Implementation of Queue using Linked List

```
#include<stdio.h>
#include<stdlib.h>
int count=0;

struct node                         //Define a node of Queue
{
        int no;
        struct node *next;
}*new, *first=NULL, *ptr;

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

void insert_end()                   //Insert a node into Queue LILO/FIFO
{
        get_element();
        count = count+1;
        if(first==NULL) first=new;                              //For 1st Node
        else
        {
                for(ptr=first; ptr->next!=NULL; ptr=ptr->next); //Traverse
                ptr->next=new;                                  //Link the nodes
        }
        printf("Element inserted in Queue\n");
}

void create_list()                  //Insert 'n' nodes into Queue
{
        int n;
        printf("Enter the number of elements to be inserted: ");
        scanf("%d", &n);
        for(int i=1; i<=n; i++)
        {
                insert_end();
        }
}

void insert()               //Menu driven Insert procedure
{
        int choice;
        printf("\nEnter 1(Insert One), 2(Insert Multiple): ");
```

```
	scanf("%d", &choice);
	switch(choice)
	{
		case 1: insert_end(); break;
		case 2: create_list(); break;
		default: printf("Wrong Choice\n");
	}
}

void delete_begin()	//Delete one node from Queue FIFO/LILO
{
	if(first!=NULL)
	{
		if(first->next==NULL) first=NULL;		//When the Queue has one node
		else
		{
			ptr=first->next;
			first=ptr;				//Change the Head Node after every deletion
		}
		count = count-1;
		printf("Element is Deleted from Queue\n");
	}
	else printf("Queue is Empty\n");
}

void display()		//Traverse and fetch the elements stored in Queue
{
	if(first==NULL) printf("Queue is empty\n");
	else
	{
		printf("\nNo. of elements in Queue: %d\n", count);
		for(ptr=first; ptr!=NULL; ptr=ptr->next)				//Traverse
			printf("Block Address:%p, Data: %d, Next: %p\n", ptr, ptr->no, ptr->next);	//Fetch data
	}
}

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_begin(); goto L1;
		case 3: display(); goto L1;
		case 4: break;
		default: printf("Wrong Choice\n"); goto L1;
	}
```

```
        return 0;
}
```