Insertion & Deletion – Singly Linked List

```
#include<stdio.h>
#include<stdlib.h>
int count=0;                    //To count no. of nodes in list

struct node                     //Define the structure of a node
{
        int no;                 //data element of a node
        struct node *next;      //self-referential element of a node
}*new, *first=NULL, *ptr;

void get_element()              //create a new node & store data
{
    new = (struct node*) malloc(sizeof(struct node));    //DMA
    printf("New Address: %p\n", new);
    printf("Enter data to insert: ");
    scanf("%d", &new->no);                               //Get data
    new->next=NULL;
}

void create_list()              //create or append the list with 'n' nodes
{
        int n;
        printf("Enter the number of elements to be inserted: ");
        scanf("%d", &n);
        count = count+n;
        for(int i=1; i<=n; i++)
        {
                get_element();                  //create new node
                if(first==NULL) first=new;      //head node
                else
                {
                        for(ptr=first; ptr->next!=NULL; ptr=ptr->next);   //traverse
                        ptr->next=new;                                    //link nodes
                }
                printf("\nElement-%d inserted\n", i);
        }
}

void insert_begin()             //Insert at beginning of list
{
        get_element();          //create node
        count = count+1;
        if(first==NULL) first=new;      //Head node
        else
        {
                new->next=first;  //swap previous head with new head
                first=new;
        }
        printf("Element inserted in Start of List\n");
}
```

```
void insert_end()
{
	get_element();		//create node
	count = count+1;
	if(first==NULL) first=new;		//Head node
	else
	{
		for(ptr=first; ptr->next!=NULL; ptr=ptr->next);	//traverse
		ptr->next=new;					//link nodes
	}
	printf("Element inserted in End of List\n");
}

void insert_pos()		//insert in middle of list
{
	struct node *temp;		//create temp for swapping node addresses
	get_element();			//create new node
	printf("Enter the position >1 and <%d: ", count);
	int pos;
	scanf("%d", &pos);		//Get position in list to insert

	if(pos>1 && pos<count)		//position should not be beginning or end of list
	{
		count = count+1;
		ptr=first;			//Set 'ptr' to head node before for loop entry
		for(int i=1; i<pos-1; i++)
		{
			ptr=ptr->next;	//Traverse the list to 'pos-1'
		}

		//Swap the addresses in nodes
		temp = ptr->next;
		ptr->next = new;
		new->next = temp;
		printf("Element inserted in given Position\n");
	}
	else printf("Cannot be inserted in beginning and end of list\n");
}

void insert()	//Menu for Insertion
{
	int choice;
	printf("\nWhere to Insert.....Enter 1(Start), 2(End), 3(Middle), 4(Create List): ");
	scanf("%d", &choice);
	switch(choice)
	{
		case 1: insert_begin(); break;
		case 2: insert_end(); break;
		case 3: insert_pos(); break;
		case 4: create_list(); break;
		default: printf("Wrong Choice\n");
	}
}
```

```
void delete_begin()
{
	if(first->next==NULL) first=NULL;		//When List has one node
	else
	{
		ptr=first->next;			//swap the node
		first=ptr;
	}
	count = count-1;
	printf("Element is Deleted from Start of List\n");
}

void delete_end()
{
	if(first->next==NULL) first=NULL;		//When List has one node
	else
	{
		for(ptr=first; ptr->next->next!=NULL; ptr=ptr->next);	//traverse the list
		ptr->next=NULL;
	}
	count = count-1;
	printf("Element is Deleted from End of List\n");
}

void delete_pos()
{
	printf("Enter the position >1 and <%d: ", count);
	int pos;
	scanf("%d", &pos);

	if(pos>1 && pos<count)		//position should not be beginning or end of list
	{
		count = count-1;
		ptr=first;			//Set 'ptr' to head node before for loop entry
		for(int i=1; i<pos-1; i++)
		{
			ptr=ptr->next;	//Traverse the list to 'pos-1'
		}
		ptr->next=ptr->next->next;	//Link the node
		printf("Element is Deleted from given Position\n");
	}
	else printf("Cannot Delete..\n");
}
```

```
void delete()               //Menu for Deletion
{
        int choice;
        printf("\nWhere do you want Delete from?.....Enter 1(Start), 2(End), 3(Middle): ");
        scanf("%d", &choice);
        if(first!=NULL)
        {
                switch(choice)
                {
                        case 1: delete_begin(); break;
                        case 2: delete_end(); break;
                        case 3: delete_pos(); break;
                        default: printf("Wrong Choice\n");
                }
        }
        else printf("List is Empty\n");
}

void display()
{
        if(first==NULL) printf("List is empty\n");                      //check for underflow
        else
        {
                printf("\nNo. of elements in List: %d\n", count);
                for(ptr=first; ptr!=NULL; ptr=ptr->next)                //Traverse list
                        printf("Block Address:%p, Data: %d, Next: %p\n", ptr, ptr->no, ptr->next);
        }
}

int main()
{
        int choice;
        L1: printf("\nEnter 1(Insert), 2(Delete), 3(Display), 4(Exit): ");
        scanf("%d", &choice);
        switch(choice)
        {
                case 1: insert(); goto L1;
                case 2: delete(); goto L1;
                case 3: display(); goto L1;
                case 4: break;
                default: printf("Wrong Choice\n"); goto L1;
        }
        return 0;
}
```