Daily Coding Problem #53

Problem

This problem was asked by Apple.

Implement a queue using two stacks. Recall that a queue is a FIFO (first-in, first-out) data structure with the following methods: enqueue, which inserts an element into the queue, and dequeue, which removes it.

Solution

We can implement this by noticing that while a stack is LIFO (last in first out), if we empty a stack one-by-one into another stack, and then pop from the other stack we can simulate a FIFO (first in first out) list.

Consider enqueuing three elements: 1, 2, and then 3:

```
stack1: [1, 2, 3]
stack2: []
```

Then emptying stack1 into stack2:

```
stack1: []
stack2: [3, 2, 1]
```

Then dequeuing three times:

```
1
2
3
```

Which retains the original order. So when enqueuing, we can simply push to our first stack. When dequeuing, we'll first check our second stack to see if any residual elements are there from a previous emptying, and if not, we'll empty all of stack one into stack two immediately so that the order of elements is correct (we shouldn't empty some elements into stack two, pop only some of

them, and then empty some more, for example).

```
class Queue:
    def __init__(self):
        self.s1 = []
        self.s2 = []

    def enqueue(self, val):
        self.s1.append(val)

    def dequeue(self):
        if self.s2:
            return self.s2.pop()
        if self.s1:
            # empty all of s1 into s2
            while self.s1:
                self.s2.append(self.s1.pop())
            return self.s2.pop()
        return None
```