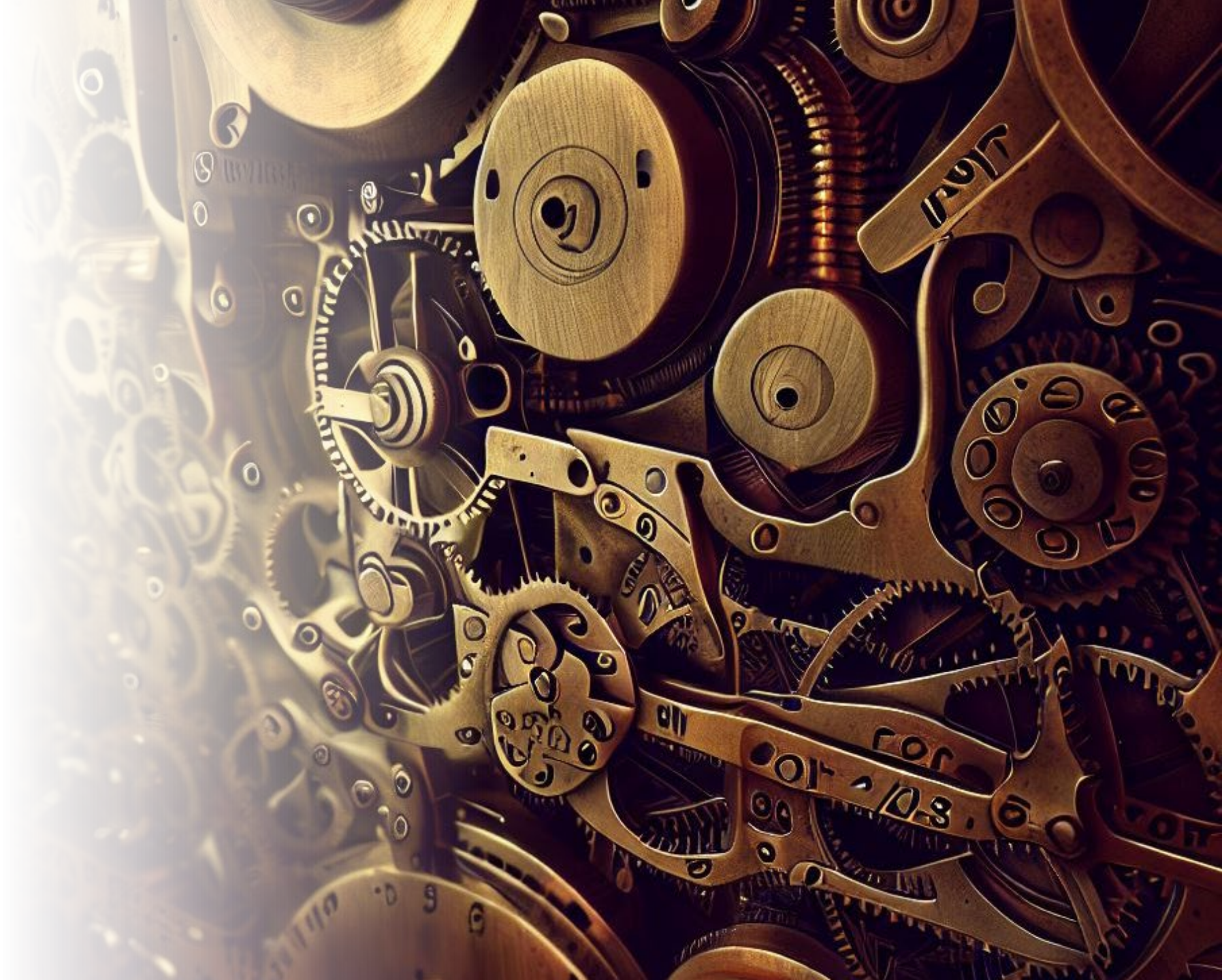
CTD Intro Week 2

javaScript Functions



```
// Anatomy of a function declaration
function aNewFunc(parm, anotherParm) { // parameter definitions
    // This is the function body, it has its own scope inside { }.
    // Computations are performed here in the body and the result is returned
    // A return statement can occur anywhere and multiple times.
    // Return exits the function and provides a value.
    // Return is not required, but if not specified, 'undefined' is returned.
    return parm + anotherParm;
}
```

What are functions?

- DRY Principal – Don't Repeat Yourself
- A way to encapsulate and reuse a piece of code
- Parameterized
  - Parameters define values which can be passed to the function
    - Function myFunc(parameter1, parameter2)
      - Parameters 1 and 2 can be referenced inside the function body { …. }
  - Arguments are the actual values used when a function is called
    - let result = myFunc(53, "myString");
- Returns a value using a 'return' statement

Defining Functions

- Named function declaractions
- Anonymous function declarations
- Shorthand function declaractions (arrow notation)
  - function(a, b) {a + b}
  - (a, b) => {a + b}
- Hoisting
  - Definitions are hoisted to the beginning of the scope
- Missing or extra parameters
  - Filled in with undefined unless defaulted
- Default values
  - function(a = true, b = 63) { … }
- Can use typeof() to check arguments
  - If (typeof(param1) != ‘string’) {throw…
- Throwing an error:
  - throw new Error(“an error message”);

