Information Technology Department - State Polytechnic of Malang
Jobsheet-03: Javascript (Data types, operators, and function)
Course: Web Programming / Web Design and Programming
Instructor: Web Design and Programming Teaching Team
September 2024

Rizqi Dewanto / 17 / SIB – 2G

Topic

1. Introduction to Basic Concepts of JavaScript
2. Data Types, Operators, and Functions in JavaScript
3. JavaScript in HTML

Objectives

Students are expected to:

1. Understanding the concept of Javascript
2. Understanding Data types, operators and functions in javascript
3. Students are able to run javascript in HTML files

Introduction

JavaScript is a client-side programming language used for web development. A client-side programming language means that the processing is done on the client's side. The client application in this case refers to web browsers such as Google Chrome and Mozilla Firefox. Client-side programming languages differ from server-side programming languages like PHP, where all the program code is executed on the server side.
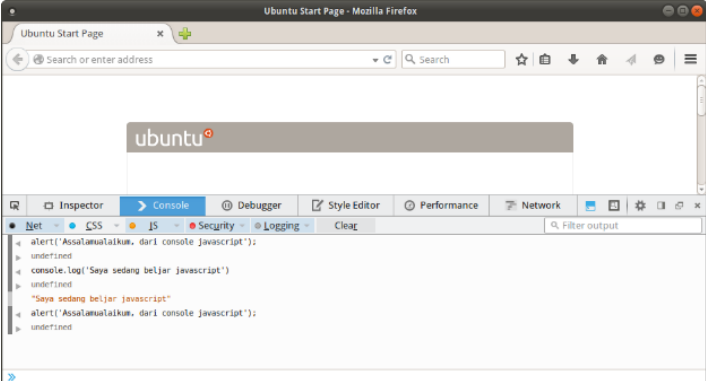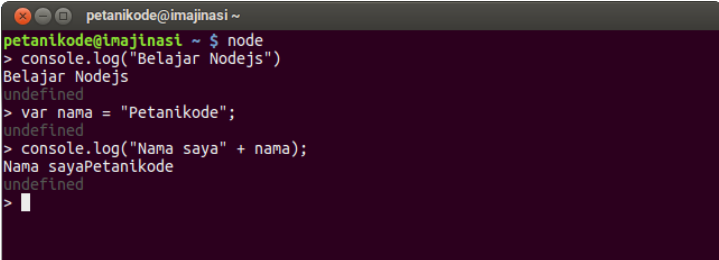
To run JavaScript, you only need a text editor and a web browser. JavaScript offers features such as being a high-level programming language, client-side, loosely typed, and object-oriented. Initially, JavaScript was developed to make interactions between users and websites faster without having to wait for processing on the web server. Before JavaScript, every interaction from the user had to be processed by the web server.

Imagine when you fill out a registration form on a website, click the submit button, wait about 10 seconds for the website to process the form, and then receive a page stating that some form fields were left unfilled. This is the kind of situation for which JavaScript was developed. The processing to check whether all form fields have been filled can be transferred from the web server to the web browser.

As JavaScript evolved, it became useful not only for form validation but also for many modern purposes. Various animations to beautify web pages, chat features, modern effects, games—all of these can be created using JavaScript. There are 3 main ways to write JavaScript tags:
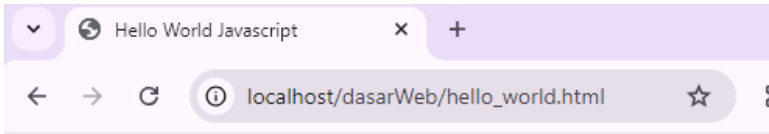
1. Writing the tag with **`<script type="text/javascript">`** at the start and ending with **`</script>`**. The attribute informs the browser that the script within the tag is JavaScript in text format.

2. Writing the tag with **`<script language="javascript">`** at the start and ending with **`</script>`**. This attribute is used to specify the version of JavaScript being used. For example, **`<script language="javascript1.2">`** indicates that the version of JavaScript used is 1.2.

