Welcome to lecture 3!

Agenda

Session Objectives

- Review homework
- Understand what functions are and how to define/use them
- Use conditional statements to make decisions in code
  - if
  - if...else
  - switch
- Use loops to repeat actions efficiently
  - while
    do...while
  - for
- Coding session
- Q&A
- Homework (due tomorrow)

Homework

Problem: Age Calculator

- Prompts the user for their birth year.
- Calculates the current age using the current year.
- Calculates the year the user will turn 100 (by adding 100 to the birth year).
- Displays both messages using alerts.

```
let birthYear = prompt("Enter your birth year:"));
const currentYear = 2025;
let age = currentYear - Number(birthYear);
let hundredYear = birthYear + 100;

alert("You are " + age + " years old.");
alert("You will turn 100 in the year " + hundredYear + ".");
```

- Important concept: Type casting used to convert a string (birthYear) to number

Type Casting

- What is type casting?
  - Converting data from one type to another
  - Implicit Type Casting: Javascript tries to perform operations between different data types
  - Explicit Type Casting: developer intentionally converts data from one type to another

```
console.log('5' + 3);              // Output: '53'; Recall '+' operator is used for concatenation & arithmetic
console.log('5' - 3);              // Output: 2

let str = "123";
let num = Number(str);             // num is 123

let num = 456;
let str = String(num);             // str is "456"

let value = 0;
let isTrue = Boolean(value);       // isTrue is false
```

Functions

Lift off: Expressions

- What is an expression?
  - A unit of code that evaluates to a value
  - Expressions simply perform a computation and returns a value
- Examples

```
5 + 3                   // arithmetic expression evaluates to 8
"Hello, " + "world!"    // string expression evaluates to "Hello, world!"
(2 + 3) * 4             // arithmetic expression evaluates to 20
```

Lift off: Statements

- What is a statement?
  - Complete instruction that performs an action
- Examples

```
let sum = 5 + 3; // creates a variable, sum, and assign it a value 8
/*
- the string (“The sum is: ”) is concatenated with an arithmetic expression
- arithmetic expression (5 + 3) produces the value 8
- so, the alert box displays the message "The sum is: 8"
*/
alert("The sum is: " + (5 + 3));
```

Functions

- What is a function?
  - A function is a reusable set of statements that performs a task or calculates a value
  - Functions can take inputs (parameters), execute a series of statements, and optionally return a value.
- Example:

```
/*
   function name: add
   parameters: x of data type number & y of data type number
   returns: a value of data type number
*/
function add(x, y) {
  return x + y;
}
```

Parameters vs. Arguments

- What are parameters?
  - Variables listed in a function’s definition
- What are arguments?
  - Actual values passed to the function when it is called
- Example

```
// 'x' and 'y' are parameters: they are placeholders defined in the function declaration
function add(x, y) {
  return x + y;
}
// 5 and 3 are arguments: they are the actual values passed to the “add” function
add(5, 3);
```

Return Type

- What are return types?
  - The type of value a function sends back to the caller after completing its execution
  - Return types: undefined, null, boolean, number, string. We'll learn more in the next lesson!
- Example:

```
// Function: isPositive
// Parameter(s): num of type number
// Returns a Boolean value
function isPositive(num) {
  return num > 0; // Evaluates to true or false based on the condition
}
console.log(isPositive(-5)); // Output: false
```

Examples

Return type:

```
/* Function name: ___; Parameters: ___; Returns: ___ */
function isEven(num) {
  return num % 2 === 0;
}
console.log("Is 4 even? " + isEven(4));   // Output: ?
```

return type:

```
/* Function name: ___; Parameters: ___; Returns: ___ */
function getNothing() {
  return null;
}
console.log("Nothing: " + getNothing()); // Output: ?
```

Functions: Problem

- Fill in the blanks. What's the output of the below code?

```
/*
        Function name: __
        Parameters: __
        Returns: __
*/
function createMultiplier(multiplier) {
  return function(num) {
    return num * multiplier;
  };
}
const multiplyBy3 = createMultiplier(3);
console.log(multiplyBy3(4)); // Output: ?
```

Conditional Statements

Conditional Statements

- What are conditional statements?
  - Allow your code to make decisions based on certain conditions
  - Conditionals checks can be achieved using if, if-else, and switch cases
- Example:

```
let score = 75;            // Declare a variable 'score', & assigns it a number value
// 'if' statement: checks a condition
if (score >= 60) {         // is score greater than or equal to 60?
  alert("You passed!");    // The alert statement executes only if the condition is true
}
// What's the output of this code?
```

Conditional Statements: if-else

- What’s an if-else condition?
  - The if-else statement provides an alternative block of code to execute if the condition is false
- Example:

```
let day = "Saturday";                                   // Declare a variable 'day' with a string value
// 'if-else' statement: checks the condition and decides which block to execute
if (day === "Saturday" || day === "Sunday") {
  console.log("It's the weekend!");                     // This statement runs if the condition is true
} else if(day === “Friday”) {
  console.log("Happy Friday!");
} else {
  console.log("I can’t wait for the weekend!");         // This statement runs if the condition is false
}
// What’s the output of this code?
```

