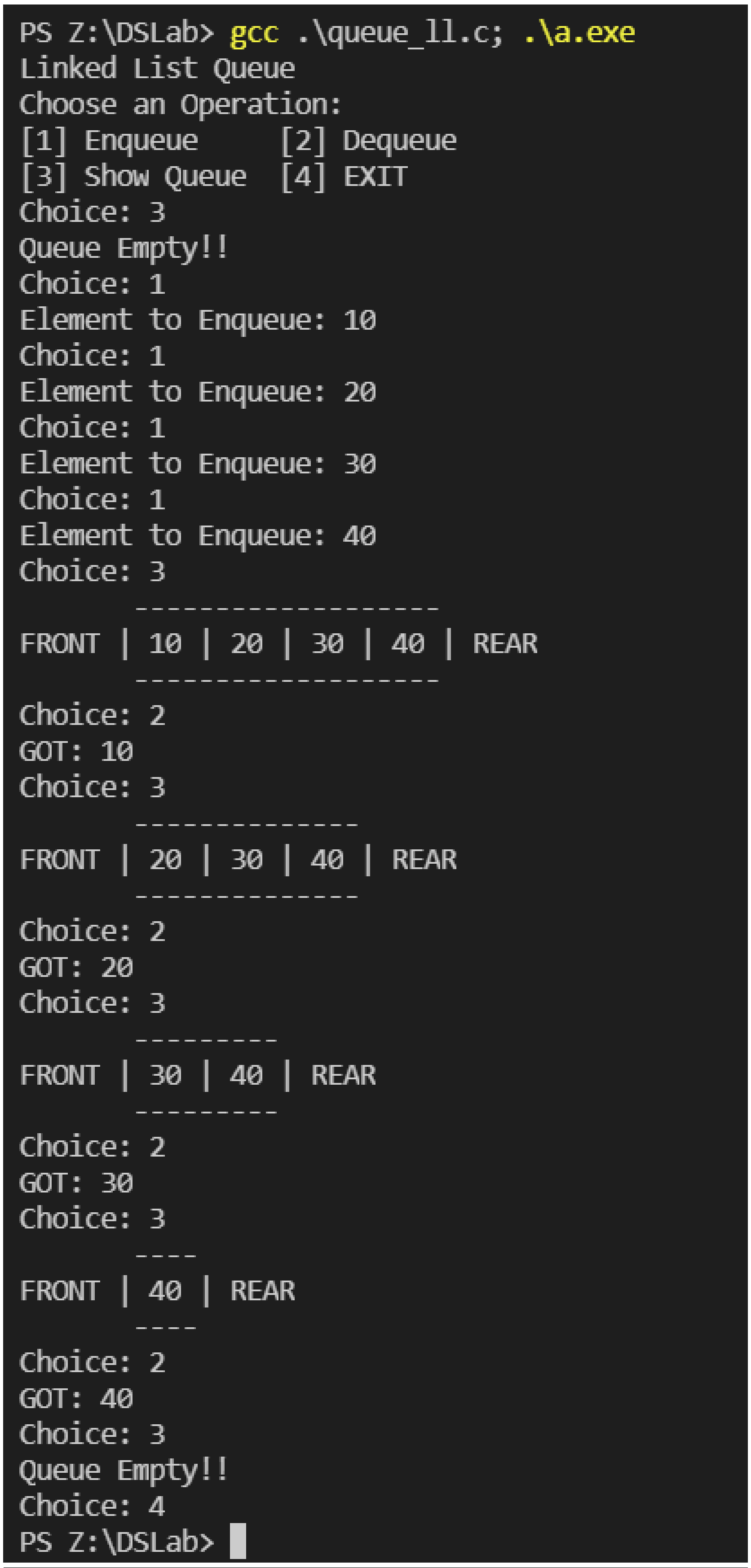
Code

```c
// Author: Gokul Raj, 235, R3A, CSE
// QUESTION: Implement a queue using Linked List.

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

// Structure to hold Singly Linked List Nodes for a Queue.
struct Node{
    int data;
    struct Node *next;
};

// FRONT and REAR of queue
struct Node *front = NULL;
struct Node *rear = NULL;

// Function to check if queue is empty
int isEmpty(){
    // returns 1 if empty, 0 otherwise
    if(front == NULL && rear == NULL)
        return 1;
    return 0;
}

// Enqueues element to rear
void enqueue(int ele){
    struct Node *newNode = malloc(sizeof(struct Node));
    if(isEmpty())
        front = newNode;
    else
        rear->next = newNode;
    newNode->next = NULL;
    newNode->data = ele;
    rear = newNode;
}

// Dequeues element from front
int dequeue(){
    struct Node *temp_n = front;
    int temp = temp_n->data;
    front = temp_n->next;
    // if dequeued last element, make rear NULL too.
    if(front == NULL)
        rear = NULL;
    free(temp_n);
    return temp;
}
```

```c
// Displays queue contents without affecting it.
void showQueue(){
    if(isEmpty()){
        printf("Queue Empty!!\n");
        return;
    }
    char *op = malloc(1024);
    char *tmp = malloc(1024);
    op[0] = '\0';
    for(struct Node *p = front; p != NULL; p = p->next){
        sprintf(tmp, "%d | ", p->data);
        strcat(op, tmp);
    }
    int len = strlen(op);
    len = len-1;
    op[len] = '\0';
    for(int i=0; i<len; i++)
        tmp[i] = '-';
    tmp[len] = '\0';
    printf("       %s\nFRONT | %s REAR\n       %s\n", tmp, op, tmp);
    free(op);
    free(tmp);
}

// Driver Code for UI
void main(){
    int choice, ele, looping = 1;
    printf("Linked List Queue\nChoose an Operation:\n[1] Enqueue\t[2] Dequeue\n[3] Show Queue\t[4]
EXIT\n");
    while(looping){
        printf("Choice: ");
        scanf("%d", &choice);
        switch(choice){
            case 1:
                printf("Element to Enqueue: ");
                scanf("%d", &ele);
                enqueue(ele);
                break;
            case 2:
                if(isEmpty()){
                    printf("Queue Empty!!\n");
                    break;
                }
                ele = dequeue();
                printf("GOT: %d\n", ele);
                break;
            case 3:
                showQueue();
                break;
            case 4:
                looping = 0;
                break;
            default:
                printf("Invalid Choice.. Try Again.\n");
        }
    }
}
```

Output



```
PS Z:\DSLab> gcc .\queue_ll.c; .\a.exe
Linked List Queue
Choose an Operation:
[1] Enqueue     [2] Dequeue
[3] Show Queue  [4] EXIT
Choice: 3
Queue Empty!!
Choice: 1
Element to Enqueue: 10
Choice: 1
Element to Enqueue: 20
Choice: 1
Element to Enqueue: 30
Choice: 1
Element to Enqueue: 40
Choice: 3
       -------------------
FRONT | 10 | 20 | 30 | 40 | REAR
       -------------------
Choice: 2
GOT: 10
Choice: 3
       --------------
FRONT | 20 | 30 | 40 | REAR
       --------------
Choice: 2
GOT: 20
Choice: 3
       ---------
FRONT | 30 | 40 | REAR
       ---------
Choice: 2
GOT: 30
Choice: 3
       ----
FRONT | 40 | REAR
       ----
Choice: 2
GOT: 40
Choice: 3
Queue Empty!!
Choice: 4
PS Z:\DSLab>
```