3. Writing the tag with **`<script language="javascript" type="text/javascript">`** at the start and ending with **`</script>`**. This mixed method combines the old and new ways of writing, allowing compatibility for web browsers that support JavaScript but may not yet support HTML fully.
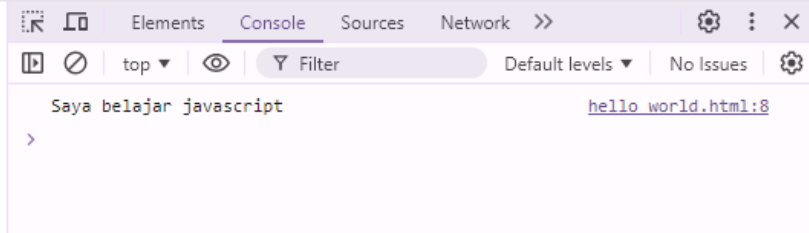
Practical Section 1: Learning Javascript

Step	Description
1	We can open the JavaScript console through **`Inspect Element -> Console`**. In the console, we can write functions or JavaScript code, and the results will be displayed immediately.

2	For example, let's try the following code:
	``js
	console.log("Hi apa kabar!");
	alert("Saya sedang belajar javascript");
	``
	Observe what appears on the console, then record your observations!.

3	If you are using Nodejs, then the way to access *the* **`console`** is to type the node's command in the Terminal.

4	Observe what happens, then record your observations. What can be concluded after trying *the Javascript* console?
	(Question No.1)
	Using the JavaScript console, you can see how different commands interact with the browser. The console.log method helps in debugging by printing messages to the console, while the alert method provides a way to interact with the user through pop-up messages.

Practical Section 2: Creating the First Javascript Program

Step	Description
1	Please open a text editor, then create a new file named `hello_world.html`
2	Type the program code below:
| | ```html
<!DOCTYPE html>
<html>
<head>
    <title>Hello World Javascript</title>
</head>
<body>
    <script>
        console.log("Saya belajar Javascript");
        document.write("Hello World!");
    </script>
</body>
</html>
``` |
3	Save it as `hello_world.html`, then open the file with a web browser.
4	Observe what happens in the browser, then record your observations
	(Question No.2)
	Hello World Javascript ✕ +
	← → C ⓘ localhost/dasarWeb/hello_world.html ☆
	Hello world!
	The console will show the message "Saya belajar javascript". The web page will display the text "Hello world!".

5	Now try to open the javascript console, **right click** page in the browser, then choose `Inspect Elements > Console`
6	Observe what happens in the Console tab, then record your results! (Question No. 3) ==The code demonstrates basic JavaScript functionality for logging messages to the console and writing content to the web page.==
7	Earlier, we wrote the command: ```console.log("Saya belajar Javascript");``` Why do you think the command is not displayed? (Question No.4) ==The console.log("Saya belajar javascript"); command is not displayed on the web page because it is designed to log messages to the browser's console, not to the web page itself.==

Practical Section 3: How to Write Javascript Code in HTML

In practicum 2 we have written javascript code in HTML, this method is an embeded writing method. Some other ways that we need to know include:

1. **Embed** (Javascript code pasted directly into HTML)
2. **Inline** (Javascript code written on HTML attributes)
3. **External** (Javascript code is written separately from the HTML file)

1. Writing Javascript Code with Embed	
Step	**Description**
1	In this way, we use the `<script>` tag to embed *the* Javascript code in the HTML. These tags can be written in the `<head>` and `<body>` tags
2	Type the program code below:

