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

struct Node{
    int data;
    struct Node * next;
};
void linkedlistTraversal(struct Node* ptr){
    while(ptr != NULL){
        printf("Element: %d\n", ptr->data);
        ptr = ptr->next;
    }
}
//time compelexity = 0(1)
struct Node * insertAtFirst(struct Node *head, int data){
    struct Node * ptr = (struct Node *) malloc(sizeof(struct Node));
    ptr -> next = head;
    ptr -> data = data;
    return ptr;
};
//time compelexity = 0(n)
struct Node * insertAtEnd(struct Node *head, int data){
    struct Node * ptr = (struct Node *) malloc(sizeof(struct Node));
    ptr->data = data;
    struct Node *p =head;

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

};
//time compelexity = 0(n)
struct Node * insertAtIndex(struct Node *head, int data, int index){
    struct Node * ptr = (struct Node *) malloc(sizeof(struct Node));
    struct Node *p = head;
    int i = 0;

    while(i!=index-1){
        p = p->next;
        i++;
    }
    ptr->data = data;
    ptr->next = p->next;
    p->next = ptr;
    return head;
};
//time compelexity = 0(1)
struct Node * insertAfterNode(struct Node *head, struct Node
*prevNode, int data){
    struct Node * ptr = (struct Node *) malloc(sizeof(struct Node));
    ptr->data = data;
```

```
    ptr->next = prevNode->next;
    prevNode->next = ptr;

    return head;
};
struct Node * delFirst(struct Node *head){
    struct Node *ptr = head;
    head = head->next;
    void free(void *ptr);
    return head;
};
struct Node * delatIndex(struct Node * head, int index){
    struct Node *p = head;
    struct Node *q = head->next;
    for(int i = 0; i<index-1; i++)
    {
        p = p->next;
        q = q->next;
    }
    p->next = q->next;
    free(q);
    return head;
};
struct Node * delLast(struct Node * head){
    struct Node *p = head;
    struct Node *q = head->next;
    while (q->next!=NULL)
    {
        p = p->next;
        q = q->next;
    }
    p->next = NULL;
    free(q);
    return head;
};
struct Node * delValue(struct Node * head, int value){
    struct Node *p = head;
    struct Node *q = head->next;
    while(q->data!=value && q->next!=NULL)
    {
        p = p->next;
        q = q->next;
    }
    if(q->data == value){
        p->next = q->next;
        free(q);
    }
    return head;
};


int main(){
    struct Node * head;
    struct Node * second;
```

```
    struct Node * third;
    struct Node * fourth;


    //Allocation of memories for nodes of linked list in the heap
    head = (struct Node *) malloc(sizeof(struct Node));
    second = (struct Node *) malloc(sizeof(struct Node));
    third = (struct Node *) malloc(sizeof(struct Node));
    fourth = (struct Node *) malloc(sizeof(struct Node));


    //Linkedlist first and seond nodes
    head->data = 7;
    head->next = second;

    // linking second and third nodes
    second->data = 11;
    second->next = third;

    // Termination of the list
    third->data = 66;
    third->next = fourth;

    fourth ->data = 89;
    fourth->next = NULL;

    printf("List before insertion:\n");
    linkedlistTraversal(head);
    //head = insertAtFirst(head, 56);
    //head = insertAtIndex(head, 56,1);
    //head = insertAtEnd(head, 56);
    head = insertAfterNode(head,second,56);
    printf("List after insertion:\n");
    linkedlistTraversal(head);

    return 0;
}
```