Conditional Statements, Loops and Functions

THRIVE
Women for Women

Recap of Week 6

Introduction and session overview

Session Overview

1. Introduction and session overview
2. Conditional Statements
3. Switch Statements
4. Loops
5. Functions
6. Code Reusability
7. Hands-on Practice
8. Review & Wrap-up

Session Goals

THRIVE
Women for Women

Conditionals

Write conditional statements using if, else, and switch

Loops

Use loops to repeat tasks (for, while, do...while)

Functions

Write reusable code using functions

Why They Matter?

Building Blocks

Without them a program is just a static list of instructions that runs once from top to bottom.

Programs are **dumb**. Without logic, they can't react. We use conditionals and loops to turn them into dynamic systems.

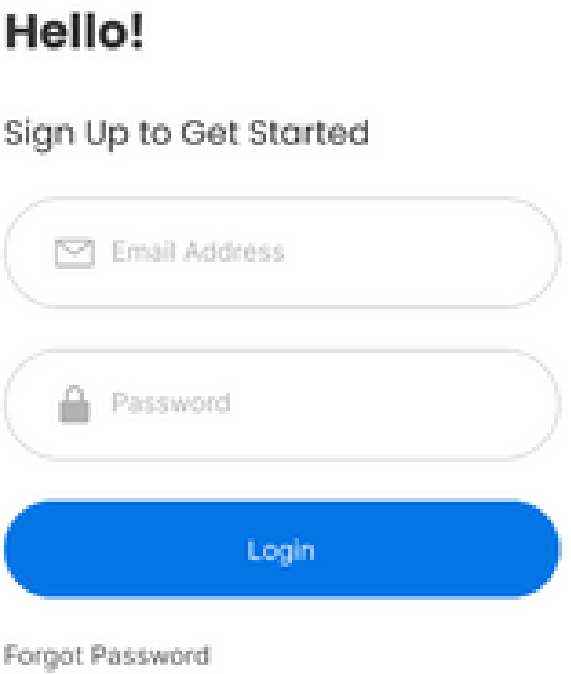
Conditional Statements

The Brain

Decision Making

Conditionals provide the logic required for interactivity. They allow programs to react differently depending on the situation.

IF password is correct → Access Granted
ELSE → Show Error Message



Boolean Expressions

Conditions must always
come down to True or False.

```
5 > 3       // true
10 === 2    // false
```

The “If” Statement

Code inside {} runs only if condition is true

```
if(isRaining) {
  bringUmbrella();
}
```

Handling Alternatives

THRIVE
Women for Women

Else & Else If

Else: Runs when the condition is false.

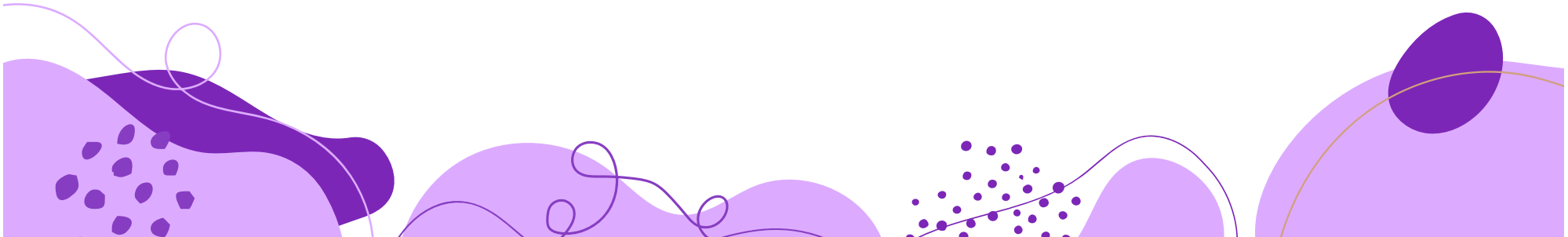
Else If: Checks a new condition. Order matters! The first match stops the rest.

```
if (score > 90) {
   grade = "A";
} else if (score > 80) {
   grade = "B";
} else {
   grade = "C";
}
```

The Ternary Operator

A shorthand for simple if...else statements.

condition ? true : false



Common mistakes in conditionals

❌ Using = instead of ===
❌ Using ==
❌ Wrong condition order
❌ Forgetting {}

Switch Statements

Switch Statments

Use Case

Used when checking one value against many possible specific values.

