//Ankit Kumar 2K20/CO/72 QueueImplementation.c

#include <stdio.h>

#include <stdlib.h>

#define MAX\_SIZE 8

struct queue {

    int size;

    int rear;

    int front;

    int \*a;

};

int isempty(struct queue \*q) { return (q->rear == q->front); }

int isfull(struct queue \*q) { return ((q->rear + 1) % q->size) == q->front; }

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

    if (isfull(q)) printf("Queue is full!!\n");

    else {

        q->rear = (q->rear + 1) % q->size;

        q->a[q->rear] = data;

        printf("Enqueued!!\n");

    }

}

void dequeue(struct queue \*q) {

    if (isempty(q)) printf("Queue is empty!!\n");

    else {

        q->front = (q->front + 1) % q->size;

        printf("Dequeued!!\n");

    }

}

void display(struct queue \*q) {

    printf("Elements : ");

    for (int i = (q->front + 1) % q->size; i <= q->rear; i++)

        printf("%d ", q->a[i]);

    printf("\n");

}

int main() {

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

    q->size = MAX\_SIZE;

    q->rear = 0;

    q->front = 0;

    q->a = (int \*)malloc(MAX\_SIZE \* sizeof(int));

    int choice;

    printf("\n1. Enqueue\n2. Dequeue\n3. Display\n4. Exit\n");

    int x;

    do {

        printf("Pick an option : ");

        scanf("%d", &choice);

        switch (choice) {

        case 1:

            printf("Enter element to be enqueued : ");

            scanf("%d", &x);

            enqueue(q, x);

            break;

        case 2:

            dequeue(q);

            break;

        case 3:

            display(q);

            break;

        default:

            break;

        }

    } while (choice != 4);

    return 0;

}

