//Queue Implementation using Linked List

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

//Definition of Queue Node
struct node
{
	int no;
	struct node *next;
}*front,*rear;				//front & rear are used as flag variables

void insert();
void delete();
void display();

void main()
{
	int ch;
	int repeat=1;
	front=rear=NULL;			//Queue empty

	while(repeat)
	{
		printf("\nQueue implementation using Linked List");
		printf("\nMenu:\n(1)Insert, (2)Delete, (3)Display, (4)Exit");
		printf("\nEnter your choice:  ");
		scanf("%d",&ch);
		switch(ch)
		{
			case 1:
				insert();
				break;
			case 2:
				delete();
				break;
			case 3:
				display();
				break;
			case 4:
				printf("\n\nExit....!!!!\n");
				exit(0);
			default:
				printf("\n Wrong choice...!!!! " );
		 }
	 }
}
```

```
//Function to insert one element into the Queue
void insert()
{
	int data;
	struct node *nw;

	printf("\nQueue Insert Operation...\n");
	printf("Enter the data: ");
	scanf("%d",&data);

	//node construction
	nw=(struct node*)malloc(sizeof(struct node));
	nw->no=data;
	nw->next=NULL;
	printf("Node in Address[%u]: %d, %u", nw, nw->no, nw->next);

	//For insertion, Overflow is not checked as Queue can grow dynamically
	if(rear==NULL||front==NULL)			//for inserting first node
		front=rear=nw;				//update rear and front for first insertion
	else						//For inserting >1 node
	{
		rear->next=nw;				//link to end of list or last node
		rear=nw;				//update rear for further insertions
	}
}

//Function to delete one element from the Queue
void delete()
{
	struct node *temp;
	int data;

	if(front==NULL||rear==NULL)			//Check Queue is empty
		printf("\nQueue Under Flow");
	else						//If Queue not empty
	{
		temp=front;
		data=temp->no;
		printf("\n\nThe deleted element = %d",data);
		front=front->next;			//Increment front to next node
	}
}
```

```
//Function to display elements into the Queue
void display()
{
	struct node *temp;
	temp=front;

	if(front==NULL||rear==NULL)			//Check Queue is empty
		printf("\nQueue is empty");
	else
	{
		printf("\n\nQueue is as follows");
		//Traverse the Queue
		while(temp!=NULL)
		{
			printf("\nAddress[%u]: %d, %u",temp, temp->no, temp->next);
			temp=temp->next;		//Traverse to next node
		}
	}
}
```