

To Rome and Back

Background: This is a combination of two [LeetCode](#) challenges based on Roman Numerals. One function converts an integer to a Roman Numeral and the other converts a Roman Number to an integer. Both functions were used to evaluate a single input.

Challenge: [LeetCode - 12. Integer to Roman](#) & [13. Roman to Integer](#)

Resources: [freeCodeCamp](#), [W3Schools](#) and [MDN Web Docs](#)

Notes: The functions below are intended to be informational and are not meant to be the only way to write a function that converts Roman Numerals to integers and vice versa. In addition to the functions below, I employed other functions and methods to combine both functions into one, set conditions and facilitate user interaction.

```
// INTEGER TO ROMAN
function toRoman(num) {
  // create a map that contains an array of key-value pairs of integers – roman
  numerals
  const intMap = new Map([[1000, 'M'], [900, 'CM'], [500, 'D'], [400, 'CD'],
    [100, 'C'], [90, 'XC'], [50, 'L'], [40, 'XL'], [10, 'X'],
    [9, 'IX'], [5, 'V'], [4, 'IV'], [1, 'I']]);
  // check constraints
  if (num <= 0 || num > 3999) {
    return 0;
  } else {
    // initialize and empty string to store the roman numeral
    let roman = "";
    // loop through the string as long as the lengths is greater than 0
    while (num > 0);
    // access the key-value pairs of the map
    for (let [key, value] of intMap) {
      // if the number of the key is less than or equal to the num
      if (num >= key) {
        // add the value/roman numeral to the roman string
        roman += value;
        // subtract the value/roman numeral from the num
        num -= key;
        // break out of the loop
        break;
      };
    };
    // return the roman numeral
  }
}
```

```

        return roman;
    };
};

```

// TESTING THE FUNCTION

```

console.log(toInt("3")); // III
console.log(toInt("4")); // IV
console.log(toInt("9")); // IX
console.log(toInt("58")); //LVIII
console.log(toInt("1994")); // MCMXCIV

```

// ROMAN TO INTEGER

```

function toInt(str) {
    // get the length of the str and assign it
    let strLength = str.length
    // create a map that contains the key-value pairs of roman numerals - integers
    const romeMap = new Map([['I', 1], ['V', 5], ['X', 10], ['L', 50], ['C', 100], ['D', 500], ['M', 1000]]);
    // check constraints
    if (strLength <= 0 || strLength > 15) {
        return 0;
    }else {
        // assign the difference of the string's length and 1 to i
        let i = strLength - 1;
        // assign the value at that key to a variable
        let result = romeMap.get(str[i]);
        // loop through the string as long as the lengths is greater than 0
        while (i > 0);
            // get the value of the current key
            const current = romeMap.get(str[i]);
            // get the value of the previous key
            const previous = romeMap.get(str[i - 1]);
            // if the current value is less than the previous value
            if (previous >= current) {
                // add the previous value to the result
                result += previous;
            } else {
                // otherwise, subtract it
                result -= previous;
            };
        // decrement by 1 for each loop
    }
}

```

```
        i--;  
    };  
    // return the Integer  
    return result;  
};  
};  
  
// TESTING THE FUNCTION  
console.log(toInt("III")); // 3  
console.log(toInt("IV")); // 4  
console.log(toInt("IX")); // 9  
console.log(toInt("LVIII")); // 58  
console.log(toInt("MCMXCIV")); // 1994
```