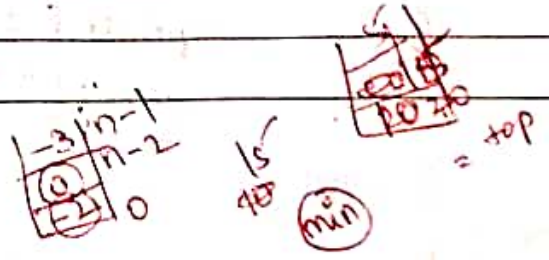
WEEK - 4

⇒ Deletion from a linked list.

```
#include <stdio.h>
#include <stdlib.h>
void pop();
void end_delete();
void delete_at_pos();
void display();
void append();
struct node
{
    int data;
    struct node *next;
};
struct node *head = NULL

void main()
{
    printf("Insert the elements in list \n");
    append();
    printf("1. Delete from beginning \n 2. Delete
        at end \n 3. Delete at particular position
        \n 4. Display \n 5. Exit \n");
    int ch;
    while (ch!=5)
    {
        printf("Enter choice :");
        scanf("%d", &ch);
        switch(ch)
        {
```



```
            case 1:
                pop()
                break;
            case 2:
                end_delete();
                break;
            case 3:
                delete_at_pos();
                break;
            case 4:
                display();
                break;
            default:
                printf("Invalid choice");
                break;
        }
    }
}

Void append()
{
    int data, n;
    printf("Enter no. of nodes:");
    scanf("%d", &n);
    for(int i=0; i<n; i++)
    {
        struct node *last = head;
        struct node *new_node;
        new_node = (struct node*) malloc
                    (sizeof(struct node));
        printf("Enter the data:");
        scanf("%d", &new_node -> data);
```

```
        new_node -> next = NULL;
        if (head == NULL)
            head = new_node;
        else
        {   while (last -> next != NULL)
                last = last -> next;
            last -> next = new_node;
        }
    }
}

void pop()
{
    struct node * ptr;
    if (head == NULL)
        printf("List is empty\n");
    else
    {
        ptr = head;
        head = ptr -> next;
        free(ptr);
        printf("Node deleted from beginning\n");
    }
}

void end_delete()
{
    struct node *ptr;
    struct node *ptr1;
    if (head == NULL)
        printf("List is empty\n");
```

```
        else if (head -> next == NULL)
        {       free(head);
                head = NULL;
        }
        else
        {
            ptr = head;
            ptr1 = head;
            while ( ptr -> next != NULL)
            {       ptr1 = ptr;
                    ptr = ptr -> next;
            }
            ptr1 -> next = NULL;
            free(ptr);
            printf(" Node deleted from end \n");
        }
}

void delete_at_pos ( )
{
    struct node *ptr;
    struct node *ptr1;
    int pos;
    printf(" Enter the position of deletion : \n");
    Scanf(" %d", &pos);
    ptr = head;
    for (int i=0 ; i<pos-1 ; i++)
    { if (ptr == NULL)
      { printf(" There are less elements \n");
        return ; }
      ptr1 = ptr;
      ptr = ptr -> next;
    }
```

```
        ptr1 -> next = ptr -> next;
        free(ptr);
        printf("Node deleted from position \n");
    }

    void display()
    {
        struct node *p = head;
        printf("List : \n");
        while(p != NULL)
        {
            printf("%d -> ", p -> data);
            p = p -> next;
        }
        printf("NULL \n");
    }
```

18/1/24

OUTPUT :-

```
Enter no. of nodes : 5
Enter the ~~choice~~ data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
1. Delete from begining
2. Delete at end
3. Delete at particular position
4. Display
5. Exit
Enter choice : 1
Node deleted from beginning
```

Enter choice : 2
Node deleted from end
Enter choice : 4
List :
2 -> 3 -> 4 -> NULL
Enter choice : 3
Enter the position of deletion :
3
Node deleted from position 3
Enter choice : 4
List :
2 -> 3 -> NULL
Enter choice : 5
Exited .