

Program 3b

Write A Program to simulate the working of a circular queue of integers using an array.
Provide the following operations: Insert, Delete & Display
The program should print appropriate messages for queue empty and queue overflow conditions

Code:

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 5

int queue[MAX];
int front = -1;
int rear = -1;

int isFull() {
    return (front == (rear + 1) % MAX);
}

int isEmpty() {
    return (front == -1);
}

void insert(int value) {
    if (isFull()) {
        printf("Queue Overflow: Unable to insert %d\n", value);
        return;
    }
    if (isEmpty()) {
        front = 0; // Set front to 0 if the queue is empty
    }
    rear = (rear + 1) % MAX;
    queue[rear] = value;
    printf("Inserted %d into the queue\n", value);
}

void delete() {
    if (isEmpty()) {
        printf("Queue Underflow: Unable to delete from the queue\n");
        return;
    }
    int deletedValue = queue[front];
    if (front == rear) {
        front = -1; // Queue becomes empty
        rear = -1;
    } else {
        front = (front + 1) % MAX;
    }
}
```

```

    printf("Deleted %d from the queue\n", deletedValue);
}

void display() {
    if (isEmpty()) {
        printf("Queue is empty\n");
        return;
    }
    printf("Queue elements: ");
    int i = front;
    while (1) {
        printf("%d ", queue[i]);
        if (i == rear) break;
        i = (i + 1) % MAX;
    }
    printf("\n");
}

int main() {
    int choice, value;

    while (1) {
        printf("\nCircular Queue Operations:\n");
        printf("1. Insert\n");
        printf("2. Delete\n");
        printf("3. Display\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insert(value);
                break;
            case 2:
                delete();
                break;
            case 3:
                display();
                break;
            case 4:
                exit(0);
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }

    return 0;
}

```

}

```
Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 85
Inserted 85 into the queue
```

```
Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 56
Inserted 56 into the queue
```

```
Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 85 56
```

```
Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
Deleted 85 from the queue
```

```
#include <stdio.h>
#include <stdlib.h>

void insert(int item, int *front, int *rear, int *size) {
    if (((*front + 1) % *size) == *front) {
        printf("Queue is Full\n");
        return;
    }
    if (*front == -1) {
        *front = 0;
    }
    *rear = (*rear + 1) % *size;
    q[*rear] = item;
    printf("Item %d inserted successfully\n", item);
}

int delete(int *q, int *front, int *rear, int *size) {
    if (*front == -1) {
        printf("Queue is empty\n");
        return;
    }
    int ele = q[*front];
    if (*front == *rear) {
        *front = -1;
        *rear = -1;
    }
    else {
        (*front) = (*front + 1) % *size;
        printf("Deleted successfully\n", ele);
    }
}

bool isfull(int *q, int *front, int *rear, int *size) {
    if ((*front + 1) % *size == *rear) {
        return true;
    }
    return false;
}

bool isempty(int *q, int *front, int *rear, int *size) {
    if (*front == -1) {
        return true;
    }
    return false;
}

int main() {
    int q[10];
    int front = -1, rear = -1, size = 10;
    int choice;
    while (1) {
        printf("Circular Queue Operations\n");
        printf("1. Insert\n2. Delete\n3. Display\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                int val;
                printf("Enter value to insert: ");
                scanf("%d", &val);
                insert(val, &front, &rear, &size);
                break;
            case 2:
                delete(q, &front, &rear, &size);
                break;
            case 3:
                printf("Queue elements: ");
                for (int i = front; i != rear; i = (i + 1) % size) {
                    printf("%d ", q[i]);
                }
                if (front == rear) {
                    printf("%d ", q[front]);
                }
                printf("\n");
                break;
            case 4:
                exit(0);
        }
    }
}
```

```
Output
Circular Queue Operations
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the element to insert: 85
Inserted 85

Circular Queue Operations
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 85

Circular Queue Operations
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 4
Code execution successful =

Circular Queue Operations
1. Insert
2. Delete
3. Display
4. Exit
```