1) (5 pts) ANL (Algorithm Analysis)

Consider an implementation of a queue with two stacks, A and B. If the last operation was an enqueue, all elements will be in stack A and the top of the stack represents the back of the queue. If the last operation was a dequeue, then all of the elements will be on stack B and the top of the stack will represent the front of the queue. For example, if we start with an empty queue and enqueue the items 6, 8, 2 and 4 in succession, the picture on the left is what both stacks look like. If we follow that up with a dequeue, we first must pop off each item from stack A and push each item onto stack B, and then remove the top element of the stack

4				
2				8
8		→		2
6				4
---------	---------		---------	---------
Stack A	Stack B		Stack A	Stack B

(a) (2 pts) What would be the total run-time, in terms of n, of n enqueue operations, followed by n dequeue operations, using this implementation. (Assume both stacks are implemented efficiently.) Please give your answer as a simplified Big-Oh answer.

As each enqueue operation adds to the top of Stack A, each of these will run in O(1) time, because they are in succession and we don't touch Stack B at all. There are n of these operations. The first dequeue will take O(n) time because all items are popped off of stack A and subsequently pushed onto stack B. Finally, each of the following n – 1 dequeue operations will each take O(1) time. Adding this all up we get nO(1) + O(n) + (n-1)O(1) = O(n).

O(n) **Grading: 2 pts all or nothing, no reasoning necessary.**

(b) (3 pts) What would be the total run-time, in terms of n, of n enqueue operations, followed by n more alternating enqueue and dequeue operations? Please give your answer as a simplified Big-Oh answer.

The first n+1 enqueue operations take O(n) time total, as described above. If dequeues and enqueues are then alternated, each with a queue size of either n+1 or n, then every single one of these operations will take O(n) time because they will involve either transferring n+1 items from Stack A to Stack B, or transferring n items from Stack B to Stack A. Since there are n of these operations, our total time is O(n) + nO(n) = O(n^2).

O(n^2) **Grading: 3 pts all or nothing, no reasoning necessary.**