Break

Stops execution! Without it, code "falls through" to the next case.

Default

Acts like the final "else" if no cases match.

Switch Syntax Code

THRIVE
Women for Women

Clean & Readable
Instead of writing five else if blocks, use a switch! It's cleaner for checking single variables.

Tip: Always include a default case to catch unexpected values.

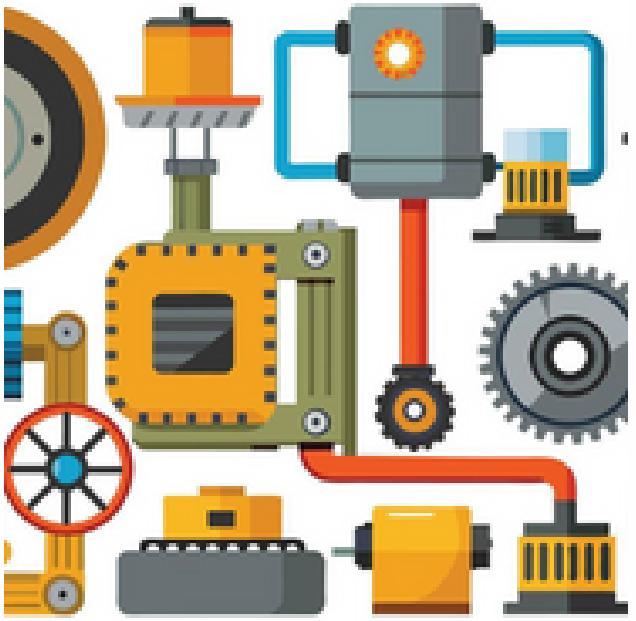
```
switch (hero) {
  case "Batman":
    console.log("Gadgets");
    break;
  case "Superman":
    console.log("Flight");
    break;
  default:
    console.log("Citizen");
}
```

Loops

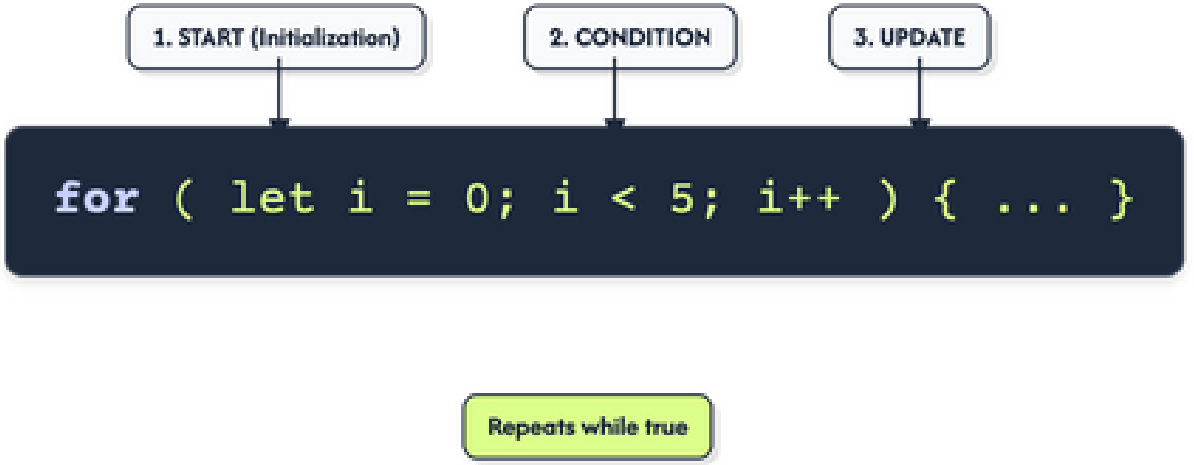
The Engine

Loops allow us to repeat code automatically. Why write the same line 100 times when you can loop it?