```
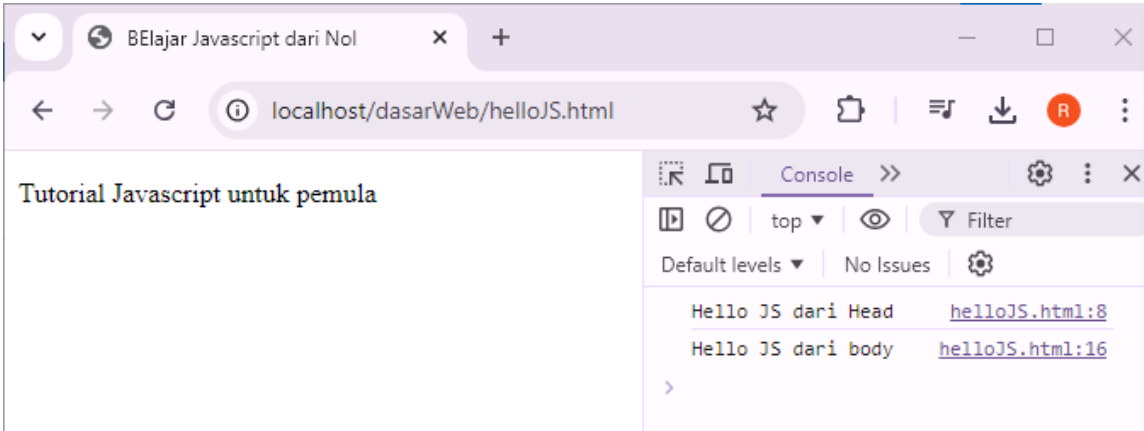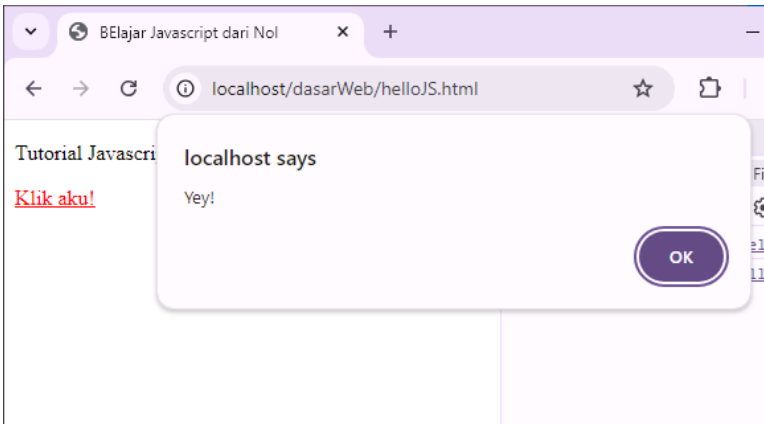<!DOCTYPE html>
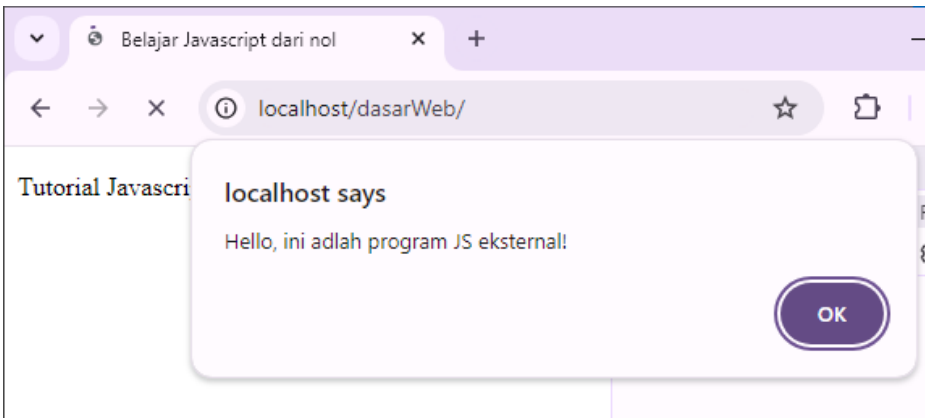<html>
    <head>
        <title>Belajar Javascript dari Nol</title>
        <script>
            // ini adalah penulisan kode javascript
            // di dalam tag <head>
            console.log("Hello JS dari Head");
        </script>
    </head>
    <body>
        <p>Tutorial Javascript untuk Pemula</p>
        <script>
            // ini adalah penulisan kode javascript
            // di dalam tag <body>
            console.log("Hello JS dari body");
        </script>
    </body>
</html>
```

Step	Description
3	Observe what happens to the browser? Record your observations
	(Question No. 5)
	The code demonstrates how JavaScript can be included in both the \<head\> and \<body\> sections of an HTML document. Messages logged with console.log appear in the console, not on the web page itself.
4	Which do you think is better, written in the **\<head\>** or **\<body\>** tag?
	(Question No. 6)
	Usually, writing JavaScript code at the end \<body\> is better because it ensures all the elements of the page are ready to be accessed.

2. Inline Javascript Code Writing
Step	**Description**

1	In this way, we'll write the javascript code inside the HTML attribute. This method is usually used to call a function on a specific event. One example is when clicked.

2	Type the program code below:
	
	 ``html
	Klik aku!
	``
	
	 Or it can also be like this:
	
	 ``html
	Klik aku!
	``

3	Observe what happens to the browser! Record your observations (Question No. 7) will bring up a button here where when typed will appear writing yey
4	What is the difference between the two program codes (Question No. 8) only difference in syntax

3. External Javascript Code Writing	
