Objective:

To implement Linked List and perform various insertion and deletion operations on it.

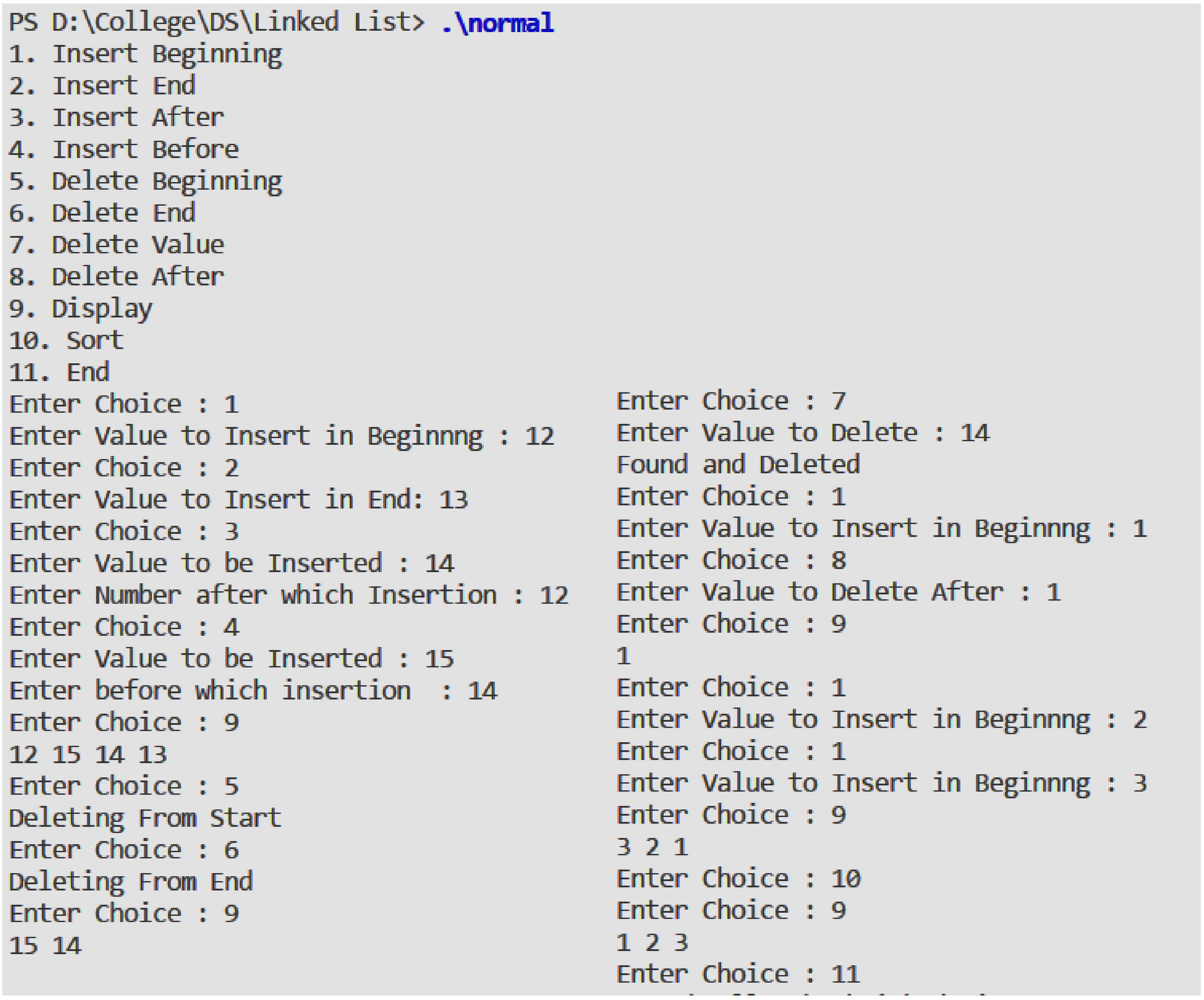
Code :

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

typedef struct node {
    int value ;
    struct node* next ;
} node ;

node* getnewnode() ;
node* insertbeg( node* , int);
node* insertend( node* , int);
node* insertafter( node* , int , int);
node* insertbefore( node* , int , int);
node* deletebeg( node* );
node* deleteend( node* );
node* deletevalue( node*, int);
node* deleteafter( node*, int );;
node* sortll( node* );;

void displayll( node* );

int main(){

    int ch , n , temp ;
    node* start = NULL ;
    printf("1. Insert Beginning \n");
    printf("2. Insert End \n");
    printf("3. Insert After \n");
    printf("4. Insert Before \n");
    printf("5. Delete Beginning \n");
    printf("6. Delete End \n");
    printf("7. Delete Value \n");
    printf("8. Delete After \n");
    printf("9. Display \n");
    printf("10. Sort \n");
    printf("11. End \n");

    do{
        printf("Enter Choice : ");
        scanf("%d" , &ch );
        switch(ch) {
            case 1 :
                printf("Enter Value to Insert in Beginnng : ");
                scanf("%d" , &n);
                start  = insertbeg(start , n);
                break;
            case 2 :
                printf("Enter Value to Insert in End: ");
                scanf("%d" , &n);
                start  = insertend(start , n);
                break;
            case 3 :
                printf("Enter Value to be Inserted : ");
                scanf("%d" , &n);
                printf("Enter Number after which Insertion : ");
                scanf("%d" , &temp);
                start  = insertafter(start , n ,temp );
                break;
            case 4 :
                printf("Enter Value to be Inserted : ");
                scanf("%d" , &n);
                printf("Enter before which insertion  : ");
                scanf("%d" , &temp);
                start  = insertbefore(start , n , temp);
                break;
            case 5 :
                printf("Deleting From Start\n");
                start = deletebeg(start);
                break;
            case 6 :
                printf("Deleting From End\n");
                start = deleteend(start);
                break;
            case 7 :
                printf("Enter Value to Delete : ");
                scanf("%d" , &n);
                start = deletevalue(start , n);
                break;
            case 8 :
                printf("Enter Value to Delete After : ");
                scanf("%d" , &n);
                start = deleteafter(start , n);
                break;
            case 9 :
                displayll(start);
                break;
            case 10 :
                start = sortll(start);
                break;
        }

