Lab 03.01 - Solution:

1. Complete this function to convert the inputted **feet** value to meters and **return** the result. Conversion units: 39.37 inches = 1 meter;

```
function convertDistance(feet) {
    // convert feet to meters
}
```

2. Improve the previous function so that it can convert in both directions: feet-to-meters and meters-to-feet. You will need to provide some way for the function to know which conversion to do.

3. Make a function that:

    - takes one number as its input (argument)
    - squares that number (multiplies it by itself)
    - logs the answer to the Console. So, if you input 4, it logs 16. Run the function three times with different inputs.

```
function squareNumber(x) {
    console.log(x ** 2); // or: x * x;
}
squareNumber(4); // 16
squareNumber(5); // 25
squareNumber(6); // 36
```

4. Make another version of the previous function, so that:
    - instead of logging the answer, it returns the answer.
    - When you call the function, save the return value to a variable.
    - Log the variable to see if the function works. So, if you input 5, it returns 25. Run the function three times with different inputs.

```
function squareNum(x) {
    return x * x; // or: x ** 2;
}

let sq4 = squareNum(4);
console.log(sq4); // 16

let sq9 = squareNum(9);
console.log(sq9); // 81

let sq12 = squareNum(12);
console.log(sq12); // 144
```

5. Make a function that:
    - takes one number as its input (argument)
    - cubes the input number
    - returns the answer So, if you input 4, you get back 64. Run the function three times with different values.

```
function cubeIt(x) {
    let cube = x ** 3;
    return cube;
}

let cube4 = cubeIt(4);
console.log(cube4); // 64

let cube5 = cubeIt(5);
console.log(cube5); // 125

let cube6 = cubeIt(6);
console.log(cube6); // 216
```

6. Make another version of the previous function that:
    - takes one number as its input (argument)
    - if the number is even, it squares the number
    - but if the number is odd, it cubes the number
    - returns the answer. So, if you input 3, you get back 27. So, if you input 4, you get back 16. Run the function three times with different values.

```
function squareOrCubeIt(x) {
    let answer = 0;
    // if x is even, dividing by 2 yields remainder of 0
    if(x % 2 == 0) { // if x is even, square it
        answer = x * x;
    } else { // x is odd, so cube it
        answer = x ** 3;
    }
    return answer;
}

let a = squareOrCubeIt(5); // cube it
console.log(a); // 125

let b = squareOrCubeIt(6); // square it
console.log(b); // 36

let c = squareOrCubeIt(7); // cube it
console.log(c); // 343
```

7. The weight of anything on the moon is one-sixth its weight on earth. Make a function called **calcMoonWeight** that takes in an earth weight number as its argument, calculates the the moon weight equivalent, and returns that number. So if the input is 180, the output is 30.

```
    function calcMoonWeight(earthWt) {
        // convert earth wt on moon wt
        var  moonWt = earthWt / 6;
        return moonWt;
    }

    let myMoonWt = calcMoonWeight(192); // 32

    console.log(myMoonWt);
```

8. Declare a function called **introducePet**, that:
    - has four parameters: **pet**, **name**, **age** and **sound**, each of which gets set when the function is called.
    - uses string interploation to make a message that includes all four arguments.
    - returns a message, such that if the arguments are **cat**, **Fluffy**, **3** and **Meow**, the returned message is: **Meow! My name is Fluffy! I am a 3-year-old cat!**. Run the function three times, with different pet inputs each time.

```
    function introducePet(pet, name, age, sound) {
        let intro = `${sound}! My name is ${name}! I am a ${age}-year-old
${pet}`;
        return intro;
    }

    let petIntro1 = introducePet('cat', 'Fluffy', 3, 'Meow');
    console.log(petIntro1);
    // Meow! My name is Fluffy! I am a 3-year-old cat!

    let petIntro2 = introducePet('dog', 'King', 2, 'Woof');
    console.log(petIntro2);
    // Woof! My name is King! I am a 2-year-old cat!

    let petIntro3 = introducePet('canary', 'Tweety', 4, 'Chirp');
    console.log(petIntro3);
    // Chirp! My name is Tweety! I am a 4-year-old canary!
```

9. Declare a function **introduceSelf** with two parameters: **name** and **city**. The function does not have a return value, but instead logs a message, such as: **Hi! My name is Jill. I am from Miami!**

```
    function introduceSelf(name, city) {
        console.log(`Hi! My name is ${name}! I am from ${food}?`);
    }
```

```
// function expects two arguments
greetByNameAndFood('Sally', 'grapes');
```

10. Declare a function with two parameters, **num1** and **num2**. The function call passes in two arguments, both numbers.
    The function does the following math:
    - If the **num1** is greater than **num2**, subtract **num2** from **num1**
    - If **num1** is less than or equal to **num2**, add the numbers together.
      Return the answer. Run the function twice, once with the numbers being subtracted, the other with the numbers being added.

```
function addOrSubtractNums(num1, num2) {
    let answer = 0;
    if(num1 > num2) {
        answer = num1 - num2;
    } else {
        answer = num1 + num2;
    }
    return answer;
}

let answr1 = addOrSubtractNums(30, 20); // 10
let answr2 = addOrSubtractNums(20, 30); // 50
```

11. Given: two sides of a right triangle as global variables **sideA** and **sideB**

- Write a function with parameters **a** and **b**
  - Function uses the Pythagorean Theorem (**$a^2 + b^2 = c^2$**) to find the hypotenuse, **c**, of **a** and **b**.
  - Function returns **c**, the hyotenuse.
  - Call the function, passing in **sideA** and **sideB** as its two arguments.

```
let sideA = 3;
let sideB = 4;

function findHypotenuse(a, b) {
    let cSquared = a**2 + b**2;
    let c = Math.sqrt(cSquared);
    return c;
}

let c1 = findHypotenuse(sideA, sideB);
console.log(c1); // 5

let c2 = findHypotenuse(6, 8);
console.log(c2); // 10

let c3 = findHypotenuse(9, 12);
console.log(c3); // 15
```

12. Write a function that:

- takes in numbers of pennies, nickels, dimes and quarters
- calculates the total value of all coins
- returns the total as dollars and cents, to two decimal places and with dollar-sign

```
    function countCoins(pennies, nickels, dimes, quarters)  {
        let cents = (pennies) + (nickels * 5) + (dimes * 10) + (quarters *
25);
        return '$' + ((cents / 100).toFixed(2));
    }
    let money1 = countCoins(250, 50, 25, 10); // 2.50 + 2.50 + 2.50 + 2.50
= $10.00
    console.log(money1);
    let money2 = countCoins(100, 100, 100, 100); // 1 + 5 + 10 + 25 =
$41.00
    console.log(money2);
```

END Lab 03.01
SEE Lab 03.01 Solution