Step	**Description**
1	In this way, we'll write the javascript code separately from the HTML file. In this method, we will write JavaScript code separately from the HTML file. This approach is usually recommended for large projects, as it makes project code management easier.
2	Let's try, create two files, namely HTML and Javascript files. ``` belajar-js/ ├── kode-program.js └── index.html ```

3	Contents of the **kode-program.js** file :
	
	 ``javascript
	alert("Hello, ini adalah program JS eksternal!");
	``

4	Contents of the `index.html` file :

	``html
	<!DOCTYPE html>
	<html>
	    <head>
	        <title>Belajar Javascript dari Nol</title>
	    </head>
	    <body>
	         Tutorial Javascript untuk Pemula
	
	        <!-- Menyisipkan kode js eksternal -->
	        <script src="kode-program.js"></script>
	    </body>
	</html>
	``

5	Observe what happens to the browser! Record your observations (Question No. 9) The text "Tutorial Javascript untuk pemula" is displayed on the web page. Any JavaScript code in kode-program.js will be executed. For example, if kode-program.js contains console.log("Hello dari file eksternal!");, you will see this message in the browser's console.

6	In the experiment, we wrote separate javascript code with HTML code.
	Then in the HTML code we insert the `src` attribute in the `<script>` tag
	
	``html
	<!-- Menyisipkan kode js eksternal -->
	<script src="kode-program.js"></script>
	``
	
	Then anything in `kode-program.js` file will be readable from `index.html` file

7	What would happen if the javascript file was in a different folder? Observe and record your observations (Question No. 10) - Correct Path: The script runs as expected. - Incorrect Path: The script fails to load, resulting in errors.
8	Suppose we have a folder structure like this:

