;
    if(ptr->next != NULL)
    {
        ptr = ptr -> next;
    }
    ptr -> prev -> next = NULL;
    free(ptr);
    printf("\nNode Deleted\n");
}
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void create(int);
4 void last_delete();
5 struct node
6 {
7     int data;
8     struct node *next;
9     struct node *prev;
10 };
11 struct node *head;
12 void main ()
13 {
14     int choice,item;
15     do
16     {
17         printf("1.Append List\n2.Delete node from end\n3.Exit\n4.Enter your choice?");
18         scanf("%d",&choice);
19         switch(choice)
20         {
21             case 1:
22                 printf("\nEnter the item\n");
23                 scanf("%d",&item);
24                 create(item);
25                 break;
26             case 2:
27                 last_delete();
28                 break;
29             case 3:
30                 exit(0);
31                 break;
32             default:
33                 printf("\nPlease enter valid choice\n");
34             }
35         }
36     }while(choice != 3);
37 }
38 void create(int item)
39 {
40
41     struct node *ptr = (struct node *)malloc(sizeof(struct node));
42     if(ptr == NULL)
43     {
44         printf("\nOVERFLOW\n");
45     }
46     else
47     {
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
48     if(head==NULL)
49     {
50         ptr->next = NULL;
51         ptr->prev=NULL;
52         ptr->data=item;
53         head=ptr;
54     }
55     else
56     {
57         ptr->data=item;
58         ptr->prev=NULL;
59         ptr->next = head;
60         head->prev=ptr;
61         head=ptr;
62     }
63     printf("\nNode Inserted\n");
64 }
65 }
66 void last_delete()
67 {
68     struct node *ptr;
69     if(head == NULL)
70     {
71         printf("\n UNDERFLOW\n");
72     }
73     else if(head->next == NULL)
74     {
75         head = NULL;
76         free(head);
77         printf("\nNode Deleted\n");
78     }
79     else
80     {
81         ptr = head;
82         if(ptr->next != NULL)
83         {
84             ptr = ptr -> next;
85         }
86         ptr -> prev -> next = NULL;
87         free(ptr);
88         printf("\nNode Deleted\n");
89     }
90 }
```

C) Deletion in doubly linked list after the specified node



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

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



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
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;
    ptr->prev=NULL;
    ptr->next = head;
    head->prev=ptr;
    head=ptr;
}
printf("\nNode Inserted\n");
}

void delete_specified( )
{
    struct node *ptr, *temp;
    int val;
    printf("Enter the value");
    scanf("%d",&val);
    temp = head;
    while(temp -> data != val)
        temp = temp -> next;
    if(temp -> next == NULL)
    {
        printf("\nCan't delete\n");
    }
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
else if(temp -> next -> next == NULL)
{
    temp ->next = NULL;
    printf("\nNode Deleted\n");
}
else
{
    ptr = temp -> next;
    temp -> next = ptr -> next;
    ptr -> next -> prev = temp;
    free(ptr);
    printf("\nNode Deleted\n");
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void create(int);
4 void delete_specified();
5 struct node
6 {
7     int data;
8     struct node *next;
9     struct node *prev;
10 };
11 struct node *head;
12 void main ()
13 {
14     int choice,item;
15     do
16     {
17         printf("1.Append List\n2.Delete node\n3.Exit\n4.Enter your choice?");
18         scanf("%d",&choice);
19         switch(choice)
20         {
21             case 1:
22                 printf("\nEnter the item\n");
23                 scanf("%d",&item);
24                 create(item);
25                 break;
26             case 2:
27                 delete_specified();
28                 break;
29             case 3:
30                 exit(0);
31                 break;
32             default:
33                 printf("\nPlease enter valid choice\n");
34         }
35     }
36 }while(choice != 3);
37 }
38 void create(int item)
39 {
40     struct node *ptr = (struct node *)malloc(sizeof(struct node));
41     if(ptr == NULL)
42     {
43         printf("\nOVERFLOW\n");
44     }
45     else
46     {
47         if(head==NULL)
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
48  {
49      ptr->next = NULL;
50      ptr->prev=NULL;
51      ptr->data=item;
52      head=ptr;
53  }
54  else
55  {
56      ptr->data=item;
57      ptr->prev=NULL;
58      ptr->next = head;
59      head->prev=ptr;
60      head=ptr;
61  }
62  printf("\nNode Inserted\n");
63 }
64
65 }
66 void delete_specified( )
67 {
68     struct node *ptr, *temp;
69     int val;
70     printf("Enter the value");
71     scanf("%d",&val);
72     temp = head;
73     while(temp -> data != val)
74     temp = temp -> next;
75     if(temp -> next == NULL)
76     {
77         printf("\nCan't delete\n");
78     }
79     else if(temp -> next -> next == NULL)
80     {
81         temp ->next = NULL;
82         printf("\nNode Deleted\n");
83     }
84     else
85     {
86         ptr = temp -> next;
87         temp -> next = ptr -> next;
88         ptr -> next -> prev = temp;
89         free(ptr);
90         printf("\nNode Deleted\n");
91     }
92 }
```

