Coding Challenges

Reverse Function - Write a function that reverses a new array (for loop+ push())

```
// Write a function that reverses a new array
const reverseArray = arr =>{ // arrow function
  let reverse = []; // empty array

  for(let i= arr.length -1; i>= 0; i--){  // loop backwards
    reverse.push(arr[i]);
  }
   return reverse
}
const sentence = ['sense.','make', 'all', 'will', 'This'];

console.log(reverseArray(sentence))
```

Loop through an array(arrow functions, for loop, unshift())

```
function reverseArray(arr){   function declaration
  let reverse = [],
      i;
  for (i = 0; i < arr.length; i++){
    reverse.unshift(arr[i]); // unshift: adds to the beginning of the array
  }
  return reverse;
}
const aliens = ["Blorgous", "Glamyx", "Wegord", "SpaceKing"];

greetAliens(aliens);
```

Add a word to all Array elements(arrow functions, for loop, push method())

Write a function, `convertToBaby()`, that takes in an array as an argument and, using a loop, returns a new array with each string in the array added with `'baby '`.

```
const convertToBaby = arr => {
  babyArr = [];
  for (let i = 0; i < arr.length; i++){
    babyArr.push(`baby ${arr[i]}`)
  }
  return babyArr
}

const animals = ['panda', 'turtle', 'giraffe', 'hippo', 'sloth', 'human'];

console.log(convertToBaby(animals))
```

A function that takes in an array of string using .forEach()

```
const veggies = ['broccoli', 'spinach', 'cauliflower', 'broccoflower'];

const politelyDecline = (veg) => {
       console.log(`No ${veg} please. I will have pizza with extra cheese.`);
}
// Write your code here:
const declineEverything = arr => arr.forEach(politelyDecline);

const acceptEverything = arr => arr.forEach(veggie => {
    console.log(`Ok, I guess I will eat some ${veggie}.`)
  }
  )
// alternative solution with for loop
const forloopEverything = arr => {
    for (i = 0; i<arr.length; i++){
      console.log(`Ok, I guess I will eat some ${arr[i]}.`)
    }
  }
```

Fix the bug – correct the nested loop

```
const numbers = [5, 3, 9, 30];

const smallestPowerOfTwo = arr => {
      let results = [];
      // The 'outer' for loop - loops through each element in the array
      for (let i = 0; i < arr.length; i++) {
            number = arr[i];

            // The 'inner' while loop - searches for smallest power of 2 great
er than the given number
            j = 1;      // the bug, change from i to j
            while (j < number) {
                  j = j * 2;    // change from i to j
            }
            results.push(j);  // change from i to j
      }
      return results
}
console.log(smallestPowerOfTwo(numbers))
// Should print the returned array [ 8, 4, 16, 32 ] instead prints the returne
d array [8]
```

Mapping an array of numbers to an array with square of each elements - .map())

```
const numbers = [2, 7, 9, 171, 52, 33, 14]

const toSquare = num => num * num

// Write your code here:
// Using an arrow function solution 1:
const squareNums = arr => arr.map(toSquare);

// Using an arrow function solution 2:
const squareNums = arr => arr.map(toSquare)

// Using an anonymous function:
```

```
const squareNums = arr => arr.map(num => num * num);

// A function declaration:
function squareNums(arr) {
    return arr.map(toSquare)
}
console.log(squareNums(numbers));
// expected output: Array [ 4, 49, 81, 29241, 2704, 1089, 196 ]
```

UpperCase or capitalize all the array values

Not only capitalize the array values but also add an exclamation point to the end: `'heya'` will become `'HEYA!'`

```
// Arrow function + .map():
const shoutGreetings = arr => arr.map(name => name.toUpperCase() + `!`);

// For loop:
const shoutGreetings = arr => {
  let shoutArray = []

  for(let i = 0; i < arr.length; i++){
    shoutArray.push(arr[i].toUpperCase() + `!`)
  }
  return shoutArray
}

// Function Declaration + map():
function shoutGreetings(arr) {
    return arr.map(name => name.toUpperCase() + `!`);
}
const greetings = ['hello', 'hi', 'heya', 'oi', 'hey', 'yo'];

console.log(shoutGreetings(greetings))
// Should print [ 'HELLO!', 'HI!', 'HEYA!', 'OI!', 'HEY!', 'YO!' ]
```

SortYears – sort all the array values in descending order using .sort()

```
// Arrow function & .sort():
const sortYears = arr => {
  return arr.sort((a, b) => b - a);
}

// Shorter syntax without return keyword:
const sortYears = arr => arr.sort((x, y) => b - a);

// Function expression:
const sortYears = function(arr){
  const years = (x, y) => y - x
  return arr.sort(years)
}
const years = [1970, 1999, 1951, 1982, 1963, 2011, 2018, 1922]

console.log(sortYears(years))
// Should print [ 2018, 2011, 1999, 1982, 1970, 1963, 1951, 1922 ]
```

