📌 Returning Values from Functions in JavaScript

In the last section, we learned how functions **accept arguments** to perform tasks dynamically. Now, we'll explore how functions can **return values** to the calling code.
Returning a value allows a function to **send data back** to where it was called, making it even more powerful.

◆ Example Without Return (No Output)

If a function **only performs an action** but does not return a value, we can't use the result elsewhere.

```
function greetUser(name) {
  alert("Hello, " + name + "!");
}

// Calling the function
greetUser("Alice"); // Displays an alert but doesn't return anything
```

✅ The function **displays an alert**, but we **cannot store** or use its output in another variable.

◆ Example With Return (Storing Output)

Let's modify the function to **return** a message instead of displaying an alert:

```
function greetUser(name) {
  return "Hello, " + name + "!";
}

// Calling the function and storing the result
var message = greetUser("Alice");

console.log(message); // Output: Hello, Alice!
```

✅ Now, instead of **just displaying a message**, the function **returns** it, so we can store it in a variable and use it later.

◆ Example: Calculating Order Total with Shipping

Let's write a function that **calculates the total cost** of an order, including shipping:

```
function calcTot(merchTot) {
  var orderTot;

  if (merchTot >= 100) {
    orderTot = merchTot; // Free shipping for orders $100+
  } else if (merchTot < 50.01) {
    orderTot = merchTot + 5; // Flat $5 shipping for orders under $50
  } else {
    orderTot = merchTot + 5 + (0.03 * (merchTot - 50)); // $5 + 3% of amount over $50
  }

  return orderTot; // Return the final total
}

// Calling the function and storing the result
var totalToCharge = calcTot(79.99);
```

```
console.log("Total amount to charge: $" + totalToCharge);
// Output: Total amount to charge: $75.60
```

✅ **How this works:**

1. The **order total** is calculated inside the function.
2. The function **returns the total amount** based on shipping rules.
3. The calling code **stores the returned value** in `totalToCharge`.
4. We **log the result** to the console.

◆ Understanding `return` in Functions

Concept	Explanation
`return`	Sends data back to the caller.
Variable stores returned value	`var totalToCharge = calcTot(79.99);`
Function can return anything	Strings, numbers, objects, or even other functions.
Function execution stops at `return`	Code after `return` **won't run**.

Example:

```
function testReturn() {
  return "This will return.";
  console.log("This won't run!"); // This line is ignored
}

console.log(testReturn()); // Output: This will return.
```

✅ **Important:** After `return`, the function **immediately stops execution**.

◆ Using a Function Inside Another Function

A function can **call another function** and use its return value:

```
function calcShip(price) {
  return price < 100 ? 5 : 0; // $5 shipping if price is under $100, otherwise free
}

function calcTot(price) {
  return price + calcShip(price); // Adds shipping to price
}

var total = calcTot(79.99);
console.log("Total cost: $" + total); // Output: Total cost: $84.99
```

✅ **Why this is useful?**

- The `calcShip` function **handles shipping costs separately**.
- The `calcTot` function **reuses** `calcShip` instead of repeating logic.
- This keeps the code **clean and modular**.

◆ Using a Function in an Expression

A function **can be used inside other calculations**:

```
var orderTot = merchTot + calcTax(merchTot);
```

✅ Here, `calcTax(merchTot)` **returns the tax**, which is then **added to `merchTot`**.

◆ Using a Function in Another Function Call

You can even pass a function **as an argument** to another function:

```
var tot = calc(merchTot, calcTax(merchTot));
```

✅ Here:

1. `calcTax(merchTot)` **returns tax value**.
2. That tax value **is passed as an argument** to `calc()`.
3. `calc()` receives **two parameters**: `merchTot` and the tax.

🎯 Key Takeaways

✓ **`return` sends data back** from a function to the calling code.
✓ You can store the **returned value in a variable** for later use.
✓ Functions can **call other functions** inside them.
✓ A function can be used **in expressions, function calls, or even alerts**.
✓ A function **always returns a single value** (even if it's an object or array).