1. Create a function that takes an array of objects (groceries) which calculates the total price and returns it as a number. A grocery object has a product, a quantity and a price, for example:

{

"product": "Milk",

"quantity": 1,

"price": 1.50

}

1. You go to a jewelry shop and try to find the most expensive piece of jewelry. You write down the name of each piece of jewelry and its price. Create a function that gets the name of the piece of jewelry with the highest price and returns "The most expensive one is the {name of jewelry piece}".

Example:

mostExpensive([{name:’DiamondEarings’, price: 980}, {name:’Gold watch’, price:250}, …])

- The most expensive one is the diamond Ring.

1. Given a word, create an object that stores the indexes of each letter in an array. Make sure the letters are the keys. Make sure the letters are symbols. Make sure the indexes are stored in an array and those arrays are values.

Examples:

mapLetters("dodo") ➞ { d: [0, 2], o: [1, 3] }

mapLetters("froggy") ➞ { f: [0], r: [1], o: [2], g: [3, 4], y: [5] }

mapLetters("grapes") ➞ { g: [0], r: [1], a: [2], p: [3], e: [4], s: [5] }

1. And who cursed the most in the fight between you and your spouse? Given an object with three rounds, with nested objects as your scores per round, return a string of who cursed the most: If you, return "ME!" If your spouse, return "SPOUSE!" If a draw, return "DRAW!"

Examples:

determineWhoCursedTheMost([{me: 10, spouse: 5 }, { me: 5, spouse: 10 },

{ me: 0, spouse: 10, }]) ➞ "DRAW!"

determineWhoCursedTheMost([{me: 40,spouse: 5}, {me: 9, spouse: 10}, { me: 9, spouse: 10}]) ➞ "ME!"

determineWhoCursedTheMost([{ me: 10,spouse: 5 }, { me: 9, spouse: 44 }, { me: 10, spouse: 55}]) ➞ "SPOUSE!"

1. Create a function that converts color in RGB format to Hex format.

Examples:

rgbToHex({red: 0, green: 128, blue: 192}) ➞ "#0080c0"

1. Create a function that takes an amount of monetary change (e.g. 47 cents) and breaks down the most efficient way that change can be made using USD quarters, dimes, nickels and pennies. Your function should return an object.

| **Coin** | **Value** |
| --- | --- |
| Penny | 0.01 |
| Nickel | 0.05 |
| Dime | 0.10 |
| Quarter | 0.25 |

Examples:

makeChange(47) ➞ { "q": 1, "d": 2, "n": 0, "p": 2 }

makeChange(24) ➞ { "q": 0, "d": 2, "n": 0, "p": 4 }

makeChange(92) ➞ { "q": 3, "d": 1, "n": 1, "p": 2 }

1. Create a function that takes an array of objects like { name: "John", notes: [3, 5, 4]} and returns an array of objects like { name: "John", avgNote: 4 }. If student has no notes (an empty array) then let's assume avgNote: 0.

Examples:

[

{ name: "John", notes: [3, 5, 4]}

]

➞

[

{ name: "John", avgNote: 4 }

]

1. Given an object with students and the grades that they made on the tests that they took, determine which student has the best Test Average. The key will be the student's name and the value will be an array of their grades. You will only have to return the student's name. You do not need to return their Test Average.

getBestStudent([{ name: ‘John’, grades: [100, 90, 80]}, {name: ‘Bob’, grades: [100, 70, 80]}…]) ➞ "John" // John's avg = 90 // Bob's avg = 83.33