Construct a MinMax Stack class for a Min Max Stack. The class should support:

1) Pushing and popping values on and off the Stack

2) Peeking at the value at the top of the stack

3) Getting both the min and max values in the stack at any given point in time

```
// O(1) time | O(n) space
class MinMaxStack {
  constructor() {
    this.stack = [];
    this.minMaxStack = [];
  }

  // O(1) time | O(1) space
  peek() {
    return this.stack[this.stack.length - 1];
  }

  // O(1) time | O(1) space
  pop() {
    this.minMaxStack.pop();
    return this.stack.pop();
  }

  // O(1) time | O(1) space
  push(number) {
    const newMinMax = { min: number, max: number };
    if (this.minMaxStack.length) {
      const lastMinMax = this.minMaxStack[this.minMaxStack.length - 1];
      newMinMax.min = Math.min(lastMinMax.min, number);
      newMinMax.max = Math.max(lastMinMax.max, number);
    }
    this.minMaxStack.push(newMinMax);
    this.stack.push(number);
  }

  // O(1) time | O(1) space
  getMin() {
    return this.minMaxStack[this.minMaxStack.length - 1].min;
  }

  // O(1) time | O(1) space
  getMax() {
    return this.minMaxStack[this.minMaxStack.length - 1].max;
  }
}
```

Idea: Have a min max stack and our actual stack. When pushing values onto a stack, we check if we should update the minMax stack's current min and max values. We then push the new minMax object (regardless if it changed or not) on to the minMax stack

Time: All methods are constant time look ups due to the nature of a stack. Since we always have the min max values at the top of our stack, getting both is constant time

Space: Constant space for all the methods. However, the stack's are O(3n) (2 arrays 1 object) or O(n) since we store every number in the stack