Lab 1

Re-submit Assignment

Due Sep 8 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip

CS-546 Lab 1

An Intro to Node

For this lab, you will be creating and running several functions to practice JavaScript syntax.

For this lab, you will make two files: `lab1.js` and `lab1.test.js` and submit them in a zip file that's named `LastName_FirstName.zip`. For example: Hill_Patrick.zip

You **should not** have any folders inside the zip file.

You **must** submit your files with the format specified, named as specified.

lab1.js

In this file, you will update the content of the functions and update the `firstName`, `lastName`, and `studentId` with the appropriate information. The function specifications are listed in the section below.

```
const questionOne = function questionOne(arr) {
    // Implement question 1 here
}

const questionTwo = function questionTwo(arr) {
    // Implement question 2 here
}

const questionThree = function questionThree(text) {
    // Implement question 3 here
}

const questionFour = function questionFour(num1, num2,num3) {
    // Implement question 4 here
}

module.exports = {
    firstName: "YOUR FIRST NAME",
    lastName: "YOUR LAST NAME",
    studentId: "YOUR STUDENT ID",
    questionOne,
```

```
    questionTwo,
    questionThree,
    questionFour
};
```

lab1.test.js

```
const lab1 = require("./lab1");

console.log(lab1.questionOne([1, 2, 3]));
// should output {'1': false, '2': true, '3': true}

console.log(lab1.questionTwo([1,2,3]));
// should output 2744

console.log(lab1.questionThree("The quick brown fox jumps over the lazy dog."));
// should output {consonants: 24, vowels: 11, numbers: 0, spaces: 8, punctuation: 1, specialCharacters: 0}

console.log(lab1.questionFour(25000, 3.11, 5));
// should output: 450.44
```

Functions to implement

questionOne(arr)

For your first function, you will calculate if all numbers in the array are prime numbers or not. You will return an `object` with the number as the key and true/false as the value That means that in `lab1.test.js`, running `lab1.questionOne([5, 3, 10])` would return `{5: true, 3: true, 10: false}`. If an empty array is passed in or if the function is called without any input parameters, just return an empty object. You do not have to worry about dealing with different data types passed in. You can assume only arrays and numbers as elements will be passed in to your function (we get to type checking and error handling in lecture 2)

To test this function, you will log the result of 5 calls to `lab1.questionOne([x, y, z])` with different inputs, like so:

```
console.log(lab1.questionOne([5, 3, 10]));
// {'5': true, '3': true, '10': false}

console.log(lab1.questionOne([2]));
// {'2': true}

console.log(lab1.questionOne([5, 10, 9]));
// {'5': true, '10': false, '9': false}

console.log(lab1.questionOne([2, 7, 9, 1013]));
// {'2': true, '7': true, '9': false, '1013': true}
```

```
console.log(lab1.questionOne([]));
// {}

console.log(lab1.questionOne());
// {}
```

questionTwo(arr);

This function will do a few things:

1. You will calculate the sum of squares for the elements in the array. For example `lab1.questionTwo([5, 3, 10])` would result in 134 for this step

2. You will then take that result, and then raise it to the 6th power: 134^6 which would result in 5789336458816

3. You will then take that result and square root it: $\sqrt{5789336458816}$ which will result in: 2406104 and that is the final value that your function will return.

If an empty array is passed in, just return `0`. You do not have to worry about dealing with different data types passed in. You can assume only arrays and numbers as elements will be passed in to your function (we get to type checking and error handling in lecture 2)

To test this function, you will log the result of 5 calls to `lab1.questionTwo([x, y, z])` with different inputs, like so:

```
console.log(lab1.questionTwo([5, 3, 10]));
//2406104

console.log(lab1.questionTwo([2]));
// 64

console.log(lab1.questionTwo([5, 10, 9]));
// 8741816

console.log(lab1.questionTwo([2, 7, 9, 10]));
// 12812904

console.log(lab1.questionTwo([]));
// 0
```

questionThree(str)

This function will return an `object` that contains the number of **consonants**, **vowels**, **numbers**, **spaces**, **punctuation, and any special characters** in the value `str`. For the purposes of this exercise, we are not counting `y` as a vowel, it would count as a consonant. your output would look like this: `{consonants: 20, vowels: 10, numbers: 7, spaces: 3, punctuation: 5, specialCharacters: 2}`

If an empty string is passed in, just return `0` for each key. You do not have to worry about dealing with different data types passed in. You can assume only strings will be passed in to your function (we get to type checking and error handling in lecture 2)

punctuation characters example : **. , ? ! ' " : ;** etc...

If you are in doubt if it's considered punctuation, you can always Google the character and punctuation or grammar. for example: " punctuation or grammar

Special Characters example: **# $ % & ^** etc..

To test this function, you will log the result of 5 calls to `lab1.questionThree(str)` with different inputs, like so:

```
console.log(lab1.questionThree("The quick brown fox jumps over the lazy dog."));
// {consonants: 24, vowels: 11, numbers: 0, spaces: 8, punctuation: 1, specialCharacters: 0}

console.log(lab1.questionThree("How now brown cow!!!"));
// {consonants: 10, vowels: 4, numbers: 0, spaces: 3, punctuation: 3, specialCharacters: 0}


console.log(lab1.questionThree("One day, the kids from the neighborhood carried my mother's groceries all the w
// {consonants: 61, vowels: 36, numbers: 0, spaces: 22, punctuation: 5, specialCharacters: 0}


console.log(lab1.questionThree("CS 546 is going to be fun & I'm looking forward to working with you all this se
// {consonants: 40, vowels: 23, numbers: 3, spaces: 17, punctuation: 3, specialCharacters: 1}

console.log(lab1.questionThree(""));
// {consonants: 0, vowels: 0, numbers:0, spaces: 0, punctuation: 0, specialCharacters: 0}
```

questionFour(num1, num2, num3)

This function will take three number inputs and then calculate the monthly payment of a loan. The first will be a loan amount, the 2nd will be the interest rate, the third will be the number of years for the term. You can look up the formula for calculating monthly payment on a loan on the internet if you do not know how to calculate it. You can assume that all input parameters will be supplied (none missing) and that they will all be numbers. You will take in the interest rate as a whole number percentage and then convert it to a decimal.

To test this function, you will log the result of 5 calls to `lab1.questionFour(num1, num2, num3)` with different inputs, like so:

```
console.log(lab1.questionFour(25000, 3.11, 5));
// Loan Amount: 25,000 , interest rate: 3.11% (0.0311), term: 5 years (5*12 = 60 monthly payments)
//Monthly Payment: 450.44
```

```
console.log(lab1.questionFour(30000, 5, 6));
//483.15

console.log(lab1.questionFour(19500, 7, 3));
//602.10

console.log(lab1.questionFour(55000, 2, 6));
//811.27

console.log(lab1.questionFour(33000, 4.5, 2));
//1440.38
```

Requirements

1. You will have to write each function
2. You must submit all files, zipped up, not contained in any folders
3. You must not use any npm dependenices in this lab