    } while ( ch != 11) ;

}

node* getnewnode(){
    node* new = malloc(sizeof( node ) );
    return new;
}

node* insertbeg( node* start  , int x){

    node* q = getnewnode() ;
    q->value = x ;
    q->next = start ;
    start = q ;
    return start ;

}

node* insertend( node* start  , int x){

    node* q = getnewnode() ;
    q->value = x ;
    node* p = start ;

    if( start == NULL ){
        start = q ;
        start->next = NULL ;
        return start;
    }
    while(  p->next != NULL ){
        p = p->next ;
    }
    q->next = p->next ;
    p->next = q ;

    return start ;

}

node* insertafter( node* start  , int x , int y){

    node* q = getnewnode() ;
    q->value = x ;
    node* p = start ;

    if( start == NULL ){
        printf("Empty Linked List. Inserting at Beginning\n");
        start = q ;
        start->next = NULL ;
        return start;
    }

    while(  p->next != NULL && p->value != y ){
        p = p->next ;
    }

    if(p->value != y ){
        printf("Not Found. Inserting at End\n");
    }

    q->next = p->next ;
    p->next = q ;

    return start ;

}
```

```
node* insertbefore( node* start  , int x , int y ){

    node* q = getnewnode() ;
    q->value = x ;
    node* p = start ;

    if( start == NULL ){
        printf("Empty Linked List. Inserting at Beginning\n");
        start = q ;
        start->next = NULL ;
        return start;
    }

    if( start->value == y ){
        q->next = start ;
        start = q ;
    } else {
        while(  p->next != NULL && p->next->value != y ){
            p = p->next ;
        }

        if(p->next == NULL ){
            printf("Not Found. Inserting at End\n");
        }

        q->next = p->next ;
        p->next = q ;

    }

    return start ;

}


node* deletebeg( node* start){

    if( start == NULL ){
        printf("Empty\n");
        return start;
    }

    node* q = start ;
    start = start->next ;
    free(q) ;

    return start ;

}

node* deleteend( node* start){

    node* q  ;
    node* p = start ;

    if( start == NULL ){
        printf("Empty\n");
        return start;
    }

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

    q = p->next ;
    p->next = p->next->next ;
    free(q) ;

    return start ;

}


node* deletevalue( node* start  , int x){


    if( start == NULL ){
        printf("List is Empty\n");
        return start;
    }

    node* q ;

    if( start->value == x ){
        q = start ;
        start = start->next ;
        return start ;
    }

    node* p = start;

    while( p->next != NULL ){
        if(p->next->value == x){
            printf("Found and Deleted\n");
            q = p->next ;
            p->next = q->next ;
            free(q);
            return(start) ;
        }
        p = p->next ;
    }

    printf("Not Found\n") ;

    return start ;

}

node* deleteafter( node* start  , int x){


    if( start == NULL ){
```

```c
        printf("List is Empty\n");
        return start;
    }

    node* q ;

    if( start->value == x ){
        start->next = start->next->next ;
        return start ;
    }

    node* p = start;

    while( p->next != NULL ){
        if(p->value == x){
            printf("Found and Deleted\n");
            q = p->next ;
            p->next = q->next ;
            free(q);
            return(start) ;
        }
        p = p->next ;
    }

    printf("Not Found\n") ;

    return start ;

}


void displayll(node* start){

    while(start != NULL ){

        printf("%d " , start->value);
        start = start->next ;

    }
    printf("\n") ;

}

node* sortll(node* start){

    node *q , *p ;
    int temp ;

    if(start == NULL ){
        return start ;
    }

    p = start ;
    while(p->next != NULL ){

        q = p->next ;
        while( q != NULL ){

            if ( p->value > q->value ){

                int temp = p->value ;
                p->value = q->value ;
                q->value = temp ;
            }
            q = q->next ;
        }
        p = p->next ;
    }
    return start ;
}
```

Output :



```
PS D:\College\DS\Linked List> .\normal
1. Insert Beginning
2. Insert End
3. Insert After
4. Insert Before
5. Delete Beginning
6. Delete End
7. Delete Value
8. Delete After
9. Display
10. Sort
11. End
Enter Choice : 1
Enter Value to Insert in Beginnng : 12
Enter Choice : 2
Enter Value to Insert in End: 13
Enter Choice : 3
Enter Value to be Inserted : 14
Enter Number after which Insertion : 12
Enter Choice : 4
Enter Value to be Inserted : 15
Enter before which insertion  : 14
Enter Choice : 9
12 15 14 13
Enter Choice : 5
Deleting From Start
Enter Choice : 6
Deleting From End
Enter Choice : 9
15 14
Enter Choice : 7
Enter Value to Delete : 14
Found and Deleted
Enter Choice : 1
Enter Value to Insert in Beginnng : 1
Enter Choice : 8
Enter Value to Delete After : 1
Enter Choice : 9
1
Enter Choice : 1
Enter Value to Insert in Beginnng : 2
Enter Choice : 1
Enter Value to Insert in Beginnng : 3
Enter Choice : 9
3 2 1
Enter Choice : 10
Enter Choice : 9
1 2 3
Enter Choice : 11
```