Conditional Statements: switch

- What's a switch condition?
  - The switch statement allows you to choose from multiple possible blocks of code based on the value of an expression
- Example:

```
let fruit = "apple";              // Declare a variable 'fruit' and assign it a string value

switch (fruit) {                  // statement evaluates the expression (fruit) & matches it against multiple cases
  case "apple":                   // does the value of fruit equal "apple"
    alert("Apple pie!");          // alert() displays an alert box with the message
    break;                        // break statement exits the switch block to prevent executing other cases
  case "banana":                  // Case for when fruit is "banana"
    alert("Banana split!");
    break;
  default:                        // The default case is executed if none of the above cases match
    alert("Fruit salad!");
}
```

Loops

Loops: An Introduction

- What are loops?
  - Allow you to execute a block of code repeatedly as long as a specified condition remains true
  - Different ways to use loops: while, do-while and for
- Example:

```
let count = 0;                                  // Initialize a counter variable

// The 'while' loop: It checks the condition (count < 3) before each iteration
while (count < 3) {
  console.log("Count is " + count);             // This line is executed if count is less than 3

  count = count + 1;                            // Increment the counter to eventually break the loop
}

// What's the output?
```

Loops: do-while

- What's a do-while?
  - A do while loop will first execute a series of steps and then check a condition, define a variable and ability to mutate the variable
- Example:

```
let count = 0;                                  // Initialize a counter variable

// The 'do-while' loop executes the block at least once before checking the condition
do {
  console.log("Count is " + count);             // This code runs before the condition is checked
  count = count + 1;                            // Increment the counter
} while (count < 3);

// What's the output?
```

Loops: for

- What's for loop?
  - Series of statements that will be executed N times given a condition, defined variable and increment/decrement option
- Example:

```
// The 'for' loop: It initializes a variable, checks the condition, and increments the variable all in one line.
for (let i = 0; i < 3; i = i + 1) {
  // This block executes for each value of i from 0 to 2.
  console.log("i is " + i);
}

// What's the output?
```

