**🔹 What This Code Is:**

You are creating **arrow functions** in JavaScript — a modern way of writing functions.

(a, b) => {

return a + b;

}

This is an **arrow function** that takes two parameters a and b, and returns their sum. But it’s **not assigned to a variable**, so it’s anonymous and not usable unless stored or immediately invoked.

**✅ Proper Assignment Example:**

const arrowMul = (a, b) => {

return a \* b;

};

Here:

* arrowMul is a **function variable**.
* It stores the **function itself**, **not the value** of a \* b.

That means:

🚫 arrowMul ≠ value of a \* b  
✅ arrowMul is a **function** that **returns** a \* b when you **call it**.

**🧠 Think of It Like This:**

**❌ Wrong Thinking:**

const arrowMul = a \* b  
This would actually try to multiply a and b immediately (which will error unless a and b exist at that time).

**✅ Correct Thinking:**

const arrowMul = (a, b) => a \* b  
This stores a **function**, which is a "machine" that takes inputs a and b, and gives you a \* b when you run it.

**✅ Usage Example:**

console.log(arrowMul(3, 4)); // Output: 12

Here, you're **calling** the arrowMul function with 3 and 4, and it returns 3 \* 4 = 12.

**🧠 Final Analogy:**

Think of arrowMul as a **calculator button** labeled "Multiply". It doesn’t hold a number. It **waits for inputs**, and when you press it (i.e., call it), it gives you the result.