Program for implementation of circular queue using linked list

ALGORITHM:

Algorithm Enqueue:

if (FRONT == NULL) // Inserting in an Empty Queue

FRONT = REAR = newNode

end ifelse

REAR -> next = newNode // Inserting after the last element

REAR = newNode

end else

REAR -> next = FRONT

end Algorithm\_Enqueue

Algorithm Dequeue:

if(FRONT == NULL) // Condition for underflow

Print “Queue Underflow”

end Dequeue

end ifelse if(FRONT == REAR) // The queue contains only one node

temp = FRONT -> data

free(temp)

FRONT = FRONT -> next

REAR -> next = FRONT

end ifelse if (FRONT == N - 1) // If FRONT is the last node

front = 0 // Make FRONT as the first node

end if

end Dequeue

PROGRAM:

#include <stdio.h>

#include <stdlib.h>

/\* Structure for creating node and pointer \*/

struct node

{

int data;

struct node \*next;

} \*rear,\*front,\*temp,\*newNode;

/\* Initialising the pointers in the linked list \*/

void create()

{

front = rear = NULL;

}

/\* Function to insert elements in a circular queue \*/

void enqueue(int data)

{

newNode = (struct node\*) malloc(sizeof(struct node));

newNode -> data = data;

newNode -> next = NULL;

if(front == NULL)

front = rear = newNode;

else

{

rear -> next = newNode;

rear = newNode;

}

rear -> next = front; // rear always points to the front

}

/\* Function to delete an element from the circular queue \*/

int dequeue()

{

int x;

if (front == NULL)

{

return -1; //for empty queue

}

else if (front == rear) // Queue has only a single node

{

x = front->data;

delete front;

front = rear = NULL;

}

else

{

node \*temp = front;

x = temp -> data;

front = front -> next;

// Make rear point to the front node

rear -> next = front;

delete temp;

}

return x;

}

/\* Function to check if the queue is empty \*/

int empty()

{

if(front == NULL)

{

return 1;

}

else

return 0;

}

/\* Function to print the elements in the circular queue \*/

void display()

{

node \*temp = front;

printf("\nCIRCULAR QUEUE : ");

do

{

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

temp = temp->next;

}while (temp != front);

}

/\* Main function \*/

int main()

{

int num,choice;

while(1)

{

printf("\n\nQUEUE OPERATIONS\n\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n\n");

scanf("%d",&choice);

switch (choice)

{

case 1:

printf("\nEnter item : ");

scanf("%d",&num);

enqueue(num);

break;

case 2:

if(!(empty()))

printf("\nDequeued element : %d",dequeue());

else

printf("\nEMPTY QUEUE\n");

break;

case 3:

display();

break;

default: exit(0);

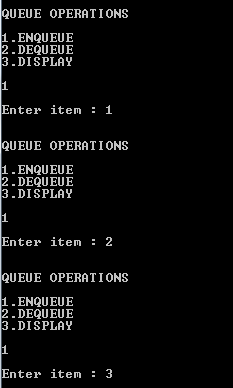
}}

return 0;

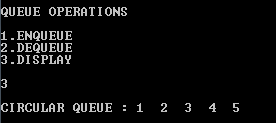
}

OUTPUT:

Enqueue elements



Display elements



Dequeue elements

