Practice exercise 6.1

1. Create a function that takes two parameters, adds the parameters together, and returns the result.

2. Set up two different variables with two different values.

3. Use your function on the two variables, and output the result using console.log.

4. Create a second call to the function using two more numbers as arguments sent to the function.

Solution:

```
// 1. Create a function that takes two parameters, adds them, and returns the result

function addNumbers(a, b) {

  return a + b;

}

// 2. Set up two different variables with two different values

let num1 = 10;

let num2 = 25;

// 3. Use your function on the two variables and output the result

let result1 = addNumbers(num1, num2);

console.log("Result of first addition:", result1);

// 4. Create a second call to the function with two more numbers

let result2 = addNumbers(50, 75);

console.log("Result of second addition:", result2);
```

Output:

```
Result of first addition: 35
Result of second addition: 125
```

Practice exercise 6.2

1. Create an array of descriptive words.

2. Create a function that contains a prompt asking the user for a name.

3. Select a random value from the array using Math.random.

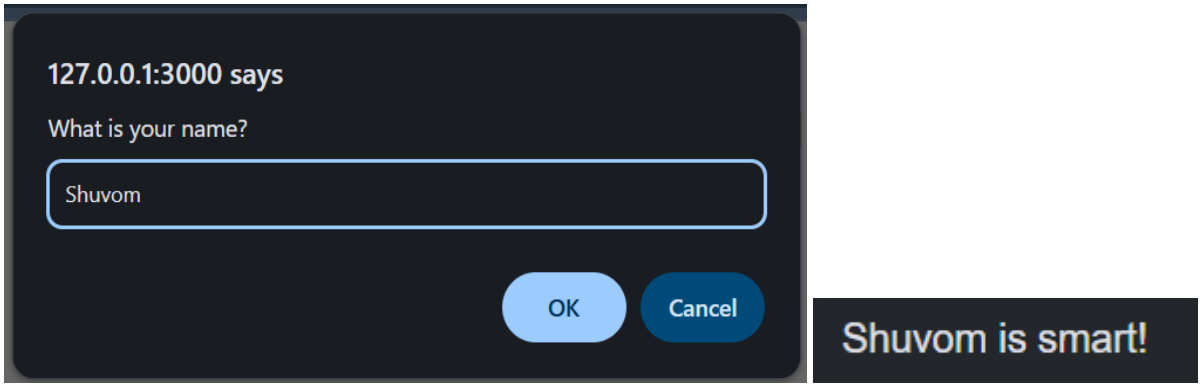
4. Output into the console the prompt value and the randomly selected array value.

5. Invoke the function.

Solution:

```
// 1. Create an array of descriptive words

let descriptiveWords = ["awesome", "brilliant", "kind", "funny", "creative", "smart", "adventurous"];

// 2. Create a function that asks the user for a name

function describePerson() {

  let name = prompt("What is your name?");

  // 3. Select a random value from the array

  let randomIndex = Math.floor(Math.random() * descriptiveWords.length);

  let randomWord = descriptiveWords[randomIndex];

  // 4. Output the name and the randomly selected word

  console.log(`${name} is ${randomWord}!`);

}

// 5. Invoke the function

describePerson();
```

Output:



Practice exercise 6.3

1. Set up two variables containing number values.

2. Set up a variable to hold an operator, either + or -.

3. Create a function that retrieves the two values and the operator string value within its parameters. Use those values with a condition to check if the operator is + or -, and add or subtract the values accordingly (remember if not presented with a valid operator, the function should default to addition).

4. Within console.log(), call the function using your variables and output the response to the console.

5. Update the operator value to be the other operator type—either plus or minus—and call to the function again with the new updated arguments.

Solution:

```
// 1. Set up two variables containing number values

let num1 = 20;

let num2 = 10;

// 2. Set up a variable to hold an operator, either + or -

let operator = "+";  // You can change this to "-" later
```

```
// 3. Create a function that takes two numbers and an operator
function calculate(a, b, op) {
  if (op === "+") {
    return a + b;
  } else if (op === "-") {
    return a - b;
  } else {
    // Default to addition if the operator is not valid
    return a + b;
  }
}
// 4. Call the function and log the result
console.log("Result:", calculate(num1, num2, operator));  // Should output 30
// 5. Update the operator and call the function again
operator = "-";
console.log("Result after updating operator:", calculate(num1, num2, operator));  // Should output 10
```

Output:

```
Result: 30
Result after updating operator: 10
```

Practice exercise 6.4

1. Set up an empty array to store the values that will be calculated within the loop.

2. Create a loop that runs 10 times, incrementing by 1 each time, creating two values each iteration. For the first value, multiply the

value of the loop count by 5. For the second value, multiply the value of the loop counter by itself.

3. Create a function that returns the value of the two parameters passed into the function when it is called. Add the values together, returning the result.

4. Within the loop, call the calculation function, passing in the two values as arguments into the function and storing the returned result in a response variable.

5. Still within the loop, push the result values into the array as it iterates through the loop.

6. After the loop is complete, output the value of the array into the console.

7. You should see the values [0, 6, 14, 24, 36, 50, 66, 84, 104, 126] for the array in the console.

Solution:

```
// 1. Set up an empty array to store calculated values
let resultsArray = [];
// 3. Create a function to return the sum of two parameters
function calculate(val1, val2) {
  return val1 + val2;
}
// 2. Create a loop that runs 10 times
for (let i = 0; i < 10; i++) {
  // Multiply loop counter by 5
  let value1 = i * 5;
  // Multiply loop counter by itself
  let value2 = i * i;
```

```
  // 4. Call the calculation function with both values
  let result = calculate(value1, value2);
  // 5. Push the result into the array
  resultsArray.push(result);
}
// 6. Output the value of the array
console.log("Calculated Results:", resultsArray);
```

Output:

```
Calculated Results: [
   0,  6, 14,  24,  36,
  50, 66, 84, 104, 126
]
```

Practice exercise 6.5

1. Create a variable value with let and assign a string value of 1000 to it.

2. Create an IIFE function and within this function scope assign a new value to a variable of the same name. Within the function, print the local value to the console.