Circular Linked List



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

Insertion:-

A) Insertion at beginning

```
#include<stdio.h>
#include<stdlib.h>
void beg_insert(int);
struct node
{
    int data;
    struct node *next;
};
struct node *head;
void main ()
{
    int choice,item;
    do
    {
        printf("\nEnter the item which you want to insert?\n");
        scanf("%d",&item);
        beg_insert(item);
        printf("\nPress 0 to insert more ?\n");
        scanf("%d",&choice);
    }while(choice == 0);
}
void beg_insert(int item)
{
    struct node *ptr = (struct node *)malloc(sizeof(struct node));
    struct node *temp;
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        ptr -> data = item;
        if(head == NULL)
        {
            head = ptr;
            ptr -> next = head;
        }
        else
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
{  
    temp = head;  
    while(temp->next != head)  
        temp = temp->next;  
    ptr->next = head;  
    temp -> next = ptr;  
    head = ptr;  
}  
printf("\nNode Inserted\n");  
}  
}
```

```
1 #include<stdio.h>  
2 #include<stdlib.h>  
3 void beg_insert(int);  
4 struct node  
5 {  
    int data;  
    struct node *next;  
8 };  
9 struct node *head;  
10 void main ()  
11 {  
    int choice,item;  
13    do  
14    {  
15        printf("\nEnter the item which you want to insert?\n");  
16        scanf("%d",&item);  
17        beg_insert(item);  
18        printf("\nPress 0 to insert more ?\n");  
19        scanf("%d",&choice);  
20    }while(choice == 0);  
21 }  
22 void beg_insert(int item)  
23 {  
24  
25    struct node *ptr = (struct node *)malloc(sizeof(struct node));  
26    struct node *temp;  
27    if(ptr == NULL)  
28    {  
29        printf("\nOVERFLOW");  
30    }  
31    else  
32    {  
33        ptr -> data = item;
```



```
34     {
35         head = ptr;
36         ptr -> next = head;
37     }
38     else
39     {
40         temp = head;
41         while(temp->next != head)
42             temp = temp->next;
43         ptr->next = head;
44         temp -> next = ptr;
45         head = ptr;
46     }
47     printf("\nNode Inserted\n");
48 }
49 }
```

B) Insertion at the end FUNCTION

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



```
    }
}

}

1 void lastinsert(struct node*ptr, struct node *temp, int item)
2 {
3     ptr = (struct node *)malloc(sizeof(struct node));
4     if(ptr == NULL)
5     {
6         printf("\nOVERFLOW\n");
7     }
8     else
9     {
10        ptr->data = item;
11        if(head == NULL)
12        {
13            head = ptr;
14            ptr -> next = head;
15        }
16        else
17        {
18            temp = head;
19            while(temp -> next != head)
20            {
21                temp = temp -> next;
22            }
23            temp -> next = ptr;
24            ptr -> next = head;
25        }
26    }
27
28 }
```

DELETION:-

A) Deletion in circular singly linked list at beginning

```
#include<stdio.h>
```

```
#include<stdlib.h>
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



```
void create(int);

void beg_delete();

struct node

{

    int data;

    struct node *next;

};

struct node *head;

void main ()

{

    int choice,item;

    do

    {

        printf("1.Append List\n2.Delete Node from beginning\n3.Exit\n4.Enter your choice?");

        scanf("%d",&choice);

        switch(choice)

        {

            case 1:

                printf("\nEnter the item\n");

                scanf("%d",&item);

                create(item);