Use Cases:
- Counting
- Repeating calculations
- Processing lists of items



The For Loop Anatomy



While vs Do...While

THRIVE
Women for Women

While Loop

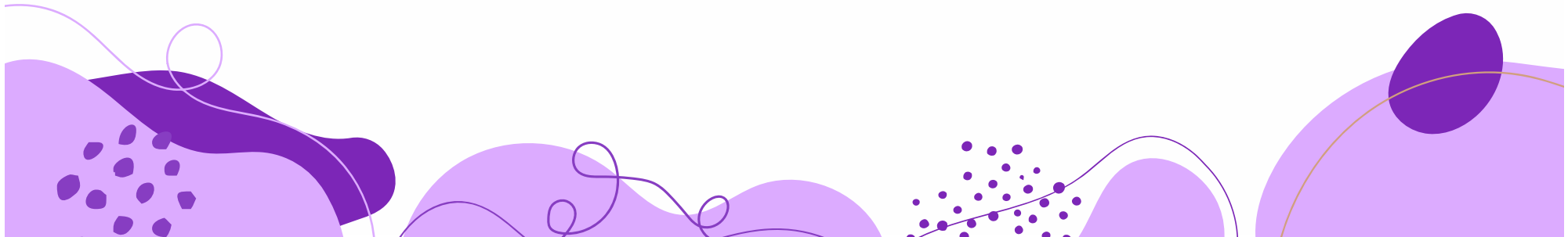
Checks the condition **before** running. It might not run at all if the condition is false initially.

```
while (condition) { ... }
```

Do...While Loop

Runs the code block **at least once**, then checks the condition at the end.

```
do { ... } while (condition);
```



Continue and Break

Break

Purpose
Immediately stops the loop entirely and jumps to the code after the loop.
When to use
When you've found what you're looking for or want to exit early.

Continue

Purpose
Skips the current iteration and moves to the next iteration of the loop.
When to use
When you want to skip certain cases but keep looping.

Use cases for each loop

For Loop

When you know **exactly how many times** you want to loop.

- Iterating over arrays or collections.
- Counting or repeating a task a fixed number of times.
- Repeating something with a specific start, end, and step.

While

When you **don't know in advance** how many times you need to loop.

- Keep looping **until a condition becomes false**.
- Usually used for **user input, reading files, or waiting for a condition**.

DO..While

When you **want the loop to run at least once**, regardless of the condition.

- Ensures **the code block executes before checking the condition**.
- Often used for menus or prompts that should appear at least once

Common loop errors

❌ Infinite loops
❌ Forgetting increment/decrement
❌ Wrong condition

Functions

The Toolbox

What is Function?

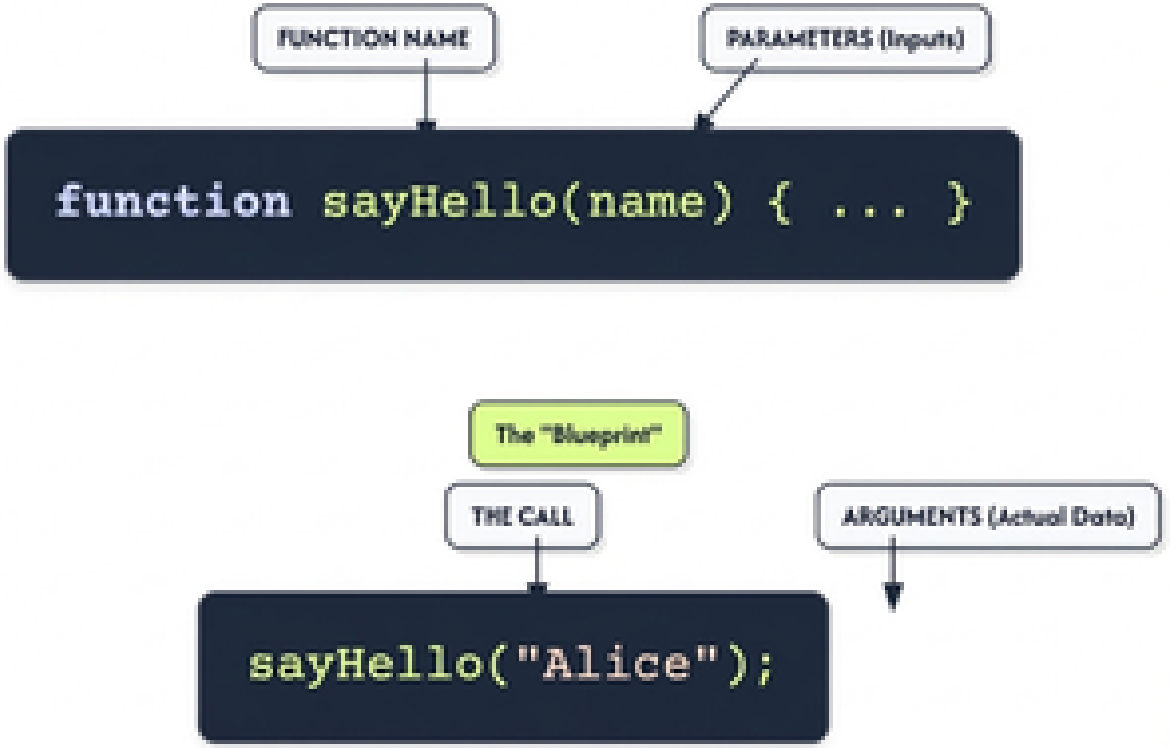
A function in JavaScript (and most programming languages) is basically a block of code that performs a specific task.

