

**3) (5 pts) ALG (Stacks and Queues)**

Consider the following function:

```
void doTheThing(void)
{
    int i, n = 9; // Note: There are 9 elements in the following array.
    int array[] = {3, 18, 58, 23, 12, 31, 19, 26, 3};

    Stack *s1 = createStack();
    Stack *s2 = createStack();
    Queue *q = createQueue();

    for (i = 0; i < n; i++)
        push(s1, array[i]);

    while (!isEmptyStack(s1))
    {
        while (!isEmptyStack(s1))
            enqueue(q, pop(s1)); // pop element from s1 and enqueue it in q
        while (!isEmptyQueue(q))
            push(s2, dequeue(q)); // dequeue from q and push onto s2

        printf("%d ", pop(s2)); // pop from s2 and print element

        while (!isEmptyStack(s2))
            push(s1, pop(s2)); // pop from s2 and push onto s1
    }
    printf("Tada!\n");

    freeStack(s1);
    freeStack(s2);
    freeQueue(q);
}
```

What will be the exact output of the function above? (You may assume the existence of all functions written in the code, such as *createStack()*, *createQueue()*, *push()*, *pop()*, and so on.)

**Solution:** 3 18 58 23 12 31 19 26 3 Tada!

(This function just ends up printing the contents of the array in order.)

**Grading:**

**5 points** for the correct output

**4 points** if their output was simply missing the “Tada!” or if their output was off by one value

**2 points** if they printed the array in reverse order.

**0 points** otherwise.

Feel free to award partial credit if you encounter something else that seems reasonable.