Roman Converter - Test Automation

In general, I haven't encountered any problems when performing the test cases we outlined in the previous post for the website to convert Roman numerals to integers and vice versa.
Except for the case where we tried to convert a Roman numeral to an integer, when we tried to test what would happen if no data was submitted. In that case, it was necessary to modify the original code of the romanToInteger function and add an if statement to test this exception.

romanToInteger - Roman to Decimal values

✓ should convert "VI" to 6

```
expect(romanToInteger('VI')).to.equal(6);
```

✓ should convert "XI" to 11

```
expect(romanToInteger('XI')).to.equal(11);
```

✓ should convert "DCCXXXVII" to 737

```
expect(romanToInteger('DCCXXXVII')).to.equal(737);
```

✓ should convert "MMMDLVII" to 3557

```
expect(romanToInteger('MMMDLVII')).to.equal(3557);
```

✓ should convert "MMMDCCCLXXXVIII" to 3888

```
expect(romanToInteger('MMMDCCCLXXXVIII')).to.equal(3888);
```

✓ should convert "MMM" to 3000

```
expect(romanToInteger('MMM')).to.equal(3000);
```

✓ should throw error for "abc"

```
expect(() => romanToInteger('abc')).to.throw("Input must be a valid Roman numeral.");
```

✓ should throw error for empty input

```
expect(() => romanToInteger('')).to.throw("Input must be a valid Roman numeral.");
```

✓ should throw error for "IIII"

```
expect(() => romanToInteger('IIII')).to.throw("Input must be a valid Roman numeral.");
```

✓ should throw error for "IC"

```
expect(() => romanToInteger('IC')).to.throw("Input must be a valid Roman numeral.");
```

✓ should throw error for "VIIII"

```
expect(() => romanToInteger('VIIII')).to.throw("Input must be a valid Roman numeral.");
```

integerToRoman - Decimal to Roman values

✓ should convert 6 to "VI"

```
expect(integerToRoman(6)).to.equal('VI');
```

✓ should convert 17 to "XVII"

```
expect(integerToRoman(17)).to.equal('XVII');
```

✓ should convert 545 to "DXLV"

```
expect(integerToRoman(545)).to.equal('DXLV');
```

✓ should convert 2468 to "MMCDLXVIII"

```
expect(integerToRoman(2468)).to.equal('MMCDLXVIII');
```

✓ should convert 3888 to "MMMDCCCLXXXVIII"

```
expect(integerToRoman(3888)).to.equal('MMMDCCCLXXXVIII');
```

✓ should convert 3000 to "MMM"

```
expect(integerToRoman(3000)).to.equal('MMM');
```

✓ should throw error for 4000

```
expect(() => integerToRoman(4000)).to.throw("The number must be between 1 and 3999.");
```

✓ should throw error for 0

```
expect(() => integerToRoman(0)).to.throw("The number must be between 1 and 3999.");
```

✓ should throw error for negative number

```
expect(() => integerToRoman(-7)).to.throw("The number must be between 1 and 3999.");
```

✓ should throw error for decimal number

```
expect(() => integerToRoman(5.57)).to.throw("The number must be an integer.");
```