Homework: Doing some operation on linked list such as traversal, Complexity, creating a list, inserting elements (front, inside, End), deletion operations.

```
#include<stdio.h>
#include<stdlib.h>
Struct Node {
        int data;
        struct Node* next;
       };
void traverse(struct Node** Read) {
        Struct Node * Current = Read;
      if(current == Null) {
          Printf("the list is Empty.\n");
           return;
       }
     while (current != Null) {
      Printf("%d—>", Current—> data);
      Current = Current—> next;
     }
        Printf("NULL\n");
}
```

Complexity

O(1)

```
Struct Node* insertAtFront (struct Node *head, int new_data {
    Struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
    if(new_node == NULL) {
        Printf("Memory allocation failed\n"),
         return head,.
        }
       new_node—>data = new_data;
        new_node —> next = head;
    return new_node;
    }
```

inserting inside O(1)

```
void InsertInside(struct Node *Prev_node, int new_data) {
    if(prev_node == NULL) {
        Printf("the given previous node cannot be NULL.\n");
        return;
    }

struct Node *new_node=(struct Node*)malloc(sizeof(struct Node));
    if(new_node == NULL){
        Printf("Memory allocation failed!\n");
        return;
    }
new_node->data = new_data;
new_node->next = Prev_node->next;
Prev_node->next= new_node;

};
```

insert at End O(n).

```
Struct Node *InsertAtEnd(struct Node *head, int new_data){
        struct Node *new_node = (struct Node*) malloc(sizeof(struct Node));
        if(new_node == Null){
        Printf("Memory allocation failed!\n
        return head;
    }
new-node = new_data,
new-node->next = NULL;

if(head == NULL) {
    Printf("inserted %d at the end (as head).\n")
     return new_node;
  }
  struct Node *Last = head;

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

```
    last -> next = new_node;
    Printf("Inserted %d at end\n", new_data);
    return head;
}
```

Deletion

```
Struct Node * deleteNode(struct Node * head, int key){
        Struct Node * current = head;
        Struct Node * prev = NULL;
    if(current != NULL && current -> data == key){
            head = current -> next;
            free(current);
            Printf("Deleted %d
```