AIM:-Implementation of circular queue using linked list

Algorithm:-

ENQUEUE(DATA)

Create a struct node type node.

Insert the given data in the new node data section and NULL in address section.

If Queue is empty then initialize front and rear from new node.

Queue is not empty then initialize rear next and rear from new node.

New node next initialize from front

DEQUEUE()

Check if queue is empty or not.

If queue is empty then dequeue is not possible.

Else Initialize temp from front.

If front is equal to the rear then initialize front and rear from null.

Print data of temp and free temp memory.

If there is more than one node in Queue then make front next to front then

initialize rear next from front.

Print temp and free temp.

PRINT()

Check if there is some data in the queue or not.

If the queue is empty print “No data in the queue.”

Else define a node pointer and initialize it with front.

Print data of node pointer until the next of node pointer becomes NULL.\*/

Program:-

#include <stdio.h>

// Declaration of struct type node

struct node

{

int data;

struct node \*next;

};

struct node \*front=-1;

struct node \*rear=-1;

// function to insert the element in the Queue

void enqueue(int x)

{

struct node \*newnode; // declaration of pointer of struct node type.

newnode=(struct node \*)malloc(sizeof(struct node)); // allocating the memory to the newnode

newnode->data=x;

newnode->next=0;

if(rear==-1) // checking whether the Queue is empty or not.

{

front=rear=newnode;

rear->next=front;

}

else

{

rear->next=newnode;

rear=newnode;

rear->next=front;

}

}

// function to delete the element from the queue

void dequeue()

{

struct node \*temp; // declaration of pointer of node type

temp=front;

if((front==-1)&&(rear==-1)) // checking whether the queue is empty or not

{

printf("\nQueue is empty");

}

else if(front==rear) // checking whether the single element is left in the queue

{

front=rear=-1;

free(temp);

}

else

{

front=front->next;

rear->next=front;

free(temp);

}

}

// function to get the front of the queue

int peek()

{

if((front==-1) &&(rear==-1))

{

printf("\nQueue is empty");

}

else

{

printf("\nThe front element is %d", front->data);

}

}

// function to display all the elements of the queue

void display()

{

struct node \*temp;

temp=front;

printf("\n The elements in a Queue are : ");

if((front==-1) && (rear==-1))

{

printf("Queue is empty");

}

else

{

while(temp->next!=front)

{

printf("%d,", temp->data);

temp=temp->next;

}

printf("%d", temp->data);

}

}

void main()

{

enqueue(34);

enqueue(10);

enqueue(23);

display();

dequeue();

peek();

}

 

