Lab 05: JavaScript Functions

Objective(s):

1. Learn Higher-Order Function
2. Learn Closures
3. Learn forEach, Map, Filter & Reduce
4. Learn Some, Every, Find, FindIndex
5. Combining Iterators

Lab Task(s):

Exercises

1. Write a function called **`countdown`** that accepts a number as a parameter and every 1000 milliseconds decrements the value and console.log it. Once the value is 0 it should log "DONE!" and stop.
2. Write a function called **`isEven`** which takes in a number and returns true if the number is even and returns false if it is not

```
isEven(2); //
true
isEven(3); //
false
```

3. Write a function called **`isOdd`** which takes in a number and returns true if the number is odd and returns false if it is not

```
isOdd(3); // true
isOdd(14); //
false
```

4. Write a function called **isPrime** which takes in a number and returns true if the number is a prime number (is greater than 1 and can only be divided in whole by itself and 1), otherwise returns false

```
isPrime(8); //
false
isPrime(17); //
true
```

5. Write a function called **numberFact** which takes in a number and a callback and returns the result of the callback with the number passed to it

```
numberFact(59,isEven); //
false
numberFact(59,isOdd); //
true
numberFact(59,isPrime); //
true
```

6. Write a function called find. It should take in an array and a callback and return the first value found in the array that matches the condition.

```
find([8,11,4,27], function(val){return val >= 10}); // 11
find([8,11,4,27], function(val){return val === 5}); //
undefined
```

7. Write a function called **findIndex**. It should take in an array and a callback and return the index of first value found in the array that matches the condition.

```
// returns 1 (index of the first value greater than or
equal to 10)
findIndex([8,11,4,27], function(val){return val >= 10});

findIndex([8,11,4,27], function(val){return val === 7}); //
undefined
```

8. Write a function called **`specialMultiply`** which accepts two parameters. If the function is passed both parameters, it should return the product of the two. If the function is only passed one parameter - it should return a function which can later be passed another parameter to return the product. You will have to use closure and arguments to solve this.

```
specialMultiply(3,4); // 12
specialMultiply(3)(4); // 12
specialMultiply(3); // returns a function
```

9. Write a function called **`printFirstAndLast`** which accepts an array (of objects) and console.log a new string with the first character and the last character of each value.

```
printFirstAndLast(['awesome','example','of','forEach'])

// ae
// ee
// of
// fh
```

10. Write a function called **`addKeyAndValue`** which accepts three parameters, an array (of objects), a key and a value. This function should return the array of objects after each key and value have been added to each object in the array.

```
addKeyAndValue([{name: 'Elie'},{name: 'Tim'},{name: 'Elie'}], "isInstructor", true)

/*
[
    {
        name: 'Elie',
        isInstructor: true
    },
  {
    name: 'Tim',
    isInstructor: true
  },
  {
    name: 'Elie',
    isInstructor: true
  }
]
*/
```

11. Write a function called **`valTimesIndex`** which accepts an array of numbers and returns a new array with each value multiplied by the index it is at in the array:

```
valTimesIndex([1,2,3]) // [0,2,6]
valTimesIndex([5,10,15]) // [0,10,30]
```

12. Write a function called **`extractKey`** which accepts two parameters, an array of objects, and the name of a key and returns an array with just the values for that key:

```
extractKey([{name: "Elie", isInstructor:true},{name: "Tim", isInstructor:true},{name:
"Matt", isInstructor:true}], "name")

// ["Elie", "Tim", "Matt"]
```

13. Write a function called **`filterLetters`** which accepts an array of letters and returns the array of occurrences of a specific letter. This function should be case insensitive.
14. Write a function called **`filterKey`** which accepts two parameters, an array of objects, and the name of a key and returns an array with only those objects which have truthy values for that key:

```
filterKey([{name: "Elie", isInstructor:true, isHilarious: false},{name: "Tim", isInstructor:true,
isHilarious: true},{name: "Matt", isInstructor:true}], "isHilarious")

// [{name: "Tim", isInstructor:true, isHilarious:true}]
```

15. Write a function called **`addKeyAndValue`** which accepts three parameters, an array (of objects), a key and a value. This function should return the array of objects after each key and value has been added. You can do this a few ways, either by reducing starting with an empty array and making copies of the object or by starting with the actual array!

```
addKeyAndValue([{name: 'Elie'},{name: 'Tim'},{name: 'Elie'}], "isInstructor", true);

/*
[
    {
        name: 'Elie',
        isInstructor: true
    },
    {
        name: 'Tim',
        isInstructor: true
    },
    {
        name: 'Elie',
        isInstructor: true
    }
]
*/
```

16. Use the following object for this set of questions:

```
let users = [
 {
   username: 'larry',
   email: 'larry@foo.com',
   yearsExperience: 22.1,
   favoriteLanguages: ['Perl', 'Java', 'C++'],
   favoriteEditor: 'Vim',
   hobbies: ['Fishing', 'Sailing', 'Hiking'],
   hometown: {
     city: 'San Francisco',
     state: 'CA'
   }
 },
 {
   username: 'jane',
   email: 'jane@test.com',
   yearsExperience: 33.9,
   favoriteLanguages: ['Haskell', 'Clojure', 'PHP'],
   favoriteEditor: 'Emacs',
   hobbies: ['Swimming', 'Biking', 'Hiking'],
   hometown: {
     city: 'New York',
     state: 'NY'
   }
 },
 {
   username: 'sam',
   email: 'sam@test.com',
   yearsExperience: 8.2,
   favoriteLanguages: ['JavaScript', 'Ruby', 'Python', 'Go'],
   favoriteEditor: 'Atom',
   hobbies: ['Golf', 'Cooking', 'Archery'],
   hometown: {
     city: 'Fargo',
     state: 'SD'
   }
 },
 {
   username: 'anne',
   email: 'anne@test.com',
   yearsExperience: 4,
   favoriteLanguages: ['C#', 'C++', 'F#'],
   favoriteEditor: 'Visual Studio Code',
   hobbies: ['Tennis', 'Biking', 'Archery'],
   hometown: {
     city: 'Albany',
     state: 'NY'
   }
 },
 {
   username: 'david',
   email: 'david@test.com',
   yearsExperience: 12.5,
   favoriteLanguages: ['JavaScript', 'C#', 'Swift'],
   favoriteEditor: 'VS Code',
   hobbies: ['Volunteering', 'Biking', 'Coding'],
   hometown: {
     city: 'Los Angeles',
     state: 'CA'
   }
 }
];
```

   a. Write a function called printEmails which console.log's each email for the users.
   b. Write a function called printHobbies which console.log's each hobby for each user.
   c. Write a function called findHometownByState which returns the first user which has a hometown of the state that is passed in
   d. Write a function called allLanguages which returns an array of all of the unique values
   e. Write a function called hasFavoriteEditor which returns a boolean if any of the users have the editor passed in
   f. Write a function called findByUsername which takes in a string and returns an object in the users array that has that username

17. Write a function called vowelCount that accepts a string and returns an object with each key being the vowel and the value being the number of times the vowel occurs in the string (the order of keys in the object does not matter).
18. Write a function called removeVowels that accepts a string and returns an array of each character that is not a vowel (y should not count as a vowel for this function).

END