KINGSLAND
UNIVERSITY

Expressions and Statements

Variables, Data Types, Operators

Table of Contents

- **Variables**: Declaring, Reading, Modifying
- **Data Types**: Numbers, Text, Others
- **Statements** (Commands)
- Reading **User Input** and Formatting **Output**
- Arithmetic **Operators**: **+**, **-**, ***, **/**
- **Expressions**
- Practical **Exercises**

Expressions and Statements

Introduction

Real Life Example

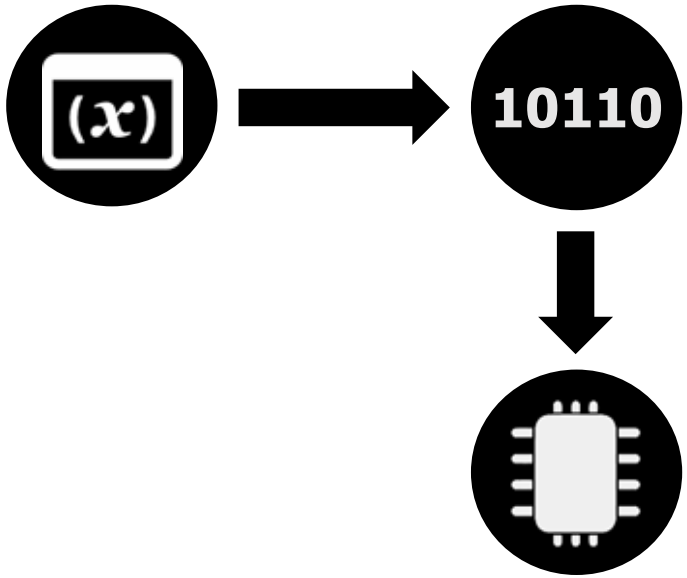
- Boxes holding different types of objects
- You have labels for each box and know what is in them
  - The **box** represents the **variable**
  - The **label** represents the **data type**

Variables

Storing Data

How Computing Works?

- Computers are machines that process data
  - **Programs** and **data** are stored in the computer memory
  - Data is stored by using **variables**

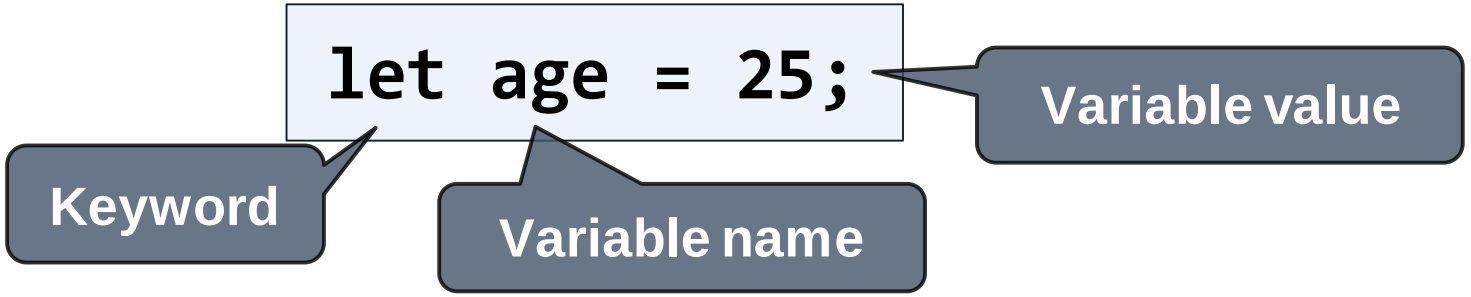


Variables

- **Variable** == named area of the computer memory
  - Stores a value from a particular data type
  - Is accessible in the program by its name
- Can be stored in the program's:
  - Operational memory (in the **stack**)
  - Dynamic memory (in the **heap**)
- Variables provide means for **storing**, **retrieving** and **modifying** data

Variables

- **Variable** == named area of the computer memory
  - Stores a **value** of a particular data type
  - Accessible in the program by its name
- Characterized by **name** (identifier) and **value** (stored information of certain type)
- Defining a variable in JavaScript:



Declaration Statements: let and var

- **`let`** – declares a variable, optionally initializing it

```
let age = 25;
console.log("Age:", age); // Age: 25
```