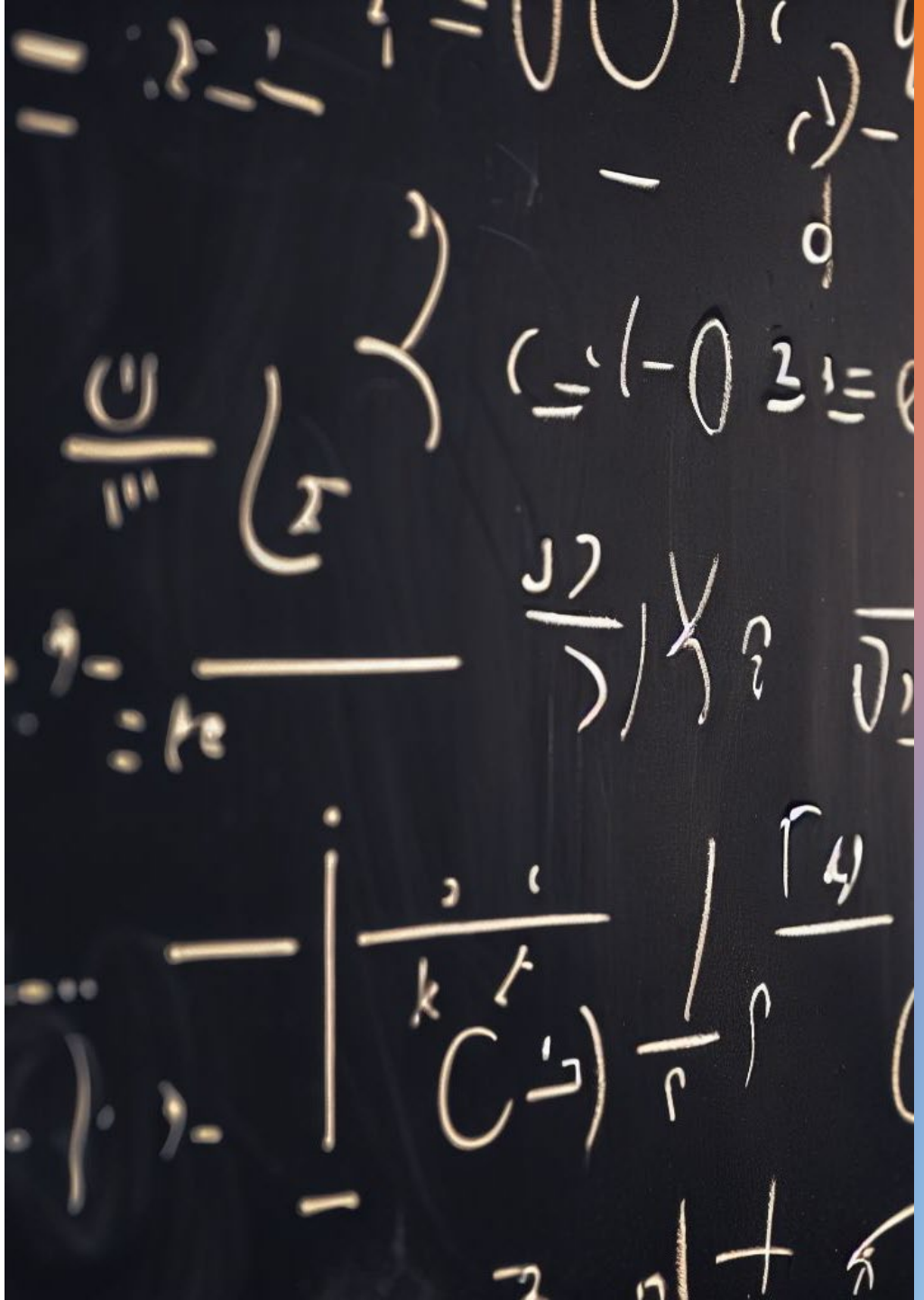
```
// anonymous function assigned to a lexically scoped variable
let funcInAVar = function(oneParam, twoParam) {
    return hoistedFunc(oneParam * twoParam);
}

// functions can be passed as parameters
// It is common in JavaScript
// Shorthand notation was invented to make it less verbose

// anonymous function calling in setTimeout
for (var i = 0; i < 10; i++) {
    setTimeout(function() {
      console.log(i);
    }, 1000 * i);
}

// shorter and simpler with => notation
for (var i = 0; i < 10; i++) {
    setTimeout(() => console.log(i) , 1000 * i);
}

// This named function can be called before it is defined.
// This is called hoisting.
function hoistedFunc(param) {
    return Math.random() * param;
}

// this function has defaulted parameters
function defParam(p1 = 53, p2 = "Tom") {
    return `${p2}'s favorite number is ${p1}`;
}
```

Dates and Times

- Unix time (epoch)
  - The number of seconds since 1/1/1970
  - Date.now() returns the number of milliseconds since 1/1/1970
- Date module
  - let today = new Date() creates a date object at the current date and time
  - today.getTime() unix epoch at the time ‘today’ was created
  - getFullYear(), getMonth(), getDate(), getDay()
  - getHours(), getMinutes(), getSeconds(), getMilliseconds()

The Math module

- Math.random()
  - Returns a floating point number from zero to 1
    - Zero is included, 1 is excluded (0.0 -> 0.9999999999)
    - Often scaled to create a random integer in a range
      - Math.floor(scaleInt * Math.random()) + 1 // integer in the range 1 -> scaleInt
- Math.floor(num) // next lower integer, remove fractional part
- Math.round(num) // rounds up or down depending on >= 0.5
- Math.abs(num) // absolute value
- Math.max(n1, n2, …) // maximum of the list of numbers



Questions