EXPERIMENT NUMBER – 4

AIM:

Create a linear queue using linked list and implement different operations such as insert, delete and display queue elements.

THEORY:

A linear queue operates on the First-In-First-Out (FIFO) principle, meaning the first element added to the queue will be the first one to be removed. This is analogous to a queue of people where the person who has been waiting the longest is the first to be served.

A linear queue can be implemented using either an array or a linked list. The key operations in a linear queue are:

1. **Enqueue**: Adding an element to the rear (or end) of the queue.
2. **Dequeue**: Removing an element from the front (or beginning) of the queue.

CODE:

INPUT:

```
#include<stdio.h>

#include<stdlib.h>

struct Node{

   int data;

   struct Node* next;

};

struct queue{

   struct Node* front;

   struct Node* rear;
```

```
};


struct Node* createNode(int data){

    struct Node* newNode=(struct Node*)malloc(sizeof(struct Node));

    newNode->data=data;

    newNode->next=NULL;

    return newNode;

}


struct queue* createQueue(){

    struct queue* q=(struct queue*)malloc(sizeof(struct queue));

    q->front=NULL;

    q->rear=NULL;

    return q;

}


void enqueue(struct queue* q, int data){

    struct Node* newNode = createNode(data);


    if(q->rear==NULL){

        q->front=q->rear=newNode;

        return;

    }
    q->rear->next=newNode;
    q->rear=newNode;
}

int dequeue(struct queue* q){
    if(q==NULL){
        printf("the Queue is empty.\n");
        return -1;
    }
    struct Node* temp=q->front;
    int data=temp->data;
    q->front=q->front->next;

    if(q->front==NULL){
        q->rear=NULL;
    }
    free(temp);
    return data;
}

void displayQueue(struct queue* q){
    if(q->front==NULL){
        printf("The queue is empty.\n");
        return;
    }


    struct Node* temp=q->front;
    printf("the elements in the queue are: \n");
    while(temp!=NULL){
        printf("%d \n",temp->data);
        temp=temp->next;
    }
    printf("\n");
}
int main(){


    struct queue* q= createQueue();
    enqueue(q,10);
    enqueue(q,20);
    enqueue(q,30);


    printf("the queue after insertion of elements: \n");
    displayQueue(q);


    printf("the dequeued element is %d \n",dequeue(q));
    printf("the queue after deletion of element: \n");
    displayQueue(q);


    enqueue(q,40);
    printf("the queue after insertion of another element: \n");
    displayQueue(q);


    return 0;
}
```

OUTPUT:

```
the queue after insertion of elements:
the elements in the queue are:
10
20
30
the dequeued element is 10
the queue after deletion of element:
the elements in the queue are:
20
30
the queue after insertion of another element:
the elements in the queue are:
20
```

30

40