- **`var`** – similar to **`let`**, by gives a wider **scope**

```
function example() {
  console.log(age); // undefined
  var age = 25;
  console.log(age); // 25
}
example();
```

Use age and declare it later

Prefer let, unless you have a good reason to use var

Declaration Statements: const

- **`const`** – declares a read-only named **constant**

```
const name = "Peter";
console.log(name); // Peter
name = "John";     // TypeError
```

Data Types

The Ranges of Variables

Data Types

- Variables store value of a certain **type**
  - Number, letter, text (string), date, color, picture, list, …
- Simple **data types** in JavaScript:
  - **Number** – `2`, `3.14`, `-1`, `1.5e38`, …
  - **String** – `'hello'`, `'I like JS'`, `"another string"`, …
  - **Boolean** – `true` or `false`
  - **Null** – denotes a non-existing object
  - **Undefined** – variable does not exists

Data Types are Dynamic

- In JavaScript data types are **dynamic**
- A single variable can be used to hold values of different data types

```
let x = 5;    // x is Number
x = "John";   // x is String
x = true;     // x is Boolean
```

```
console.log(typeof(5));
console.log(typeof('abc'));
```

Statements

Commands in the Computer Programs

Statements

- The **actions** that a program takes, are expressed as **statements**
- Common actions include:
  - **Declaring** a variable

```
let counter;
```

  - **Assigning** a value

```
counter = 1;
```

  - **Declaring** + **initializing**

```
let counter = 1;
```

  - **Printing** a value (invoking a function)

```
console.log(counter);
```

  - **Modifying** a value

```
counter++;
```

```
sum = a + b;
```

Reading User Input and Formatting Output

Working with User Input

Console Output

- Everything we print on the console is converted to **`string`**
- Use the "**`console.log`**" function

```
console.log('Hello world!');
```

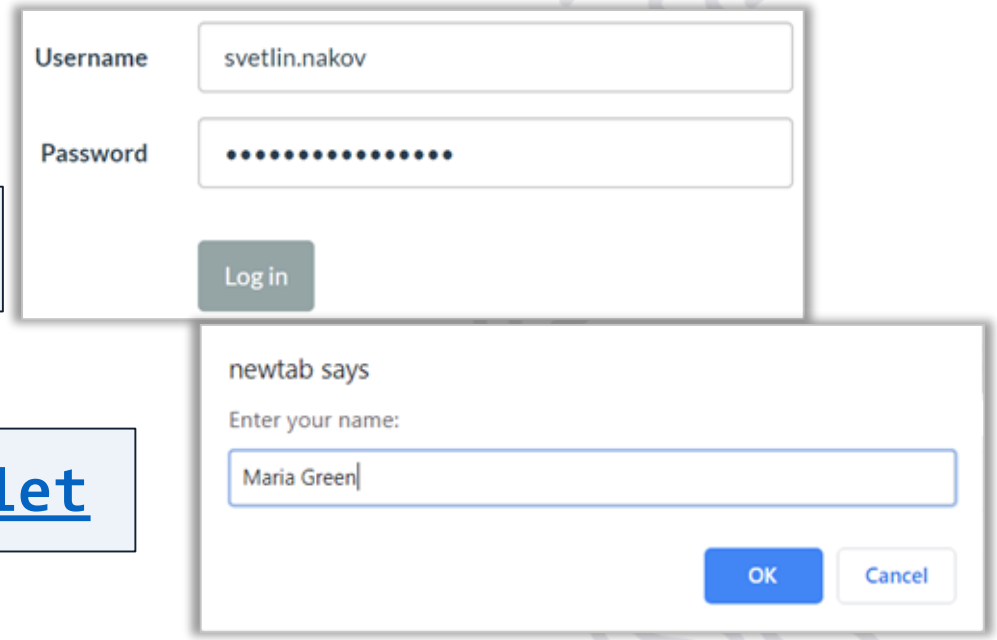
```
console.log(123);
```

```
console.log("Hello" + 123);
```

Reading User Input

- In software systems the user input comes from many sources
  - **UI controls**, e.g. text boxes
  - **Popup dialog** in the browser

```
let name = prompt("Enter your name:");
```



  - External **service**, e.g. REST API

```
https://api.github.com/users/myetherwallet
```

  - **Parameter** in a function call

```
function calculate(name) { … }
```

Functions and Parameters

- Defining a function:

```
function printAge(number) {
    console.log("Age: " + number);
}
```