```
belajar-js/
├── js/
│     └── kode-program.js
└── index.html
```

So to insert the **kode-program.js** file into the HTML, we can write the following code:

```
<script src="js/kode-program.js"></script>
```

Because the **kode-program.js** file is in the **js** directory.

We can also insert javascript that exists on the internet by providing the full URL address.

Example:

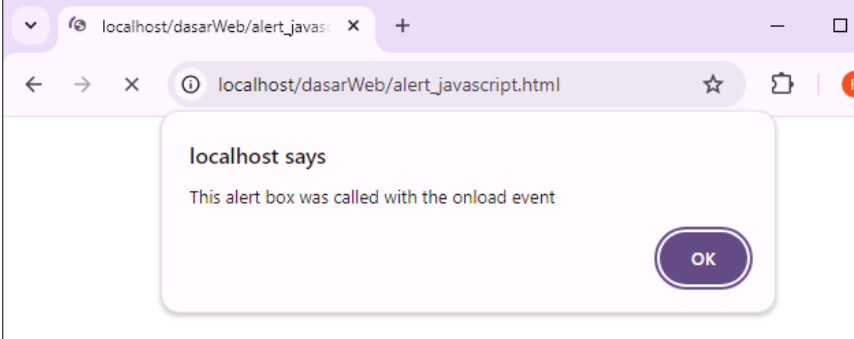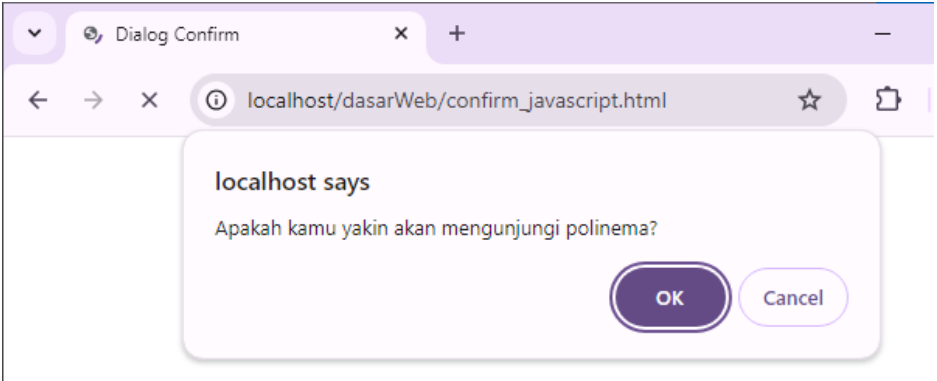```
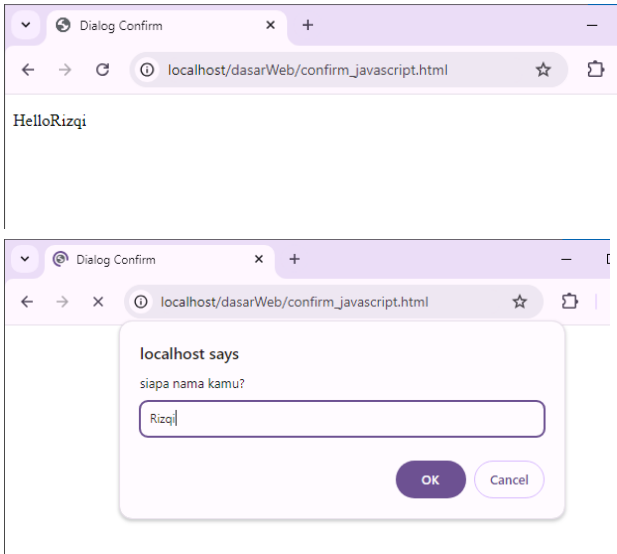<script src="https://www.petanikode.com/js/kode.js"></script>
```

Practical Section 4: Dialogue Window

A dialog window is a window used to interact with users. There are three types of dialog windows in Javascript:

1. The **alert()** dialog window;
2. The **confirm()** dialog window;
3. The prompt dialog **window();**

Step	Description
1	Create a new file **alert_javascript.html** and save it in the project folder.
2	Type the program code below
| | ```
<html>
<head>
<script type="text/javascript">
function message()
{
alert("This alert box was called with
the onload event")
}
</script>
</head>
<body onload="message()">
</body>
</html>
``` |
| 3 | Observe what appears on the browser |
| 4 | Record your observations
(Question No. 11) |

An alert box appears immediately after the page loads, showing the specified message. The alert box requires user interaction (clicking "OK") to proceed.

5	Create a new file named `confirm_javascript.html` and save it in the project folder
6	Type the program code below

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Dialog Confirm</title>
    </head>
    <body>
    <script>
        var yakin = confirm("Apakah kamu yakin akan mengunjungi polinema?");

        if (yakin) {
            window.location = "https://www.polinema.ac.id";
        } else {
            document.write("Baiklah, tetap di sini saja ya :)");
        }
    </script>
    </body>
</html>
```

7	Observe what appears on the browser
8	Record your observations

(Question No. 12)

Clicking "OK" redirects to the specified URL.
Clicking "Cancel" should display the message on the page, but due to the typo, it will not work as expected.

