Homework Assignment #2
Due: September 28, 2022 at 11:59 p.m.

1. In this question, you will design an implementation of a readable queue, which stores a sequence of elements and supports the following operations.

- ENQUEUE(x): adds item x to the end of the sequence and returns nil.
- DEQUEUE: removes and returns the first element of the sequence. (If the sequence is empty, this operation does not change the sequence and returns nil.)
- READ(i): returns the ith element of the sequence, where i is a positive integer. If i is greater than the number of elements in the sequence, this operation returns nil.

For EECS5101 students only: Your queue should also support one additional operation:

- REVERSE: reverses the order of all the elements in the sequence and returns nil.

Your implementation should use $O(n)$ words of memory space when the queue contains n elements. Assume an element of the queue can be stored in a single word. The amortized time per operation should be $O(1)$ if the queue is initially empty.

(a) Give a high-level description of your implementation in a couple of sentences.

(b) Give pseudocode for your implementation. Comment your code to explain what it is doing. If it is not obvious that one of the algorithms for your operations works, briefly explain why it is correct. Your pseudocode should be at a similar level of detail as the pseudocode given for algorithms in the textbook.

(c) Show that your implementation has $O(1)$ amortized time per operation when the queue is initially empty.

(d) Describe a simple way to ensure the amortized time per operation is $O(1)$, no matter what the initial state of the queue is.