

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
struct Node{
    int data;
    struct Node*next;
};
struct Node*head=NULL;

struct Node* createNode(int value){
    struct Node* newNode=(struct Node*)malloc(sizeof(struct Node));
    newNode->data=value;
    newNode->next=NULL;
    return newNode;
}
void insertAtBeg(int value){
    struct Node* newNode=createNode(value);
    newNode->next=head;
    head=newNode;
}
void insertAtEnd(int value){
    struct Node* newNode=createNode(value);
    if(head==NULL){
        head=newNode;
        return;
    }
    struct Node*temp=head;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    temp->next=newNode;
}
void insertAtPosition(int value,int pos){
    struct Node* newNode=createNode(value);
    if(pos==1){
        newNode->next=head;
        head=newNode;
        return;
    }
    struct Node* temp=head;
    for(int i=1;temp!=NULL && i<pos-1;i++){
        temp=temp->next;
    }
    if(temp==NULL){
        printf("position not found!\n");
        return;
    }
    newNode->next=temp->next;

```

```

    temp->next=newNode;

}
void deleteBegin(){
    if(head==NULL){
        printf("List is empty");
        return;
    }
    struct Node* temp=head;
    head=head->next;
    printf("%d deleted from beginning\n",temp->data);
    free(temp);
}
void deleteEnd(){
    if(head == NULL){
        printf("Linked list is empty\n");
        return;
    }
    if(head->next == NULL){
        printf("%d deleted from end\n", head->data);
        free(head);
        head = NULL;
        return;
    }
    struct Node* temp = head;
    while(temp->next->next != NULL){
        temp = temp->next;
    }
    printf("%d deleted from end\n", temp->next->data);
    free(temp->next);
    temp->next = NULL;
}

void deleteAtPos(int pos){
    if(head==NULL){
        printf("linked list is empty");
        return;
    }
    if(pos==1){
        deleteBegin();
        return;
    }
    struct Node* temp=head;
    for(int i=1;temp!=NULL&& i<pos-1;i++){
        temp=temp->next;
    }
    if(temp->next==NULL||temp==NULL){
        printf("position not found");
        return;
    }
}

```

```

        struct Node*del=temp->next;
        temp->next=del->next;
        printf("%d deleted from %d position",del->data,pos);
        free(del);
    }
    void display(){
        struct Node* temp=head;
        if(temp==NULL){
            printf("linked list is empty.\n");
            return;
        }
        printf("linked list: ");
        while(temp!=NULL){
            printf("%d->",temp->data);
            temp=temp->next;
        }
        printf("NULL\n");
    }
    int main(){
        int choice, value, pos;
        while(1){
            printf("\nlinked list menu:\n");
            printf("1. insert at beginning\n");
            printf("2. insert at end\n");
            printf("3. insert at a position\n");
            printf("4. delete from beginning\n");
            printf("5. delete from end\n");
            printf("6. delete from a position\n");
            printf("7. display\n");
            printf("8. exit\n");
            printf("enter your choice: ");
            scanf("%d", &choice);
            switch (choice) {
                case 1:
                    printf("Enter value: ");
                    scanf("%d", &value);
                    insertAtBeg(value);
                    break;
                case 2:
                    printf("Enter value: ");
                    scanf("%d", &value);
                    insertAtEnd(value);
                    break;
                case 3:
                    printf("Enter position: ");
                    scanf("%d", &pos);
                    printf("Enter value: ");
                    scanf("%d", &value);
                    insertAtPosition(value, pos);

```

```
        break;
    case 4:
        deleteBegin();
        break;
    case 5:
        deleteEnd();
        break;
    case 6:
        printf("Enter position: ");
        scanf("%d", &pos);
        deleteAtPos(pos);
        break;
    case 7:
        display();
        break;
    case 8:
        exit(0);
    default:
        printf("Invalid choice!\n");
}
}
```