Algorithm for Circular Queue.

// Circular queue implementation in C.

```
#define SIZE 5
int front = -1, rear = -1;

// check if the queue is full.
int isFull() {
if ((front == rear+1) || (front == 0 && rear == SIZE-1))
    return 1;
    return 0;
}
    // check if the queue is empty
int isEmpty() {
if (front == -1) return 1;
    return 0;
}
// Adding an element.
void enQueue(int element) {
printf("\n Queue is full! \n");
    else {
    if (front == -1) front = 0;
    rear = (rear+1) % SIZE;
    items[rear] = element;
    printf("\n Inserted -> %d", element);
  }
}
```

```
// Removing an element
int deQueue(){
int element;
if (isEmpty()){
printf("Queue is empty!");
return(-1);
} else {
element = items[front];
if (front == rear){
front = -1;
rear = -1;
}
else {
front = (front+1) % SIZE;
}
printf("Deleted element -> %d\n", element);
return(element);
}
}

// Display the queue
void display(){
int i;
if (isEmpty())
printf("Empty Queue");
else {
printf("Front -> %d", front);
printf("Items -> ");
for(i=front; i!=rear; i=(i+1)%SIZE){
printf("%d", items[i]);
}printf("%d", items[i]);
printf("Rear -> %d", rear);
```