JavaScript Hands-On Lab
Module 2: Functions, Scope, and Closures

Instructor: Lebian Wilfried **Date:** July 2, 2025

Objective

By completing this lab, you will be able to define and use functions, understand scope and closures, and work with the 'this' keyword in JavaScript.

1 Function Basics

Task 1: Function Declaration

Create a function that greets a user by name.

```
function greet(name) {
    console.log("Hello, " + name + "!");
}

greet("Alice");
```

Task 2: Function Expression

Convert the above function into a function expression.

```
const greet = function(name) {
    console.log("Hello, " + name + "!");
};
greet("Bob");
```

Task 3: Arrow Function

Rewrite it using arrow syntax.

```
const greet = (name) => {
    console.log(`Hello, ${name}!`);
};
greet("Charlie");
```

2 Parameters and Return Values

Task: Create a calculator function

Write a function that takes two numbers and an operator, then returns the result.

```
function calculate(a, b, op) {
    if (op === "+") return a + b;
    if (op === "-") return a - b;
    if (op === "*") return a * b;
    if (op === "/") return a / b;
    return "Invalid operator";
}
console.log(calculate(4, 2, "*")); // 8
```

3 Scope

Task 1: Block Scope with `let`

```
let x = 10;
{
    let x = 5;
    console.log(x); // 5
}
console.log(x);     // 10
```

Task 2: Function Scope

```
function testScope() {
    var localVar = "I'm local";
    console.log(localVar);
}
testScope();
// console.log(localVar); // Uncomment to see the error
```

Task 3: Lexical Scope

```
function outer() {
    let outerVar = "Outer";

    function inner() {
        console.log(outerVar);
    }

    inner();
}
outer();
```

4 Closures

Task: Counter using Closure

Create a function that returns another function, keeping count:

```
function createCounter() {
    let count = 0;

    return function() {
        count++;
        console.log(count);
    };
}

const counter = createCounter();
counter(); // 1
counter(); // 2
counter(); // 3
```

Challenge: Create a 'createAdder(x)' function that returns a function which adds 'x' to any number passed to it.

5 The `this` Keyword

Task 1: Object with Method

```
const user = {
    name: "John",
    greet: function() {
        console.log("Hi, I'm " + this.name);
    }
};
user.greet(); // "Hi, I'm John"
```

Task 2: Arrow Function in Object

What happens if you use an arrow function?

```
const user = {
    name: "Alice",
    greet: () => {
        console.log("Hi, I'm " + this.name);
    }
};
user.greet(); // "Hi, I'm undefined"
```

Why? Arrow functions do not bind their own 'this'.

Submission Instructions

- Submit a '.js' file with each section clearly labeled in comments.
- Test each function. Add example outputs using 'console.log()'.
- **Let your instructor verify your outputs before submission.**
- Bonus tasks are encouraged but optional.