

Aim:

Write a program to implement queue using **linked lists**.

Sample Input and Output:

```

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 1
Enter element : 57
Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 1
Enter element : 87
Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 5
Queue size : 2
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 3
Elements in the queue : 57 87
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 2
Deleted value = 57
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 2
Deleted value = 87
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 3
Queue is empty.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 5
Queue size : 0
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 6

```

Source Code:QueueUsingLL.c

```

#include <conio.h>
#include <stdio.h>
#include "QueueOperationsLL.c"
int main() {
    int op, x;
    while(1) {
        printf("1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op) {
            case 1:
                printf("Enter element : ");
                scanf("%d",&x);
        }
    }
}

```

```

        enqueue(x);
        break;
    case 2:
        dequeue();
        break;
    case 3:
        display();
        break;
    case 4:
        isEmpty();
        break;
    case 5:
        size();
        break;
    case 6: exit(0);
}
}
}

```

QueueOperationsLL.c

```

#include<stdio.h>
#include<stdlib.h>
int count;
struct node
{
    int data;
    struct node*next;
};
struct node *front=NULL,*rear=NULL;
int enqueue(int x)
{
    struct node *newnode;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=x;
    newnode->next=NULL;
    if(front==NULL && rear==NULL)
    {
        front=rear=newnode;
        count++;
        printf("Successfully inserted.\n");
    }
    else
    {
        rear->next=newnode;
        rear=newnode;
        count++;
        printf("Successfully inserted.\n");
    }
}
void dequeue()
{
    if(front==NULL && rear==NULL)
    {

```

```

        printf("Queue is underflow.\n");
    }
    else if(front==rear)
    {
        printf("Deleted value = %d\n",front->data);
        front=rear=NULL;
        count--;
    }
    else
    {
        struct node *temp;
        temp=front;
        printf("Deleted value = %d\n",temp->data);
        front=front->next;
        count--;
        free(temp);
    }
}
void display()
{
    if(front==NULL && rear==NULL)
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Elements in the queue : ");
        struct node*temp=front;
        while(temp!=NULL)
        {
            printf("%d ",temp->data);
            temp=temp->next;
        }
        printf("\n");
    }
}
void isEmpty()
{
    if(front==NULL && rear==NULL)
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Queue is not empty.\n");
    }
}
void size()
{
    printf("Queue size : %d\n",count);
}

```

User Output

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 2

Enter your option : 2

Queue is underflow. 3

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 3

Enter your option : 3

Queue is empty. 4

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 4

Enter your option : 4

Queue is empty. 5

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 5

Enter your option : 5

Queue size : 0 1

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1

Enter your option : 1

Enter element : 44

Successfully inserted. 1

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1

Enter your option : 1

Enter element : 55

Successfully inserted. 1

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1

Enter your option : 1

Enter element : 66

Successfully inserted. 1

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1

Enter your option : 1

Enter element : 67

Successfully inserted. 3

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 3

Enter your option : 3

Elements in the queue : 44 55 66 67 2

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 2

Enter your option : 2

Deleted value = 44 2

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 2

Enter your option : 2

Deleted value = 55 5

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 5

Enter your option : 5

Queue size : 2 4

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 4

Enter your option : 4

Queue is not empty. 6

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 6

Enter your option : 6

Test Case - 2**User Output**

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1

```
Enter your option : 1
Enter element : 23
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 234
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 45
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 456
Successfully inserted. 2
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 2
Enter your option : 2
Deleted value = 23 3
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 3
Enter your option : 3
Elements in the queue : 234 45 456 2
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 2
Enter your option : 2
Deleted value = 234 3
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 3
Enter your option : 3
Elements in the queue : 45 456 4
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 4
Enter your option : 4
Queue is not empty. 5
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 5
Enter your option : 5
Queue size : 2 6
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit 6
Enter your option : 6
```