    
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
break;  
  
case 2:  
  
beg_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));  
  
struct node *temp;  
  
if(ptr == NULL)  

```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
}
```

else

```
{
```

```
    ptr -> data = item;
```

```
    if(head == NULL)
```

```
    {
```

```
        head = ptr;
```

```
        ptr -> next = head;
```

```
    }
```

else

```
{
```

```
    temp = head;
```

```
    while(temp->next != head)
```

```
        temp = temp->next;
```

```
    ptr->next = head;
```

```
    temp -> next = ptr;
```

```
    head = ptr;
```

```
}
```

```
printf("\nNode Inserted\n");
```

```
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

}

```
void beg_delete()
```

```
{
```

```
    struct node *ptr;
```

```
    if(head == NULL)
```

```
{
```

```
        printf("\nUNDERFLOW\n");
```

```
}
```

```
else if(head->next == head)
```

```
{
```

```
    head = NULL;
```

```
    free(head);
```

```
    printf("\nNode Deleted\n");
```

```
}
```

```
else
```

```
{
```

```
    ptr = head;
```

```
    while(ptr -> next != head)
```

```
        ptr = ptr -> next;
```

```
    ptr->next = head->next;
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
free(head);  
  
head = ptr->next;  
  
printf("\nNode Deleted\n");  
  
}  
  
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



```
1 #include<stdio.h>
2 #include<stdlib.h>
3 void create(int);
4 void beg_delete();
5 struct node
6 {
7     int data;
8     struct node *next;
9 };
10 struct node *head;
11 void main ()
12 {
13     int choice,item;
14     do
15     {
16         printf("1.Append List\n2.Delete Node from beginning\n3.Exit\n4.Enter your choice?");
17         scanf("%d",&choice);
18         switch(choice)
19         {
20             case 1:
21                 printf("\nEnter the item\n");
22                 scanf("%d",&item);
23                 create(item);
24                 break;
25             case 2:
26                 beg_delete();
27                 break;
28             case 3:
29                 exit(0);
30                 break;
31             default:
32                 printf("\nPlease Enter valid choice\n");
33         }
34     }while(choice != 3);
35 }
36 void create(int item)
37 {
38     struct node *ptr = (struct node *)malloc(sizeof(struct node));
39     struct node *temp;
40     if(ptr == NULL)
41     {
42         printf("\nOVERFLOW");
43     }
44     else
45     {
46         ptr -> data = item;
47         if(head == NULL)
48         {
```



```
49         head = ptr;
50         ptr -> next = head;
51     }
52     else
53     {
54         temp = head;
55         while(temp->next != head)
56             temp = temp->next;
57         ptr->next = head;
58         temp -> next = ptr;
59         head = ptr;
60     }
61     printf("\nNode Inserted\n");
62 }
63 }
64 void beg_delete()
65 {
66     struct node *ptr;
67     if(head == NULL)
68     {
69         printf("\nUNDERFLOW\n");
70     }
71     else if(head->next == head)
72     {
73         head = NULL;
74         free(head);
75         printf("\nNode Deleted\n");
76     }
77     else
78     {
79         ptr = head;
80         while(ptr -> next != head)
81             ptr = ptr -> next;
82             ptr->next = head->next;
83             free(head);
84             head = ptr->next;
85             printf("\nNode Deleted\n");
86     }
87 }
```

B) Deletion in Circular singly linked list at the end

```
#include<stdio.h>
```

```
#include<stdlib.h>
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



```
void create(int);

void last_delete();

struct node

{

    int data;

    struct node *next;

};

struct node *head;

void main ()

{

    int choice,item;

    do

    {

        printf("1.Append List\n2.Delete Node from end\n3.Exit\n4.Enter your choice?");

        scanf("%d",&choice);

        switch(choice)

        {

            case 1:

                printf("\nEnter the item\n");

                scanf("%d",&item);

                create(item);

    
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
break;

case 2:

last_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));

struct node *temp;

if(ptr == NULL)

{

printf("\nOVERFLOW\n");

}

}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC
GRADE A+
ACCREDITED UNIVERSITY

```
else
{
    ptr -> data = item;
    if(head == NULL)
    {
        head = ptr;
        ptr -> next = head;
    }
    else
    {
        temp = head;
        while(temp->next != head)
            temp = temp->next;
        ptr->next = head;
        temp -> next = ptr;
        head = ptr;
    }
    printf("\nNode Inserted\n");
}
}
```



