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

struct Node{
    int data;
    struct Node *link;
};




//Add some more nodes
struct Node* add_some_nodes(struct Node *head, int data){
    struct Node *ptr = NULL;
    ptr = malloc(sizeof(struct Node));
    ptr->data = data;
    ptr->link = NULL;

    head->link = ptr;
    head = ptr;
    return head;
}

//Traverse the list
void traverse_list(struct Node *head){
    struct Node *ptr = head;

    if(ptr == NULL){
        printf("\nList is already empty\n");
        return;
    }else{
        printf("\nData inside the list is\n");
        while(ptr != NULL){
            printf("%d\t", ptr->data);
            ptr = ptr->link;
        }
        printf("\n");
    }
}


//Add new nodes at the end
void add_at_end(struct Node *head, struct Node *last){
    struct Node *ptr = head;

    if(ptr == NULL){
        printf("List is already empty\n");
        return;
    }else{
        while(ptr->link != NULL){
            ptr = ptr->link;
        }
        ptr->link = last;
    }
}


//Add new node at beg
```

```c
void add_new_at_beg(struct Node **head, struct Node *first){
    struct Node *ptr = *head;

    first->link = *head;
    *head = first;
}


//Count number of nodes
int count_no_of_nodes(struct Node *head){
    struct Node *ptr = head;
    int count = 0;

    if(ptr == NULL){
        return count;
    }else{
        while(ptr != NULL){
            count++;
            ptr = ptr->link;
        }
        return count;
    }
}


//Add new node at any pos
void add_at_pos(struct Node **head, struct Node *new, int pos){
    struct Node *ptr = *head;

    if(pos == 1){
        add_new_at_beg(head, new);
        return;
    }else{
        pos--;
        while(--pos){
            ptr = ptr->link;
        }
        new->link = ptr->link;
        ptr->link = new;
    }
}


//Delete first node
void del_first_node(struct Node **head){
    struct Node *ptr = *head;

    if(ptr == NULL){
        printf("List is already empty\n");
        return;
    }else{
        *head = (*head)->link;
        free(ptr);
        ptr = NULL;
    }
}

//Delete last node
void del_last_node(struct Node **head){
    struct Node *temp1 = *head;
    struct Node *temp2 = *head;

    if(*head == NULL){
        printf("List is already empty\n");
        return;
    }else if((*head)->link == NULL){
        free(*head);
        *head = NULL;
    }else{
        while(temp1->link != NULL){
            temp2 = temp1;
            temp1 = temp1->link;
        }
        temp2->link = NULL;
        free(temp1);
        temp1 = NULL;
    }

}



//Delete a node at some position
void del_at_pos(struct Node **head, int pos){
    struct Node *ptr = *head;
    struct Node *temp1 = *head;
    struct Node *temp2 = *head;

    if(ptr == NULL){
        printf("List is already empty\n");
        return;
    }else if((*head)->link == NULL){
        free(*head);
        *head = NULL;
    }else{
        while(--pos){
            temp1 = temp2;
            temp2 = temp1->link;
        }
        temp1->link = temp2->link;
        free(temp2);
        temp2 = temp1 = NULL;
    }
}



//Delte all nodes (forward deletion)
void del_all_nodes(struct Node **head){
    struct Node *ptr = NULL;

    if(*head == NULL){
        printf("List is already empty\n");
        return;
    }else{
        while((*head) != NULL){
            ptr = *head;
            *head = (*head)->link;
            free(ptr);
            ptr = NULL;
```

```c
        }
    }
}




//Testing
int main(){
    struct Node *head = NULL;
    head = malloc(sizeof(struct Node));
    head->data = 100;
    head->link = NULL;

    struct Node *ptr = head;

    int yourChoice;
    do{
        system("CLS");
        printf("\n\n");
        printf("\t <Insertion Section>\n");
        printf("<1>. Add some new nodes\n");
        printf("<2>. Traverse the list\n");
        printf("<3>. Add new node at the end\n");
        printf("<4>. Add new node at beg\n");
        printf("<5>. Count number of nodes\n");
        printf("<6>. Add new node at any position\n");
        printf("\t <Deletion Section>\n");
        printf("<7>. Delete first Node\n");
        printf("<8>. Delete last Node\n");
        printf("<9>. Delete node at any position\n");
        printf("<10>. Delete all nodes in one short\n");
        printf("<11>. Exit\n");
        printf("Enter you choice:  ");
        scanf("%d", &yourChoice);

        switch(yourChoice){
            case 1:
                printf("\nEnter number of new nodes to add: ");
                int newNodes, newData;
                scanf("%d", &newNodes);
                for(int i = 1; i <= newNodes; i++){
                    printf("\tEnter data of new node: ");
                    scanf("%d", &newData);
                    ptr = add_some_nodes(ptr, newData);
                }
                printf("\n\tHit enter to continue:-...");
                getch();
                break;

            case 2:
                traverse_list(head);
                printf("\n\tHit enter to continue:-...");
                getch();
                break;

            case 3:
                printf("\n\tEnter data of new node: ");
                scanf("%d", &newData);
                struct Node *last = malloc(sizeof(struct Node));
                last->data = newData;
                last->link = NULL;

                add_at_end(head, last);
                break;

            case 4:
                printf("\n\tEnter data of new node: ");
                scanf("%d", &newData);
                struct Node *first = malloc(sizeof(struct Node));
                first->data = newData;
                first->link = NULL;

                add_new_at_beg(&head, first);
                break;

            case 5:
                printf("\n\t");
                int noOfNodes = 0;
                noOfNodes = count_no_of_nodes(head);

                printf("No of nodes are: %d", noOfNodes);
                printf("\n\tHit enter to continue:-...");
                getch();
                break;

            case 6:
                printf("\n\tEnter position of new node to insert: ");
                int pos;
                scanf("%d", &pos);

                printf("\n\tEnter data of new node: ");
                scanf("%d", &newData);
                struct Node *newNode = malloc(sizeof(struct Node));

                newNode->data = newData;
                newNode->link = NULL;

                add_at_pos(&head, newNode, pos);
                break;

            case 7:
                del_first_node(&head);
                break;

            case 8:
                del_last_node(&head);
                break;

            case 9:
                printf("\tEnter position no to delete: ");
                scanf("%d", &pos);

                del_at_pos(&head, pos);
                break;

            case 10:
                del_all_nodes(&head);
                break;

            case 11:
                exit(1);

            default:
                printf("Invalid Choice\n");
        }
    }while(1);


    return 0;
}


```