# Lecture 2: Control Flow, Loops

# What is Control Flow

- **Control Flow** refers to the order in which a computer executes statements in a script.
- By default, code is executed from top to bottom, unless specific instructions change the flow.

# Conditional Operators

**Conditional Operators** allow decision-making in code by executing different blocks on conditions.

They determine which path a program will take depending on the evaluation of expressions.

**If .... else Statements** **Ternary Operator** **switch Statement**

# If ... else Statement

- The **if** statement evaluates a condition and executes a block of code if the condition is true.
- The **else** clause allows the execution of code when the condition is false.

```
let temperature = 30;

if (temperature > 25) {
    console.log("it's a hot day!")
}
```

```
let temperature = 30;

if (temperature > 25) {
    console.log("it's a hot day!")
} else {
    console.log("it's a cold day")
}
```

# Else If

The **else if** statement allows you to test multiple conditions in sequence, executing a specific block of code if the associated condition is true, and moving on to the next condition if it is false.

```
let num = 10;

if (num > 10) {
    console.log('Greater than 10');
} else if (num === 10) {
    console.log("It's exactly 10")
} else {
    console.log('Lower than 10')
}
```

```
let num = 10;

if (num > 10) {
    console.log('Greater than 10');
} else if (num === 10) {
    console.log("It's exactly 10")
}
```

# Ternary Operator

The **ternary operator** is a concise, single-line shorthand for an **if...else** statement.

It provides a quick way to make simple conditional decisions in your code.

```
condition ? expressionIfTrue : expressionIfFalse;
```

# Ternary Operator (Examples)

```
let access = (age > 14) ? true : false;

if (access) {
    console.log('open');
} else {
    console.log('close');
}
```

```
let age = 18;

let message = (age < 3) ? 'wow' :
    (age < 18) ? 'Hi' :
    (age < 100) ? 'hello' :
    'Something new...';

console.log(message);
```

# switch Operator

The **switch** statement evaluates an expression and matches it against multiple possible cases.

It simplifies checking a value against multiple conditions, making code cleaner.

```
switch (dayNumber) {
    case 1:
        day = "Monday";
        break;
    case 2:
        day = "Tuesday";
        break;
    default:
        day = null;
}
```

# What are Loops?

**Loops** are a way to repeatedly execute a block of code as long as a certain condition is met.

They help automate repetitive tasks, making code more efficient and concise.

**Types of Loops in JavaScript:**

- **while** loop
- **do...while** loop
- **for** loop

# while loop

A **while** loop is a control flow statement that repeatedly executed a block of code **as long as a specified condition remains true.**

It checks the condition **before** executing the code, meaning the loop may not run at all if the condition is initially false.

```
let i = 1;

while (i < 3) {
    console.log(i);
    i++;
}
```

# do...while loop

```
let i = 1;

do {
    console.log(i);
    i++;
} while (i < 3);
```

- A **do...while** loop executed a block of code **at least once,** and then repeatedly executed the block as long as the specified condition is true.
- The main difference from a **while** loop is that the condition is checked **after** the code block has been executed, meaning the loop will always run **at least once.**

# for loop

- A **for** loop allows you to repeatedly execute a block of code **a specific number of times.**
- It is commonly used when you know in advance how many times the loop should run.

```
for (let i = 0; i < 3; i++) {
    console.log(i);
}
```

# Syntax of a for loop

- **Initialization –** This statement is executed once at the beginning of the loop. It typically sets up a counter variable.
- **Condition –** Before each iteration, this condition is checked. If it evaluates to **true,** the loop continues, if **false,** the loop stops.
- **Increment –** After each iteration of the loop, this statement is executed to update the counter variable (usually incremented)

```
for (initialization; condition; increment) {
    // Code to be executed
}
```