Filter array elements to find common items in two arrays, using .filter(), .includes()

```
// Using .filter() method & .includes()
const justCoolStuff = (arr1, arr2) => {
  return arr1.filter(element => arr2.includes(element));
}
// Shorter syntax without return keyword
const justCoolStuff = (arr1, arr2) => arr1.filter(e => arr2.includes(e));

const coolStuff = ['gameboys', 'skateboards', 'backwards hats', 'fruit-by-the-
foot', 'pogs', 'my room', 'temporary tattoos'];

const myStuff = [ 'rules', 'fruit-by-the-
foot', 'wedgies', 'sweaters', 'skateboards', 'family-
night', 'my room', 'braces', 'the information superhighway'];

console.log(justCoolStuff(myStuff, coolStuff))
// Should print [ 'fruit-by-the-foot', 'skateboards', 'my room' ]
```

Array of objects: Find out whether or not every item in the array has entirely `plant`-based origins using .every()

```
const isTheDinnerVegan = arr => arr.every(lunch => lunch.source === "plant");

const dinner = [{name: 'hamburger', source: 'meat'}, {name: 'cheese', source:
'dairy'}, {name: 'ketchup', source:'plant'}, {name: 'bun', source: 'plant'}, {
name: 'dessert twinkies', source:'unknown'}];

console.log(isTheDinnerVegan(dinner))
// Should print false
```

Array of objects: sort the array in ascending order

```
const speciesArray = [ {speciesName:'shark', numTeeth:50}, {speciesName:'dog',
 numTeeth:42}, {speciesName:'alligator', numTeeth:80}, {speciesName:'human', n
umTeeth:32}];

// Arrow function & .sort():
sortSpeciesByTeeth = arr => arr.sort((a, b) => a.numTeeth - b.numTeeth);
// note: b - a is descending order
// As a function expression AND using a named function:
const sortSpeciesByTeeth = function (arr) {
      const compareTeeth = (a, b) => a.numTeeth - b.numTeeth
      return arr.sort(compareTeeth)
}
console.log(sortSpeciesByTeeth(speciesArray))
```

Array.prototype.findIndex()

The findIndex() method returns the **index** of the first element in the array **that satisfies the provided testing function**. Otherwise, it returns -1, indicating that no element passed the test.

```
const findMyKeys = arr => arr.findIndex(key => key === "keys");

// Feel free to comment out the code below to test your function

const randomStuff = ['credit card', 'screwdriver', 'receipt', 'gum', 'keys', '
used gum', 'plastic spoon'];

console.log(findMyKeys(randomStuff))
// Should print 4
```

Object methods, getters and setters

```
const dogFactory = (name, breed, weight) => {
    return {
      _name: name,
      _breed: breed,
      _weight: weight,
      get name() {
      return this._name;
      },
      set name(newName) {
        this._name = newName;
      },
      get breed() {
      return this._breed;
      },
      set breed(newBreed) {
        this._breed = newBreed;
      },
      get weight() {
      return this._weight;
      },
      set weight(newWeight) {
```

```
        this._weight = newWeight;
      },
      bark(){
        return `ruff! ruff!`;
      },
      eatTooManyTreats(){
         return this._weight++;
      }
  }
}
```

Coding Challenge – Intermediate

Write a function factorial() that takes a number as an argument and returns the factorial of the number.

```
function factorialize(num) {

  if (num === 0 || num === 1)
    return 1;
  // We start the FOR loop with i = 4
  // We decrement i after each iteration
  for (let i = num - 1; i >= 1; i--) {
        num *= i;
// We store the value of num at each iteration
  }
  return num;
}
factorialize(5);
```

solution 2:

```
// const factorial = function(num){
//   if(num === 1){
//     return num;
// }else{
//   return num * factorial ( num-1 );
// }
// }
// console.log(factorial(6))
```

```
const factorial = n => {
  let result = 1;

  for (let i=n; i>0; i--) {
    result *= i;
  }

  return result;
}
console.log(factorial(5));
```

Write a function subLength() that takes 2 parameters, a string and a single character.
The function should search the string for the two occurrences of the character and return the length between them including the 2 characters. If there are less than 2 or more than 2 occurrences of the character the function should return 0.

```
const subLength = (str, char) => {
    let charCount = 0;
    let len = -1;

    for (let i=0; i<str.length; i++) {
      if (str[i] == char) {
        charCount++;
        if (charCount > 2) {
          return 0;
        }
        if (len == -1) {
          len = i;
        } else {
          len = i - len + 1
        }
      }
    }
    if (charCount < 2) {
      return 0;
    }

    return len;
  };
```

Write a function groceries() that takes an array of object literals of grocery items.

The function should return a string with each item separated by a comma except the last two items should be separated by the word 'and'. Make sure spaces (' ') are inserted where they are appropriate.

```
const groceries = list => {
  let listString = ''

  for (let i=0; i<list.length; i++) {
    listString += list[i].item;
    if (i < list.length - 2) {
      listString += ', ';
    } else if (i == list.length - 2){
      listString += ' and ';
    }
  }

  return listString;
}
```

Below is Not part of the coding practice challenge!

Multiplication timetables

```
let multiplier = 5;

for (i = 0; i < 11; i++){
   let result = multiplier * i;
   console.log(`${multiplier} * ${i} = ${result}`);
// result
"5 * 0 = 0"
> "5 * 1 = 5"
> "5 * 2 = 10"
> "5 * 3 = 15"
> "5 * 4 = 20"
> "5 * 5 = 25"
> "5 * 6 = 30"
> "5 * 7 = 35"
> "5 * 8 = 40"
> "5 * 9 = 45"
> "5 * 10 = 50
```