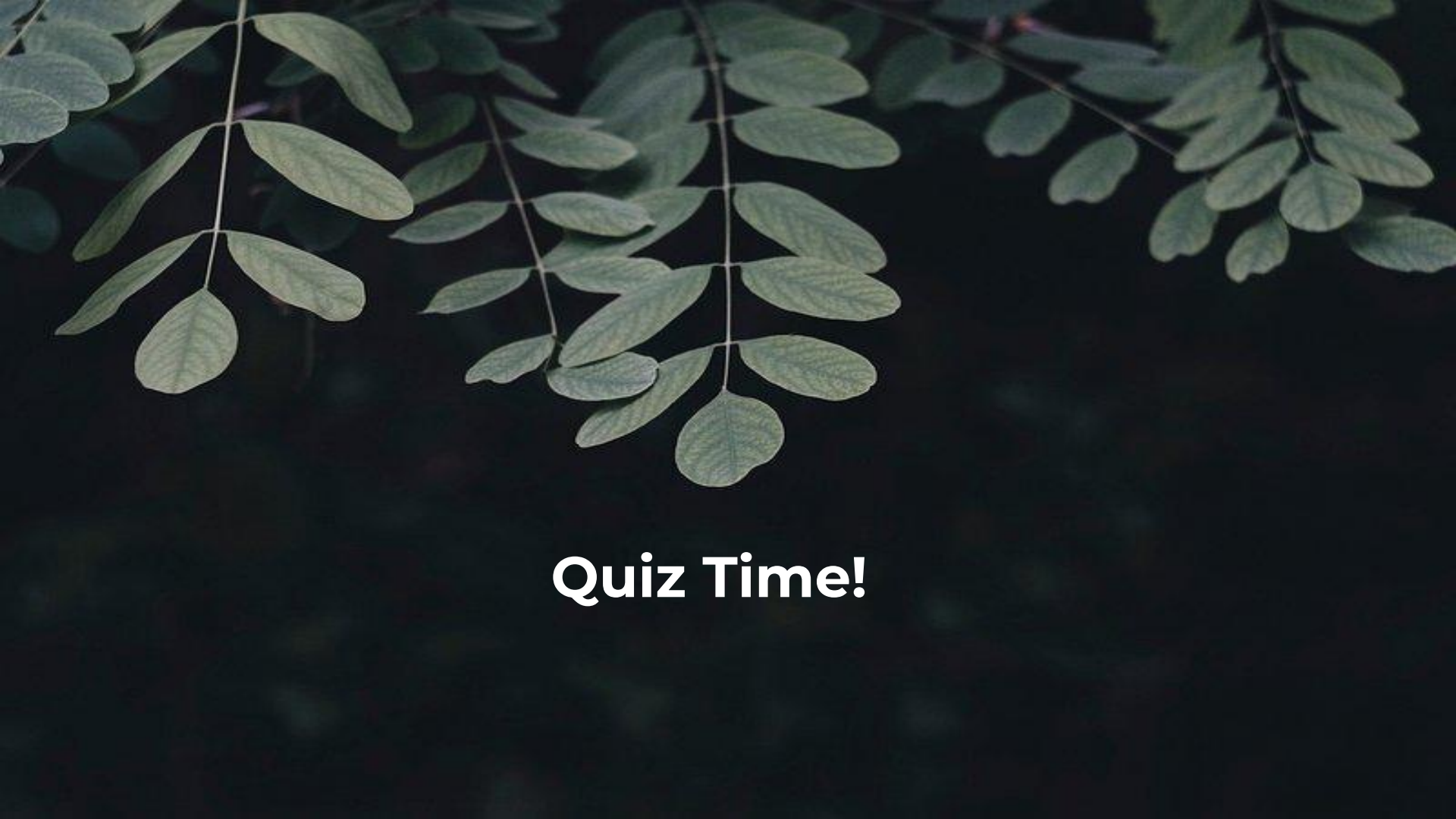
Loops with conditionals

Conditional loop with break:

```
let count = 1;                                  // Initialize counter
while (count <= 10) {                           // Loop until count is less than or equal to 10
  console.log("Count is " + count);
  if (count === 7) {                            // is count equal to 7?
    alert("Count reached 7, stopping loop!");   // Alert the user
    break;                                      // Exit the loop immediately
  }
  count = count + 1;                            // Increment the counter
}
```

Conditional loop with continue:

```
// Loop through numbers 1 to 10
for (let i = 1; i <= 10; i = i + 1) {
  // If the number is divisible by 3, skip the rest of the loop for this iteration
  if (i % 3 === 0) {
    continue;
  }
  console.log(i);                               // This line is skipped when i is a multiple of 3
}
```


Quiz Time!

1. Functions

- If a function does not use a return statement, what value is returned when you call it?
    - A) string
    - B) boolean
    - C) undefined
    - D) null

```
function noReturn() {
        /* Returns nothing */
}

console.log(noReturn()); // output: undefined
```

Correct Answer: C

2. Functions

- In a function definition, what are the variables in the parentheses called? And what are the values you provide when calling the function called?
    - A) Syntax; Semantics
    - B) Null; Undefined
    - C) Parameters; Arguments

```
/*
        Function name: subtract
        Parameters: a (data type: number), b (data type: number)
        Returns: a number
*/
function subtract(a,b) {
        return a - b;
}
console.log(subtract(3,5)); // output: ?
```

Correct Answer: C

3. Loops

- How many times will the following loop run?
  - A) 6
  - B) 5
  - C) 4
  - D) 3

```
for (let i = 0; i < 5; i++) {
        console.log(i);
}
```

Correct Answer: B

4. Switch statement

- What keyword do we use to prevent execution from falling through to the next case?
  - A) Break
  - B) Default
  - C) Switch
  - D) Function

```
let temperature = 45;
switch (temperature) {
  case temperature <= 20:
    console.log("Cold");
    break;
  case (temperature > 20 && temperature <= 30):
    console.log("Warm!");
    break;
  default:
    console.log("Hot");
}
```

Correct Answer: A

Recap

- Functions: reusable code blocks
  - takes an input, performs actions; may return a value
- Conditional statements: tools for decision-making
  - If, if/else or switch
- Loops: useful to repeat code blocks
  - while: evaluates a condition before repeatedly running some code
  - do-while: runs a code once, evaluates a condition and repeatedly runs the code
  - for: runs some code N number of times. You define N.

References

- Variable naming rules
- Reserved Keywords in Javascript
- Functions
  - Javascript Info: Functions & Function Expressions
  - Mozilla Documentation: Functions
- Conditionals
  - Javascript Info: if-else
  - Mozilla Documentation: switch, if-else
  - Eloquent Javascript: switch
- Loops
  - Javascript Info: loops
  - Mozilla Documentation: loops

Task: Shopping Cart Total Calculator

- Write a program that simulates a simple shopping cart calculator. Your program should:
  - Repeatedly prompt the user to enter the price of an item
  - If the user enters a negative number or a non-number value, display an alert saying "Invalid price, try again!" and skip adding that input
  - Continue prompting until the user clicks cancel
  - Calculate the total cost of the entered items
  - After the user finishes entering prices, use an if statement to check the total:
    - If the total is greater than 100, alert "Your total is Rs X. You qualify for a discount!"
    - Otherwise, alert "Your total is $X."
  - Write a function named calculateTotal that handles the price-collecting loop and returns the total

Deadline: Sunday, March 9th before 6PM. Send to nikhil.nair48@gmail.com