/* Nishant Puri
 * np2577
* HW2 Programming Q2
 */

Runtime Analysis of TwoStackQueue

1. Enqueue: O(1)

```
public void enqueue(AnyType x){
        S1.push(x);
    }
```

   The enqueue method cosists of one call to S1.push. Thus the runtime is the same as runtime for push() method in MyStack class. MyStack class pushes an element on the stack by adding it to the begining of its LinkedList(instance variable of MyStack class). Since the cost of adding an element at index 0 of a LinkedList is O(1) operation the runtime of push() is also O(1). Thus runtime of enqueue is also O(1)

2. Dequeue: Worst Case: O(n), Best Case: O(1)

```
public AnyType dequeue() {
if (!S2.isEmpty()) {
        return S2.pop();
} else if (!S1.isEmpty()) {
        transfer();
        return S2.pop();
} else {
        System.out.println("Queue Empty");
}
return null;
}
```

   If S2 is non-empty then dequeue is a single call to the push menthod of MyStack. For MyStack this is a O(1) operation because removing an element from index 0 of LinkedList is O(1) operation. However in the worst case all the n elements in the queue are on Stack 1 and Stack2 is empty. In such a scenario the transfer() method is called which loops through S1 and one-by-one pushes an element off it and puts it on S2. This takes linear time in size of Stack1(and thus the size of the queue) since there is one push and put operation per element in the queue.