Why Function?

A function is a reusable block of code. If you find yourself copying and pasting logic, you should use a function.

Declaration: Define it once.
Call: Use it many times.

Anatomy of a Function



Return vs Console.log()

Return

Purpose: Sends a value back from the function so it can be used later.
Used for: Getting a result from a function and using it in your program.

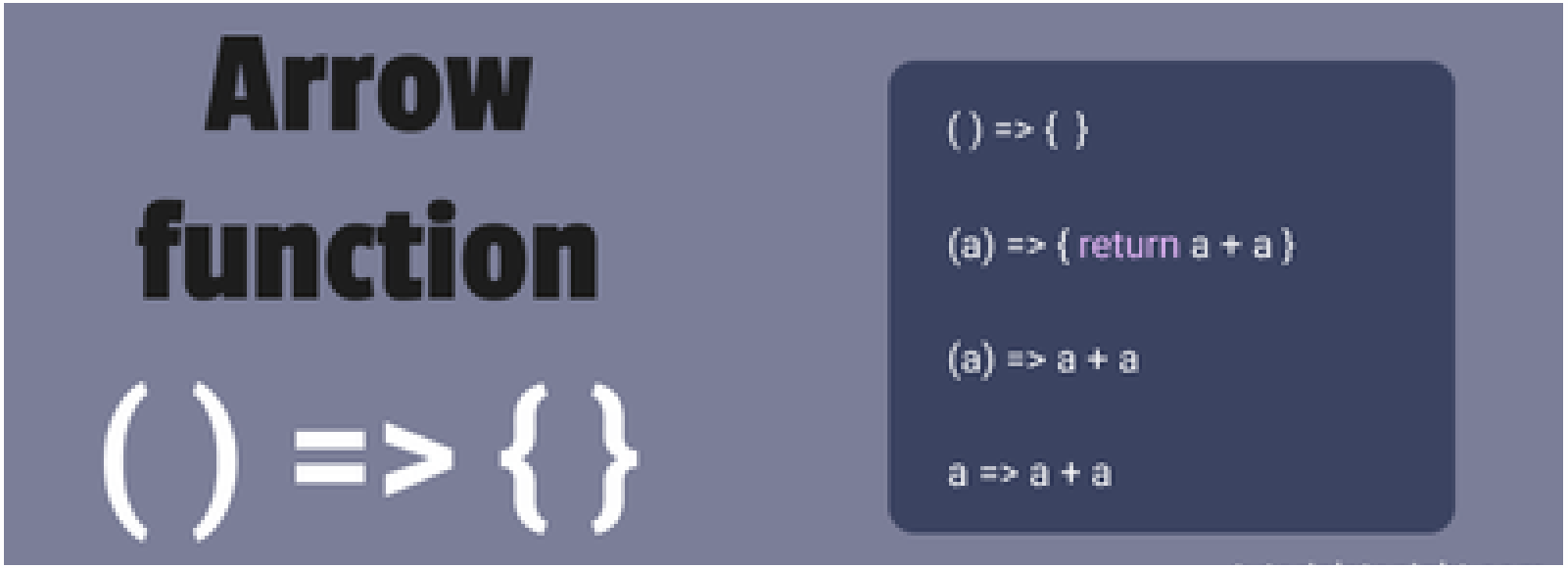
Stops the function immediately after returning.