Function name

Parameters

Body of the function

- Invoking a function:

```
printAge(5);  // Age: 5
printAge(10); // Age: 10
```

Passing Multiple Parameters

- You can pass multiple parameters to a function

```
function printSum(firstNum, secondNum) {
    console.log(firstNum + secondNum);
}

printSum(5, 10);          // 15
console.log(firstNum); // undefined
```

Both variables live in the scope of the function

Cannot be used outside the scope

Formatting Output

- Formatting output using **template strings**
  - Enclosed by the back-tick (` `) instead of quotes
  - Placeholders are indicated by the dollar sign **$** and curly braces: **${expression}**

```
let name = "Maria", town = "Paris";

console.log(`I am ${name} from ${town}.`);

// I am Maria from Paris.
```

Parsing Numbers

- Parsing a number:

```
let number = Number('2.5');
```

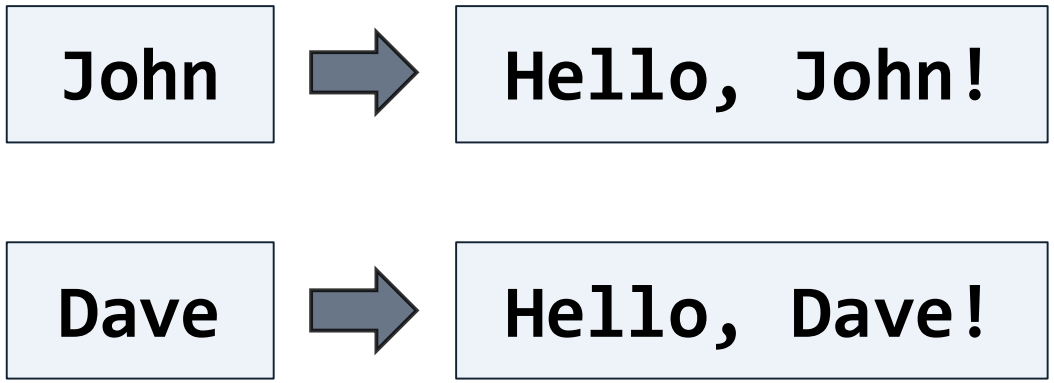
- Example: calculating square's area by given side

```
function calculateSquareArea(input) {
  let a = Number(input);
  let area = a * a;
  console.log(area.toFixed(2));
}
```

Format the number with 2 digits after the decimal point

Problem: Greeting

- Write a function that:
  - Receives a user input: **name**, from the console
  - Prints "**`Hello, {name}!`**" where **`{name}`** is the **user input**