3. Create an IIFE expression, assigning it to a new result variable, and assign a new value to a variable of the same name within this scope. Return this local value to the result variable and invoke the function. Print the result variable, along with the variable name you've been using: what value does it contain now?

Solution:

```
// 1. Create a variable value with let and assign a string value of 1000 to it
```

```
let value = "1000";
// 2. Create an IIFE (Immediately Invoked Function Expression)
(function() {
  let value = "500"; // local scope variable
  console.log("Inside first IIFE (local value):", value); // should print 500
})();
// 3. Create another IIFE assigned to a variable named result
let result = (function() {
  let value = "250"; // local to this IIFE
  return value;
})();
// Print both the outer variable and the result
console.log("Result from second IIFE:", result);  // should print 250
console.log("Global 'value' variable:", value);   // should still be 1000
```

Output:

```
Inside first IIFE (local value): 500
Result from second IIFE: 250
Global 'value' variable: 1000
```

Practice exercise 6.6

1. Create a function that contains a condition within it checking if the argument value is 0.

2. If the parameter is equal to 0, it should return the value of 1. Otherwise, it should return the value of the argument multiplied by the value returned from the function itself, subtracting one from the

value of the argument that is provided. This will result in running the block of code until the value reaches 0.

3. Invoke the function, providing an argument of whatever number you want to find the factorial of. The code should run whatever number is passed initially into the function, decreasing all the way to 0 and outputting the results of the calculation to the console. It could also contain a console.log() call to print the current value of the argument in the function as it gets invoked.

4. Change and update the number to see how it affects the results.

Solution:

```
// 1. Create a function that checks if the argument value is 0
function factorial(n) {
  console.log("Current value of n:", n); // To track recursive calls
  // 2. Base condition
  if (n === 0) {
    return 1;
  } else {
    // 2. Recursive case: n * factorial(n - 1)
    return n * factorial(n - 1);
  }
}
// 3. Invoke the function with a chosen number
let number = 5;  // You can change this value to test different numbers
let result = factorial(number);
// Output the final result
console.log(`Factorial of ${number} is:`, result);
```

Output:

```
Current value of n: 5
Current value of n: 4
Current value of n: 3
Current value of n: 2
Current value of n: 1
Current value of n: 0
Factorial of 5 is: 120
```

Practice exercise 6.7

1. Set the start variable at a value of 10, which will be used as the starting value for the loop.

2. Create a function that takes one argument, which is the countdown value.

3. Within the function, output the current value of the countdown into the console.

4. Add a condition to check if the value is less than 1; if it is, then return the function.

5. Add a condition to check if the value of the countdown is not less than 1, then continue to loop by calling the function within itself.

6. Make sure you add a decrement operator on the countdown so the preceding condition eventually will be true to end the loop. Every time it loops, the value will decrease until it reaches 0.

7. Update and create a second countdown using a condition if the value is greater than 0. If it is, decrease the value of the countdown by 1.

8. Use return to return the function, which then invokes it again and again until the condition is no longer true.

9. Make sure, when you send the new countdown value as an argument into the function, that there is a way out of the loop by using the return keyword and a condition that continues the loop if met.

Solution:

```
// 1. Set the start variable at a value of 10
let start = 10;
// 2. Create a function that takes one argument (countdown value)
function countdown(value) {
  // 3. Output the current countdown value
  console.log("Countdown:", value);
  // 4. Check if value is less than 1; if so, end the function
  if (value < 1) {
    console.log("Blast off! ");
    return;
  }
  // 6. Decrement the countdown value
  value--;
  // 5 & 8. If value is still >= 1, call the function again (recursive loop)
  return countdown(value); // 9. This keeps the loop going until condition fails
}
//  Start the countdown
countdown(start);
// 7. Create a second countdown function using "if value > 0"
function anotherCountdown(value) {
```

```
  if (value > 0) {
    console.log("Counting down again:", value);
    return anotherCountdown(value - 1);
  } else {
    console.log("Second countdown complete! ");
    return;
  }
}
//  Trigger second countdown
anotherCountdown(start);
```

Output:

```
Countdown: 10
Countdown: 9
Countdown: 8
Countdown: 7
Countdown: 6
Countdown: 5
Countdown: 4
Countdown: 3
Countdown: 2
Countdown: 1
Countdown: 0
Blast off! 🚀
Counting down again: 10
Counting down again: 9
Counting down again: 8
Counting down again: 7
Counting down again: 6
Counting down again: 5
Counting down again: 4
Counting down again: 3
Counting down again: 2
Counting down again: 1
Second countdown complete!
```

Practice exercise 6.8

1. Set a variable name and assign a function to it. Create a function expression with one parameter that outputs a provided argument to the console.

2. Pass an argument into the function.

3. Create the same function as a normal function declaration.

Solution:

```
// Normal function declaration
function displayMessageDeclaration(message) {
  console.log("Function Declaration:", message);
}
// Pass an argument into the function
displayMessageDeclaration("Hello from the function declaration!");
```

Output:

```
Function Expression: Hello from the function expression!
Function Declaration: Hello from the function declaration!
```