

## Solution for Section #4

---

### Solution 1: Roman Numerals [courtesy of Eric Roberts]

```
/*
 * Function: romanToDecimal
 * -----
 * Traverses what we'll assume to be a well-formed Roman numeral
 * and returns the decimal equivalent.
 */
function romanToDecimal(symbolToValueMap, numeral) {
    let result = 0;
    for (let i = 0; i < numeral.length; i++) {
        let currentSymbol = numeral[i];
        let currentValue = symbolToValueMap[currentSymbol];
        if (currentValue === undefined) return -1; // improper numeral
        if (i < numeral.length - 1) {
            let nextSymbol = numeral[i + 1];
            let nextValue = symbolToValueMap[nextSymbol];
            if (nextValue === undefined) return -1; // improper numeral

            if (currentValue < nextValue) {
                result += nextValue - currentValue;
                i++; // Skip next letter
            } else {
                result += currentValue;
            }
        } else { // on last symbol? no next symbol!
            result += currentValue;
        }
    }

    return result;
}
```

## Solution 2: Scaling Recipes

```

/***
 * Function: findUnitOfMeasure
 * -----
 * Returns the index of the record within CONVERSIONS that houses
 * information about the named unit of measure.
 */
function findUnitOfMeasure(unit) {
    for (let i = 0; i < CONVERSIONS.length; i++) {
        if (CONVERSIONS[i].unit === unit) return i;
    }
    return -1; // unit of measure isn't one to be normalized
}

/***
 * Function: normalize
 * -----
 * Adjusts the supplied ingredient record so that the quantity
 * is expressed in the most appropriate unit of measure.
 */
function normalize(ingredient) {
    let index = findUnitOfMeasure(ingredient.unit);
    if (index === -1) return;
    for (let i = index - 1; i >= 0; i--) {
        ingredient.amount *= CONVERSIONS[i].amount;
    }
    ingredient.unit = CONVERSIONS[0].unit;
    for (let i = 0; i < CONVERSIONS.length - 1; i++) {
        if (ingredient.amount < CONVERSIONS[i].amount) break;
        ingredient.amount /= CONVERSIONS[i].amount;
        ingredient.unit = CONVERSIONS[i + 1].unit; // advance to higher unit
    } // bound is length - 1, as each iteration accesses i and i + 1
}

/***
 * Function: scale
 * -----
 * Scales the recipe so that the quantities expressed are enough for
 * precisely 'servings' servings.
 */
function scale(recipe, servings) {
    let scale = servings/recipe.servings;
    for (let i = 0; i < recipe.ingredients.length; i++) {
        recipe.ingredients[i].amount *= scale;
        normalize(recipe.ingredients[i]);
    }
    recipe.servings = servings;
}

```

## Solution 3: Transitioning to Python