—> linked list (insertion)

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

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

struct Node* createNode (int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(
                                          struct Node));
    newNode -> data = value;
    newNode -> next = NULL;
    return newNode;
}

void displaylist (struct Node* head) {
    struct Node* current = head;
    while (current != NULL) {
        printf("%d ->", current -> data);
        current = current -> next)
    }
    printf("NULL\n");
}

~~Struct~~ void Node* insertAtBeg (struct Node* head, int value){
    struct Node* newNode = createNode (value);
    newNode -> next = head;
    return newNode;
```

```
void inseotatpos (stouct Node* head, int value, int pos) {
    stouct Node* newNode = createNode (value);
    stouct Node* curient= head;
    foo (int i = 1; i<pos - 1 && current != NULL; i++) {
        cuooent = cuooent ->next;
    }
    if (cuooent != NULL) {
      newNode ->next= current ->next;
      current -> next = newNode;
    } eles {
       pointf(" Invalid position\n");
    }
}

void inseotatend (Stouct Node* head, int value) {
    stouct Node* new Node = createNode (value);
    stouct Node* curient= head;
    while (current -> next != NULL) {
        cullent= cuooent -> next;
  }
   current -> next= newNode;

 int main (){
    stouct Node* head = NULL;
    head = inseot atBeg (head, 1);
    inseotAtEnd (head, 3);
    inseotatPos (head, 2, 2);
    displaylist (head);
    return0;
 }
```

O/P

```
Linked list : 1 ⟶ NULL
linked list : 1 → 2 → NULL
Linked list : 1 → 2 → 3 ⟶ NULL
```

—> Linked list (Delehon)

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

stouct Node{
    int data;
    stouct Node* next;
 }:

Stouct Node* createNode (int data) {
  stouct Node* newnode = (stouct Nde*) malloc (sizeof(
                                          stouct Node));

  newNode->data = data;
  newNode ->next = NULL;
  retuon new Node;
}

Stouct Node* deletefiost (stouct Node* head) {
    if (head == NULL) {
        pointf ("List Empty");
        rehoon NULL;
    }
    stouct Node* temp = head;
    head = head->next;
    free(temp);
    retuon head;
 }
```

```
Struct Node* deleteElement (struct Node*head, int value){
    ~~if(~~
    struct Node* current = head;
    struct Node* p = NULL;
    while (current != NULL && current -> data != value){
        p = current;
        current = current -> next;
    }

  if (current != NULL){
        if (p == NULL){
            head = head -> next;
    } else {
            p -> next = current -> next;
        }
      free (current);
    } else {
       printf("Element %d not found", value);
    }
    return head;
}

Struct Node* deletelast (struct Node* head) {
    ~~if (head ==~~
    if (head -> next == NULL){
        free(head);
        return NULL;
    }
Struct Node* current = head;
struct Node* p = NULL;
while (current->next != NULL) {
   p = current;
   current = current->next;
}
   free(current);
   p->next = NULL;
   return head;
 }
}

void display_list(struct Node* head) {
   printf("linked list : ");
   while (head != NULL) {
      printf("%d -> ", head->data);
      head = head->next;
   }
    printf("NULL\n");
}

int main() {
    struct Node* head = NULL;
    head = ~~insert~~ deletefirst(head);
    display_list(head);
    head = deleteElement(head, 2);
    display_list(head);
    head = deletelast(head);
    displaylist(head);
    return 0;
}
```

O/P

```
Linked list:  1 -> 2 -> 3 -> NULL
Linked list:   2 -> 3 -> NULL
linked list:   3 -> NULL
linked list: NULL
```

22/01/24