3) (10 pts) ALG (Queues)

Suppose we wish to implement a queue using an array. The structure of the queue is shown below.

```
struct queue {
    int *array;
    int num_elements;
    int front;
    int capacity;
};
```

The queue contains the array and three attributes: the current number of elements in the array, the current front of the queue, and the maximum capacity. Elements may be added to the queue not just at the end of the array but also in the indices at the beginning of the array before front. Such a queue is called a circular queue.

Write a function to implement the dequeue functionality for the queue, while ensuring that no null pointer errors occur. Your function should take in 1 parameter: a pointer to the queue. Your function should return the integer that was dequeued. If the queue is NULL or if there are no elements to dequeue, your function should return 0.

```
int dequeue(struct queue * q) {

     // Check for NULL - 1 point
     if (q == NULL)
          return 0;

     // Check for no elements - 2 points
     if(q->num_elements == 0)
          return 0;

     // Obtain value to return - 1 point
     int retval = q->array[q->front];

     // Update front, with circular logic - 3 points
     q->front = (q->front + 1) % q->capacity;

     // Update number of elements - 2 points
     q->num_elements = q->num_elements - 1;

     // Returns value - 1 point
     return retval;
}
```