Functions Lab

- Due No Due Date
- Points 1
- Submitting a website url

[(https://github.com/learn-co-curriculum/phase-1-functions-lab)](https://github.com/learn-co-curriculum/phase-1-functions-lab) [(https://github.com/learn-co-curriculum/phase-1-functions-lab/issues/new)](https://github.com/learn-co-curriculum/phase-1-functions-lab/issues/new)

Learning Goals

- Practice writing functions
- Explain calling functions from within other functions
- Practice basics of control flow and `return` statements

Introduction

Scuber, our transportation company, has gained some traction among moms and dads in New York City. It has also received a few requests.

First, Scuber's executives want to ensure that Scuber's headquarters is near its customers. They would like you to write a function, `distanceFromHqInBlocks` , that takes in a pickup location for a passenger, and returns the *number of blocks* from Scuber headquarters on 42nd Street. For example, if the pickup location is `50` (i.e., 50th Street), the return value should be `8` .

The `distanceFromHqInBlocks` function's return value can then be used for another function, `distanceFromHqInFeet` , that translates the number of blocks from headquarters to the distance in feet. Each block in Manhattan is 264 feet long.

Next, customers want Scuber to calculate the number of feet travelled based on the distance. Write a function called `distanceTravelledInFeet` that takes in both the start and destination blocks, and returns the number of feet travelled. Remember, each block in Manhattan is 264 feet long. For example `distanceTravelledInFeet(34, 38)` representing 34th St to 38th St, returns 1056 (`(38-34)*264`). You can assume that we are only calculating distance uptown/downtown, not from river to river.

Getting Started

If you haven't already, fork and clone this lab into your local environment. Remember to **fork** a copy into your GitHub account first, then **clone** from that copy. Navigate into its directory in the terminal, then run `code .` to open the files in Visual Studio Code.

Instructions

To complete all of your work assignments, you'll need to write four functions:

- `distanceFromHqInBlocks` : Returns the number of blocks from Scuber's headquarters to the pickup location.

  ```
  function distanceFromHqInBlocks(someValue) {
    //returns the number of blocks given a value
  }
  ```

- `distanceFromHqInFeet` : Returns the number of feet from Scuber's headquarters to the pickup location. Use your `distanceFromHqInBlocks` function to help return the correct value here. Try something like this:

```
function distanceFromHqInFeet(someValue) {
  distanceFromHqInBlocks(someValue);
  // call the distanceFromHqInBlocks function from inside the distanceFromHqInFeet function,
  // passing the argument from distanceFromHqInFeet into distanceFromHqInBlocks

  // the return value of distanceFromHqInBlocks can then be used to calculate feet
}
```

- `distanceTravelledInFeet` : Calculates the number of feet a passenger travels given a starting block and an ending block — it only calculates distance North and South (uptown/downtown). It uses the knowledge that a block is 264 feet long.

  ```
  function distanceTravelledInFeet(start, destination) {
    //returns the number of feet traveled
  }
  ```

- `calculatesFarePrice` : Given the same starting and ending block as the previous test (*hint hint*), return the fare for the customer. The first four hundred feet are free. For a distance between 400 and 2000 feet, the price is 2 cents per foot (not including 400, which are free!). Then Scuber charges a flat fare of $25 for a distance over 2000 feet and under 2500 feet. Finally, Scuber does not allow any rides over 2500 feet — the function returns `'cannot travel that far'` if a ride over 2500 feet is requested.

```
function calculatesFarePrice(start, destination) {
  //returns the fare for the customer
}
```

Understanding the Tests

Run `npm test` and look at the tests currently breaking (remember you need to run `npm install` first). You will need to write four functions from scratch, and test them as you go to get them working. Remember to look through the tests, which are located in `test/indexTest.js` . It's a bit easier to think through the testing process in steps.

1. The first describe statement, `describe('index.js', function()` , tells us that the file being tested is the `index.js` file.
2. Our second describe statement, `describe('distanceFromHqInBlocks()', function()` , lets us know what the name of our function is. In this case, our test is expecting to find a function called `distanceFromHqInBlocks()` .
3. Our third statement, `it('returns a distance in blocks', function()` , describes what our function is doing. In this case, we're going to be calculating some kind of distance.
4. Finally, our last statement, `expect(distanceFromHqInBlocks(43)).to.equal(1)` , tells us what our test is expecting the result to be. There are a couple really important clues here. Our first one is the 43. That tells us that our function is expecting an argument to be passed in, so we'll need to specify some kind of variable. Our other clue is the 1, because that tells us what the result is. If we're passing in 43, we're expecting our function to calculate that it's 1 block from headquarters.

Try reading through all of the tests first, before you write any code! It will help you decide how to structure your code in the cleanest and most efficient manner.

After you have all the tests passing, remember to commit and push your changes up to GitHub, then submit your work to Canvas using CodeGrade.

Resources

- **MDN Docs - Functions (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Functions)**