```c
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int data;
    struct node * next;
};
struct node *head=NULL;
void insertbegin();
void display();
void insertposition(int,int,struct node *);
void insertend(int);
void deletebegin();
void singlysearch();
int main()
{
    int item,n,op;
    struct node*p;
    while(op!=7){
        printf("\n********Menu**************\n");
        printf("1-> Insert\n");
        printf("2-> Insert at end\n");
        printf("3->Insert at position\n");
        printf("4->Search for data\n");
        printf("5->Delete data\n");
        printf("6->Display\n");
        printf("7->Exit\n");
        printf("Enter your choice: \n");
        scanf("%d", &op);

    switch(op){
        case 1:
            insertbegin();
            break;
        case 2:
            insertend(item);
            break;
        case 3:
            insertposition(n,item,head);
            break;
        case 4:
            singlysearch();
            break;
        case 5:
            deletebegin();
            break;
        case 6:
            display();
            break;
        case 7:
            printf("program exiting....");
            break;
        default:
            printf("invaild choice");
        return 0;
        }
    }
}

void insertposition(int pos,int data,struct node *head)
{
    struct node *temp;
    struct node *newnode=(struct node *)malloc(sizeof(struct node));
    newnode->next=NULL;
    printf("\nenter at position:");
    scanf("%d",&pos);
    pos=pos-1;
    printf("enter data:");
    scanf("%d",&newnode->data);
    newnode->next=NULL;
    if(pos==0)
        {
            head=newnode;
        }
    else{
        temp=head;
        while(--pos)
            {
                temp=temp->next;
            }
        newnode->next = temp->next;
        temp->next = newnode;
        }
}
void insertbegin(){
    struct node *newnode;
    newnode=(struct node*)malloc(sizeof(struct node));
    printf("enter data:");
    scanf("%d",&newnode->data);
    newnode->next=NULL;
    if(head==NULL){
            head=newnode;
    }
    else{
            newnode->next=head;
            head=newnode;
        }
}

void insertend(int item)
{
    struct node *temp;
    struct node *newnode=(struct node *)malloc(sizeof(struct node *));
    printf("\nenter data for end node ");
    scanf("%d",&newnode->data);
    if(head==NULL){
            newnode->next=NULL;
            head=newnode;
    }
    else{
            temp=head;
            while(temp->next!=NULL){
            temp=temp->next;
            }
    temp->next=newnode;
    newnode->next=NULL;
    }
}

void deletebegin()
{
    struct node *temp;
    if(head==NULL){
        printf("\nDLL Is empty\n");
    }
    else if(head->next==NULL){
        head=NULL;
        free(head);
        printf("Begin Node deleted\n");
    }
    else{
        temp=head;
```

```c
        head=head->next;
        printf("\n\nBegin node %d deleted\n",temp->data);
        free(temp);
    }
}

void singlysearch()
{
    struct node *temp;
    int value,count=0,flag=0;
    temp=head;
    printf("\nenter the value to search :");
    scanf("%d",&value);
    while(temp!=NULL){
        if(temp->data==value){
            flag=1;
            count++;
        }
        temp=temp->next;
    }
    if(flag==1){
        printf("Search value %d found",value);
    }
    else{
        printf("search value not found");
    }
}

void display()
{
    struct node * temp;
    if(head==NULL)
        {
            printf("list is empty\n");
    }
    else{
            temp=head;
            printf("List elements\n");
            while(temp!=NULL)
                {
                    printf("%d\t",temp->data);
                    temp=temp->next;

                }
        }
}
```