Solution: Greeting

- First variant

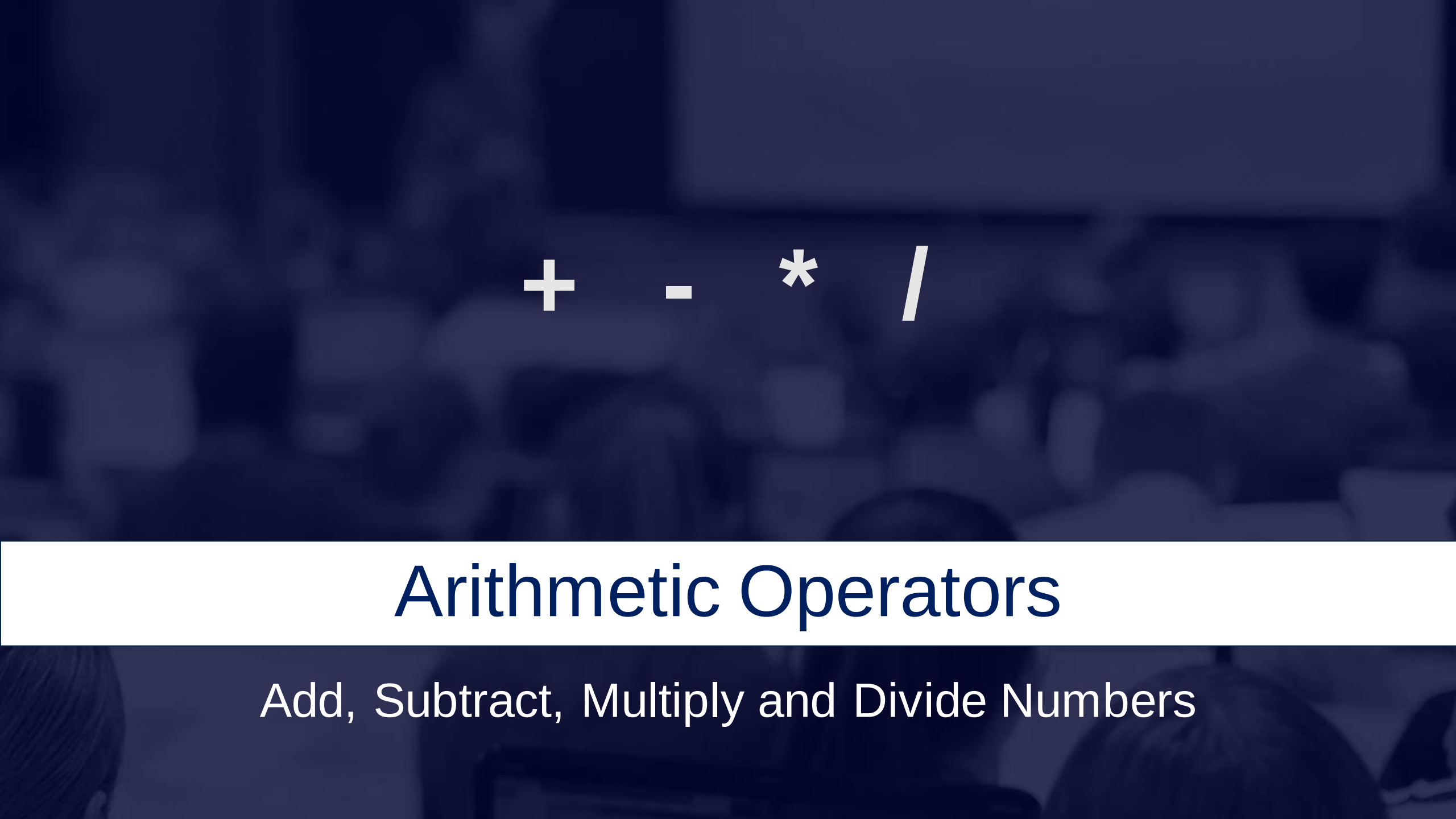
```
function sayHello(name) {
  console.log('Hello, ' + name + '!');
}
```

Concatenation

- Second variant

```
function sayHello(name) {
  console.log(`Hello, ${name}!`);
}
```

Template string



Arithmetic Operators

Add, Subtract, Multiply and Divide Numbers

Arithmetic Operators: + and -

- **Adding** numbers (operator **+**)

```
let a = 5;
let b = 7;
let sum = a + b;
console.log(sum); // 12
```

- **Subtracting** numbers (operator **-**)

```
let a = 15;
let b = 7;
console.log(a - b); // 8
```

Arithmetic Operators: * and /

- **Multiplying** numbers (operator *****)

```
let a = 5;
let b = 7;
console.log(a * b); // 35
```

- **Dividing** numbers (operator **/**)

```
let a = 25;
console.log(a / 4); // 6.25
console.log(a / 0)  // Infinity
console.log(0 / 0)  // NaN
```

Arithmetic Operators: %

- Modulo / **remainder** from division (operator **%**)

```
let a = 7;
let b = 2;
console.log(a % b);   // 1
```

7 = 3 * 2 + 1

```
console.log(3 % 2);   // 1
console.log(4 % 2);   // 0
console.log(3.5 % 1); // 0.5
```

3 = 1 * 2 + 1

4 = 2 * 2 + 0

3.5 = 3 * 1 + 0.5

Expressions

Combining Values with Operators

Expressions

- **Expressions** == sequences of operators, literals and variables which are evaluated to a value
  - Consist of at least one **operand**
  - Can have 1 or more **operators**

```
let y = x + 5;
```

```
let name = "John Doe";
```

```
let r = (150-20) / 2 + 5;
```

Summary

- Variables hold data
- Data types define data ranges
- Statements define commands
- Reading and printing user input
- Simple operations
  - Arithmetic operators: +, -, *, /, %
  - Formatting output: `${expr}`
- Expressions == operators + values

Questions?

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © Kingsland University – https://kingslanduniversity.com

KINGSLAND UNIVERSITY

THANK YOU