Linear Queue operations using a Linked List

Code:

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

typedef struct Node {
int data;     struct
Node *next;
} Node;

typedef struct {
Node *front;
   Node *rear;
} Queue;

void initializeQueue(Queue *q); int
isEmpty(Queue *q); void
enqueue(Queue *q, int value); int
dequeue(Queue *q);
void displayQueue(Queue *q);

int main() {
Queue q;
   initializeQueue(&q);

   int choice, value;

   while (1) {
      printf("\nQueue Operations:\n");
printf("1. Enqueue\n");        printf("2.
Dequeue\n");        printf("3. Display
Queue\n");        printf("4. Exit\n");
printf("Enter your choice: ");
      scanf("%d", &choice);

      switch (choice) {
case 1:
            printf("Enter the value to enqueue: ");
scanf("%d", &value);               enqueue(&q,
value);               break;           case 2:
            value = dequeue(&q);
            if (value != -1) {
               printf("Dequeued value: %d\n", value);
            }
break;
case 3:
            displayQueue(&q);
break;           case 4:
exit(0);           default:
            printf("Invalid choice! Please try again.\n");
      }
   }

   return 0;
}
void initializeQueue(Queue *q) {     q-
>front = NULL;
   q->rear = NULL;
}
int isEmpty(Queue *q) {
   return q->front == NULL;
}
void enqueue(Queue *q, int value) {     Node
*newNode = (Node *)malloc(sizeof(Node));
   if (!newNode) {
      printf("Memory allocation failed! Cannot enqueue.\n");
return;
   }
   newNode->data = value;
   newNode->next = NULL;

   if (isEmpty(q)) {        q->front =
q->rear = newNode;     } else {
      q->rear->next = newNode;
      q->rear = newNode;
   }
   printf("Enqueued %d into the queue.\n", value);
}
```

```
int dequeue(Queue *q) {
   if (isEmpty(q)) {
      printf("Queue is empty! Cannot dequeue.\n");
return -1;
   }

   Node *temp = q->front;
int value = temp->data;
   q->front = q->front->next;

   if (q->front == NULL) { // Reset rear when the queue becomes empty        q-
>rear = NULL;
   }

   free(temp);
   return value;
}
void displayQueue(Queue *q) {
   if (isEmpty(q)) {
      printf("Queue is empty!\n");
      return;
   }
   printf("Queue elements: ");
Node *current = q->front;
   while (current) {
      printf("%d ", current->data);
      current = current->next;
   }
   printf("\n");
}
```

OUTPUT:

```
Queue Operations:
1. Enqueue
2. Dequeue
3. Display Queue
4. Exit
Enter your choice: 1
Enter the value to enqueue: 1
Enqueued 1 into the queue.

Queue Operations:
1. Enqueue
2. Dequeue
3. Display Queue
4. Exit
Enter your choice: 1
Enter the value to enqueue: 2
Enqueued 2 into the queue.

Queue Operations:
1. Enqueue
2. Dequeue
3. Display Queue
4. Exit
Enter your choice: 1
Enter the value to enqueue: 3
Enqueued 3 into the queue.

Queue Operations:
1. Enqueue
2. Dequeue
3. Display Queue
4. Exit
Enter your choice: 3
Queue elements: 1 2 3

Queue Operations:
1. Enqueue
2. Dequeue
3. Display Queue
4. Exit
Enter your choice: 2
Dequeued value: 1
```