Practical-08

8C : Implement a Circular Queue and perform the Queue operations: Enqueue , Dequeue and Print using Menu Driver Program such as  1.Add, 2.Delete and 3. Print and 4. Exit.

//circular queue implementation

#include <stdio.h>

#include <stdlib.h>

#define MAX 5

struct CircularQueue {

int items[MAX];

int front, rear;

};

// Function prototypes

void enqueue(struct CircularQueue\* q, int value);

void dequeue(struct CircularQueue\* q);

void display(struct CircularQueue\* q);

void menu();

int main() {

struct CircularQueue q;

q.front = -1;

q.rear = -1;

int choice, value;

while (1) {

menu();

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:

dequeue(&q);

break;

case 3:

display(&q);

break;

case 4:

printf("Exiting program.\n");

exit(0);

default:

printf("Invalid choice, please try again.\n");

}

}

return 0;

}

// Function to display the menu

void menu() {

printf("\nCircular Queue using Array\n");

printf("1. Enqueue\n");

printf("2. Dequeue\n");

printf("3. Display\n");

printf("4. Exit\n");

}

// Function to enqueue an element

void enqueue(struct CircularQueue\* q, int value) {

if ((q->front == 0 && q->rear == MAX - 1) || (q->rear == (q->front - 1) % (MAX - 1))) {

printf("Queue is full\n");

return;

} else if (q->front == -1) { // Insert First Element

q->front = q->rear = 0;

q->items[q->rear] = value;

} else if (q->rear == MAX - 1 && q->front != 0) {

q->rear = 0;

q->items[q->rear] = value;

} else {

q->rear++;

q->items[q->rear] = value;

}

printf("Enqueued %d\n", value);

}

// Function to dequeue an element

void dequeue(struct CircularQueue\* q) {

if (q->front == -1) {

printf("Queue is empty\n");

return;

}

int data = q->items[q->front];

q->items[q->front] = -1;

if (q->front == q->rear) {

q->front = q->rear = -1;

} else if (q->front == MAX - 1) {

q->front = 0;

} else {

q->front++;

}

printf("Dequeued %d\n", data);

}

// Function to display the queue

void display(struct CircularQueue\* q) {

if (q->front == -1) {

printf("Queue is empty\n");

return;

}

printf("Queue elements: ");

if (q->rear >= q->front) {

for (int i = q->front; i <= q->rear; i++)

printf("%d ", q->items[i]);

} else {

for (int i = q->front; i < MAX; i++)

printf("%d ", q->items[i]);

for (int i = 0; i <= q->rear; i++)

printf("%d ", q->items[i]);

}

printf("\n");

}

Output:

