

```

#include<stdlib.h>

#include <stdio.h>

void create();

void display();

void insert_begin();

void insert_end();

void insert_pos();

void delete_begin();

void delete_end();

void delete_pos();

struct node
{
    int info;

    struct node *next;
};

struct node *start=NULL;

int main()
{
    int choice;

    while(1){

        printf(" MENU \n");

        printf(" 1.Create \n");

        printf(" 2.Display \n");

        printf(" 3.Insert at the beginning \n");

        printf(" 4.Insert at the end \n");

        printf(" 5.Insert at specified position \n");

        printf(" 6.Delete from beginning \n");

        printf(" 7.Delete from the end \n");

        printf(" 8.Delete from specified position \n");

        printf(" 9.Exit \n");

        printf("-----\n");
    }
}

```

```
printf("Enter your choice: \n");  
scanf("%d",&choice);  
switch(choice)  
{  
case 1:  
create();  
break;  
case 2:  
display();  
break;  
case 3:  
insert_begin();  
break;  
case 4:  
insert_end();  
break;  
case 5:  
insert_pos();  
break;  
case 6:  
delete_begin();  
break;  
case 7:  
delete_end();  
break;  
case 8:  
delete_pos();  
break;  
case 9:  
exit(0);  
break;
```

```
default:
printf(" Wrong Choice:");
break;
}
}
return 0;
}
void create()
{
struct node *temp,*ptr;
temp=(struct node *)malloc(sizeof(struct node));
if(temp==NULL)
{
printf("Out of Memory Space:");
exit(0);
}
printf("Enter the data value for the node: ");
scanf("%d",&temp->info);
temp->next=NULL;
if(start==NULL)
{
start=temp;
}
else
{
ptr=start;
while(ptr->next!=NULL)
{
ptr=ptr->next;
}
ptr->next=temp;
```

```

    }
}

void display()
{
    struct node *ptr;
    if(start==NULL)
    {
        printf("List is empty: ");
        return;
    }
    else
    {
        ptr=start;
        printf("The List elements are: ");
        while(ptr!=NULL)
        {
            printf("%d ",ptr->info );
            ptr=ptr->next ;
        }
    }
}

void insert_begin()
{
    struct node *temp;
    temp=(struct node *)malloc(sizeof(struct node));
    if(temp==NULL)
    {
        printf("Out of Memory Space: ");
        return;
    }
    printf("Enter the data value for the node: " );

```

```

scanf("%d",&temp->info);
temp->next =NULL;
if(start==NULL)
{
start=temp;
}
else
{
temp->next=start;
start=temp;
}
}
void insert_end()
{
struct node *temp,*ptr;
temp=(struct node *)malloc(sizeof(struct node));
if(temp==NULL)
{
printf("Out of Memory Space: ");
return;
}
printf("Enter the data value for the node: " );
scanf("%d",&temp->info );
temp->next =NULL;
if(start==NULL)
{
start=temp;
}
else
{
ptr=start;

```

```

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

ptr->next =temp;
}
}

void insert_pos()
{
struct node *ptr,*temp;

int i,pos;

temp=(struct node *)malloc(sizeof(struct node));

if(temp==NULL)
{
printf("Out of Memory Space: ");
return;
}

printf("Enter the position for the new node to be inserted: ");
scanf("%d",&pos);

printf("Enter the data value of the node: ");
scanf("%d",&temp->info) ;

temp->next=NULL;

if(pos==0)
{
temp->next=start;
start=temp;
}

else
{
for(i=0,ptr=start;i<pos-1;i++) { ptr=ptr->next;
if(ptr==NULL)

```

```

{
printf("Position not found:[Handle with care] ");
return;
}
}
temp->next = ptr->next ;
ptr->next=temp;
}
}
void delete_begin()
{
struct node *ptr;
if(ptr==NULL)
{
printf("List is Empty: ");
return;
}
else
{
ptr=start;
start=start->next ;
printf("The deleted element is :%d ",ptr->info);
free(ptr);
}
}
void delete_end()
{
struct node *temp,*ptr;
if(start==NULL)
{
printf("List is Empty:");

```

```

exit(0);
}
else if(start->next ==NULL)
{
ptr=start;
start=NULL;
printf("The deleted element is:%d ",ptr->info);
free(ptr);
}
else
{
ptr=start;
while(ptr->next!=NULL)
{
temp=ptr;
ptr=ptr->next;
}
temp->next=NULL;
printf("The deleted element is:%d ",ptr->info);
free(ptr);
}
}
void delete_pos()
{
int i,pos;
struct node *temp,*ptr;
if(start==NULL)
{
printf("The List is Empty: ");
exit(0);
}

```



```

else
{
printf("Enter the position of the node to be deleted: ");
scanf("%d",&pos);
if(pos==0)
{
ptr=start;
start=start->next ;
printf("The deleted element is:%d ",ptr->info );
free(ptr);
}
else
{
ptr=start;
for(i=0;i<pos;i++) { temp=ptr; ptr=ptr->next ;
if(ptr==NULL)
{
printf("Position not Found: ");
return;
}
}
temp->next =ptr->next ;
printf("The deleted element is:%d ",ptr->info );
free(ptr);
}
}
}

```

C:\Users\Lenovo\Desktop\ds\LINKED LIST OPERATIONS.exe

```
MENU
1.Create
2.Display
3.Insert at the beginning
4.Insert at the end
5.Insert at specified position
6.Delete from beginning
7.Delete from the end
8.Delete from specified position
9.Exit
-----
Enter your choice:
3
Enter the data value for the node: 123
MENU
1.Create
2.Display
3.Insert at the beginning
4.Insert at the end
5.Insert at specified position
6.Delete from beginning
7.Delete from the end
8.Delete from specified position
9.Exit
-----
Enter your choice:
2
The List elements are: 123 MENU
1.Create
2.Display
3.Insert at the beginning
4.Insert at the end
5.Insert at specified position
6.Delete from beginning
7.Delete from the end
8.Delete from specified position
9.Exit
-----
Enter your choice:
345
Wrong Choice: MENU
1.Create
2.Display
3.Insert at the beginning
4.Insert at the end
5.Insert at specified position
6.Delete from beginning
7.Delete from the end
8.Delete from specified position
9.Exit
```