Console

Purpose: Prints something to the console.
Used for: Checking values, debugging, or seeing output while running code.

It just shows it on the screen.

Arrow Functions



Block vs Function

Block Scope { }

Variables declared with let or const inside curly braces are "trapped" inside.

```
{
  let blockVar = "Hidden";
}
console.log(blockVar); // Error: Not defined
```

Block vs Function

THRIVE
Women for Women

Function Scope

Variables declared with var, let, or const inside a function stay there.

```
function test() {
  var funcVar = "I'm local";
}
console.log(funcVar); // Error: Not defined
```

Block vs Function

Feature	var	let	const
Scope	Function	Block	Block
Reassign?	Yes	Yes	No
Hoisting	Yes (undefined)	No (Error)	No (Error)

*Pro-Tip: Always use **const** by default. Switch to **let** only if you know the value will change.*

Code Reusability

The Hashtag Generator

```
const tag1 = "#" + "javascript";
const tag2 = "#" + "coding";
const tag3 = "#" + "webdev";
```

Instruction: Identify the repetitive work above and write a reusable function called makeTag to handle it.
Expected Result: #javascript

Exercises

Exercise 1

Write a function **checkAge(age)** that:

- Returns "Child" if age < 13
- Returns "Teenager" if age is between 13 and 19
- Returns "Adult" if age ≥ 20

Exercise 2

Write a function that prints all numbers from 1 to 10 using:

- A **for** loop
- A **while** loop

Exercise 3

Write a function **greetUser(name)** that returns:

"Hello, <name>! Welcome to Thrive Club."

Exercise 4

Write a function **getDayName(dayNumber)** using a switch statement:

- 1 → Monday
- 2 → Tuesday
- ...
- 7 → Sunday
- Any other number → "Invalid day"

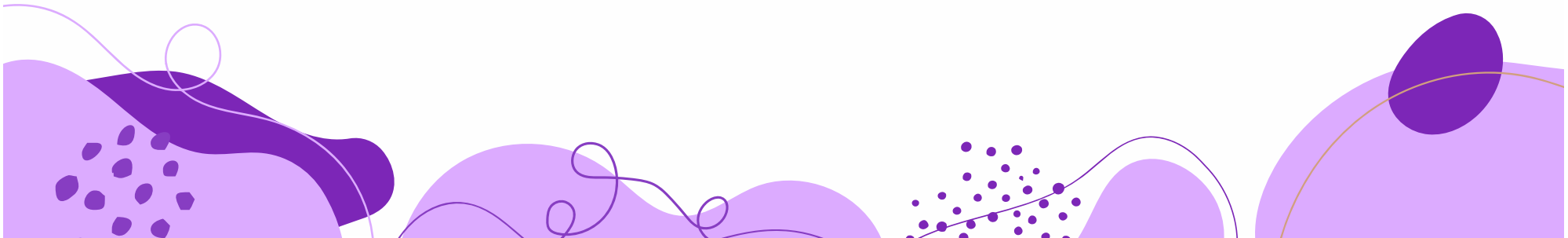
Home work

Calculator Functions
Create reusable functions:

- **add(a, b)**
- **subtract(a, b)**
- **multiply(a, b)**
- **divide(a, b)** (handle division by zero)

Home work

```
const students = [
  { name: "A", attempts: [45, 60,
78] },
  { name: "B", attempts: [30, 40] },
  { name: "C", attempts: [90] },
  { name: "D", attempts: [] }
]
```



Q/A