SD400: Problem Solving

Lesson 2: Basic Programming and Data Types

Wholeness

- We are going to study about languages that enable humans to communicate with computers. Therefore, we tell a computer how to do things, take input, and to create an output from it.
- A computer program is a tool that allows us to do less and accomplish more, however, we must be able to instruct it. Similarly, TM is a tool that allows us to to reach to a higher level of restfulness and alertness.

Lesson Objectives

- What is a Programming Language
- Introduction to JavaScript
- Data Types
- Writing basic programs
- Converting flowcharts to code
- Input / Output in JavaScript

What is a Programming Language

A programming language is a formalized system of communication used by computers to instruct them to perform specific tasks or operations. It serves as a bridge between human understanding and the binary language (0s and 1s) that computers use to execute instructions.

Key characteristics of programming languages

- **Human-Readable**: Programming languages are designed to be human-readable and writable. They use words, symbols, and syntax that are easier for humans to understand and work with.
- **Instruction Set**: A programming language provides a set of instructions and commands that can be used to describe algorithms and processes. These instructions can range from simple arithmetic operations to complex data manipulation.
- **Abstraction**: Programming languages allow developers to work at different levels of abstraction. They can write high-level code that abstracts away low-level details, making it easier to focus on solving problems.

Key characteristics of programming languages

- **Execution**: Once a program is written in a programming language, it needs to be translated into machine code (binary) for the computer to execute. This translation is typically performed by a compiler or interpreter, depending on the language.
- **Variables and Data Types**: Programming languages provide mechanisms for declaring and using variables to store and manipulate data. They also define various data types, such as integers, strings, and arrays, to represent different kinds of information.
- **Control Flow**: Programming languages offer constructs for controlling the flow of a program, including conditional statements (if-else), loops (for, while), and branching.

Key characteristics of programming languages

- **Functions/Methods**: Languages allow the creation of functions or methods, which are reusable blocks of code that can be called with different inputs. This promotes modularity and code reusability.
- **Libraries and Frameworks**: Many programming languages have extensive libraries and frameworks that provide pre-written code and functionalities to simplify common tasks.
- **Portability**: Programs written in a programming language can often be executed on different computer platforms with minimal or no modification, as long as there's a compatible interpreter or compiler for that platform.

Examples of

Examples of popular programming languages include

- Python
- JavaScript
- Java
- C++
- C#
- Ruby
- PHP
- and many more.

Each language has its own syntax, strengths, and weaknesses, making it suitable for different types of tasks and applications. Programmers choose the appropriate language based on the requirements and goals of a project.

Introduction to JavaScript

- *JavaScript* was initially created to “make web pages alive”.
- The programs in this language are called *scripts*. They can be written right in a web page’s HTML and run automatically as the page loads.
- Scripts are provided and executed as plain text. They don’t need special preparation or compilation to run.
- In this aspect, JavaScript is very different from another language called Java .

Why is it called JavaScript?

- When JavaScript was created, it initially had another name: “LiveScript”. But Java was very popular at that time, so it was decided that positioning a new language as a “younger brother” of Java would help.
- But as it evolved, JavaScript became a fully independent language with its own specification called ECMAScript, and now it has no relation to Java at all.

JavaScript can also work on server-side

- Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine.
- The browser has an embedded engine sometimes called a “JavaScript virtual machine”.
- Different engines have different “codenames”. For example:
  - V8 – in Chrome, Opera and Edge.
  - SpiderMonkey – in Firefox.
  - …There are other codenames like “Chakra” for IE, “JavaScriptCore”, “Nitro” and “SquirrelFish” for Safari, etc.

Data Types

- JavaScript allows us to work with primitives (strings, numbers, etc.) as if they were objects. They also provide methods to call as such. We will study those soon, but first we'll see how it works because, of course, primitives are not objects (and here we will make it even clearer).

Primitive vs Object

- A primitive
  - Is a value of a primitive type.
  - There are 7 primitive types: string, number, bigint, boolean, symbol, null and undefined.
- An object
  - Is capable of storing multiple values as properties.
  - Can be created with {}, for instance: {name: "John", age: 30}. There are other kinds of objects in JavaScript: functions, for example, are objects.

Example of primitive types

String ➔ `let greeting = "Hello, World!";`

Number ➔ `let age = 30;`

BigInt ➔ `let bigNumber = 1234567890123456789012345678901234567890n;`

Boolean ➔ `let isJavaScriptFun = true;`

Symbol ➔ `const uniqueSymbol = Symbol("description");`

null ➔ `let emptyValue = null;`

undefined ➔ `let notDefined;`

Note!

- It's important to note that **null** is distinct from **undefined**. undefined is a primitive value that represents a variable that has been declared but hasn't been assigned any value yet. null is typically used when you want to explicitly indicate that a variable should have no value or that an object property should be empty.

Example of objects

```
let student = { id : 111 , name: "Dean"}
```

- You store a function as one of its properties.

```
let john = { name: "John",
sayHi: function() { console.log("Hi buddy!"); }
};
```

→ `john.sayHi(); // Hi buddy!`

Details of Primitives

- Here's the paradox faced by the creator of JavaScript:
  - There are many things one would want to do with a primitive, like a string or a number. It would be great to access them using methods.
  - Primitives must be as fast and lightweight as possible.
- The solution looks a little bit awkward, but here it is:
  1. Primitives are still primitive. A single value, as desired.
  2. The language allows access to methods and properties of strings, numbers, booleans and symbols.
  3. In order for that to work, a special "object wrapper" that provides the extra functionality is created, and then is destroyed.
- The "object wrappers" are different for each primitive type and are called: String, Number, Boolean, Symbol and BigInt. Thus, they provide different sets of methods.
- **We will dive in deeper into these topics in Week 3**

Writing basic programs with JavaScript

- In order to focus on problem solving, we will be using an online editor https://playcode.io/javascript
- Calculate the average of three numbers

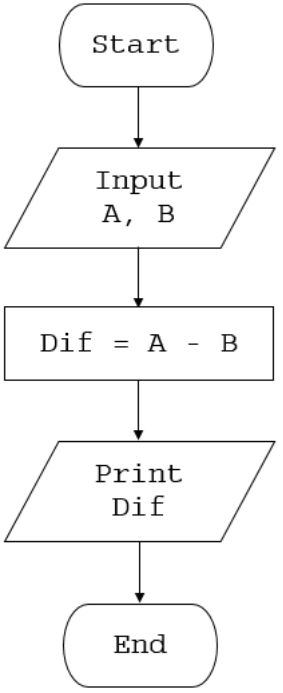
```
let x = 3;
let y = 4;
let z = 5;

let sum = x + y + z;
let avg = sum / 3;

console.log(avg);
```

Convert Flowchart to Code

PRINT DIFFERENCE OF 2 NUMBERS



Adding Input/Output

There are multiple ways of getting input, in this course we will be using the prompt() function since we are using the browser.

```
let pName = prompt("Please enter your name: ");

const greetingMessage = "Hello " + pName;

console.log(greetingMessage);
```

Exercise

- Write a program that ask the user to enter a value in lb and convert it to kg and print out the result.

Main Point

- Input and Output are how the computer interacts with humans.

- *Science of creative intelligence*: In general every action has an equal and opposite reaction. Our goal is to create the optimal reaction based on the users action.