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
void last_delete()
{
    struct node *ptr, *preptr;
    if(head==NULL)
    {
        printf("\nUNDERFLOW\n");
    }
    else if (head ->next == head)
    {
        head = NULL;
        free(head);
        printf("\nNode Deleted\n");
    }
    else
    {
        ptr = head;
        while(ptr ->next != head)
        {
            preptr=ptr;
            ptr = ptr->next;
        }
    }
}
```



```
preptr->next = ptr -> next;  
  
free(ptr);  
  
printf("\nNode Deleted\n");  
  
}  
  
}
```

5. Observations/Discussions/ Complexity Analysis:

In a doubly-linked list, the time complexity for inserting and deleting an element is **O(1)**.

Circular Linked List Complexity

Circular Linked List Complexity	Time Complexity	Space Complexity
Insertion Operation	O(1) or O(n)	O(1)
Deletion Operation	O(1)	O(1)

6. Result/Output/Writing Summary:-

Doubly Linked List

INSERTION:-

A) Insertion in doubly linked list at beginning



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
Enter the item which you want to insert?
```

```
5
```

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

```
0
```

```
Enter the item which you want to insert?
```

```
24
```

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

```
2
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console. █
```

B) Insertion in doubly linked list at the end

```
Enter the item which you want to insert?
```

```
35
```

```
Node Inserted
```

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

```
2
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console.
```

C) Insertion in doubly linked list after Specified node



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

Enter the item which you want to insert?

23

Node Inserted

Press 0 to insert more ?

2

...Program finished with exit code 0

Press ENTER to exit console.

DELETION :

A) Deletion at beginning

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

Enter the item
12

Node Inserted
1.Append List
2.Delete node from beginning
3.Exit
4.Enter your choice?2

Node Deleted
1.Append List
2.Delete node from beginning
3.Exit
4.Enter your choice?3

...Program finished with exit code 0
Press ENTER to exit console.



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

B) Deletion in doubly linked list at the end

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

Enter the item
23

Node Inserted
1.Append List
2.Delete node from end
3.Exit
4.Enter your choice?2

Node Deleted
1.Append List
2.Delete node from end
3.Exit
4.Enter your choice?3

...Program finished with exit code 0
Press ENTER to exit console.
```

C) Deletion in doubly linked list after the specified node



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

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

Enter the item
10

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

Enter the item
20

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

Enter the item
30

Node Inserted
1.Append List
2.Delete node
3.Exit
4.Enter your choice?2
Enter the value20

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

...Program finished with exit code 0
Press ENTER to exit console.
```

Circular Linked List

Insertion:-

A) Insertion at beginning



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC GRADE A+
ACCREDITED UNIVERSITY

```
Enter the item which you want to insert?
```

```
23
```

```
Node Inserted
```

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

```
2
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console.■
```

DELETION:-

A) Deletion in circular singly linked list at beginning

```
1.Append List  
2.Delete Node from beginning  
3.Exit  
4.Enter your choice?1
```

```
Enter the item
```

```
34
```

```
Node Inserted  
1.Append List  
2.Delete Node from beginning  
3.Exit  
4.Enter your choice?2
```

```
Node Deleted  
1.Append List  
2.Delete Node from beginning  
3.Exit  
4.Enter your choice?3
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console.
```

B) Deletion in Circular singly linked list at the end



DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.



- ```
1.Append List
2.Delete Node from end
3.Exit
4.Enter your choice?1
```

```
Enter the item
34
```

- ```
Node Inserted
1.Append List
2.Delete Node from end
3.Exit
4.Enter your choice?2
```

- ```
Node Deleted
1.Append List
2.Delete Node from end
3.Exit
4.Enter your choice?3
```

```
...Program finished with exit code 0
Press ENTER to exit console.
```

## Learning outcomes (What I have learnt):

1. Linked list.
2. Types of Linked List.
3. Operation .

## Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
|---------|------------|----------------|---------------|
| 1.      |            |                |               |
| 2.      |            |                |               |
| 3.      |            |                |               |
|         |            |                |               |



# DEPARTMENT OF ACADEMIC AFFAIRS

Discover. Learn. Empower.

NAAC  
GRADE A+  
ACCREDITED UNIVERSITY