9	Type the program code below

| | ```
<!DOCTYPE html>
<html>
    <head>
        <title>Dialog Promp</title>
    </head>
    <body>
        <script>
            var nama = prompt("Siapa nama kamu?", "");
            document.write("<p>Hello "+ nama +"</p>");
        </script>
    </body>
</html>
``` | |
| 10 | Observe what appears on the browser | |
| 11 | Record your observations (Question No. 13) | |

A prompt dialog box appears asking for the user's name.
After entering the name and clicking "OK", the web page displays a personalized greeting, such as "Hello [name]".

Practical Section 5: Variables

The way to create a variable that is commonly used in javascript is to use the var keyword followed by the name of the variable and its value.

Example: `var title = "Learn Javascript Programming";`

Displaying the contents of a Variable

To display the contents of the variables, we can utilize functions to display outputs such as:

- The `console.log()` function returns the output to the javascript console;

- The `document.write()` function returns the output to an HTML document;
- and the `alert()` function returns the output to the dialog window.

Step	Description
1	Create a new file `variable_javascript.html` and save it in the project folder.
2	Type the program code below

	``html
	<!DOCTYPE html>
	<html lang="en">
	<head>
	    <title>Belajar Variabel dalam Javascript</title>
	    <script>
	        // membuat variabel
	        var name = "Javascript";
	        var visitorCount = 50322;
	        var isActive = true;
	
	        // menampilkan variabel ke jendela dialog (alert)
	        alert("Selamat datang di " + name);
	
	        // menampilkan variabel ke dalam HTML
	        document.write("Nama Situs: " + name + "
	");
	        document.write("Jumlah Pengunjung: " + visitorCount + "
	");
	        document.write("Status Aktif: " + isActive + "
	");
	
	    </script>
	</head>
	<body>
	</body>
	</html>
	``

3	Observe what appears on the browser
4	Record your observations (Question No. 14) An alert box welcomes the user with the message "Selamat datang di JavaScript". The web page displays the site name, visitor count, and active status.

Deleting Variables

In JavaScript, deleting variables is uncommon. However, in programs where careful memory management is crucial, removing variables is important to ensure more efficient memory usage. This can be achieved using the **delete** keyword.

Example:

```
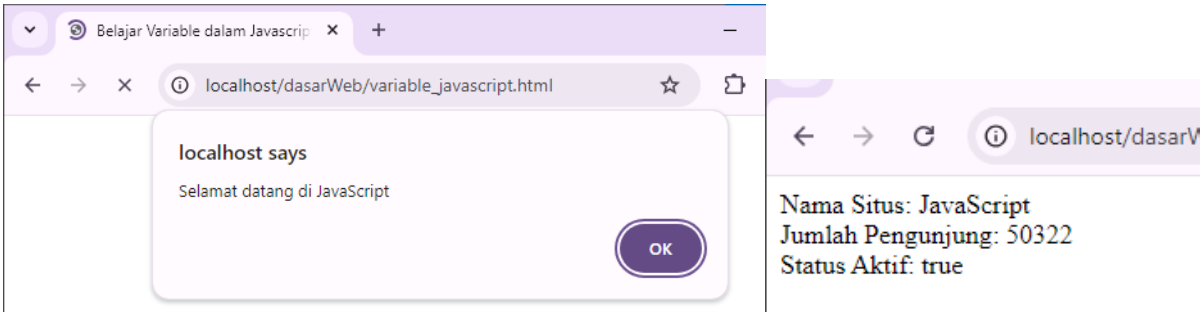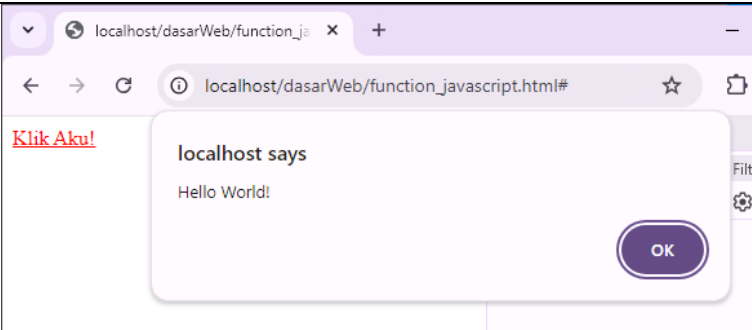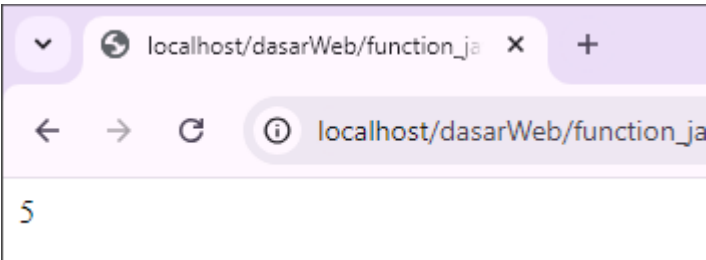var bookTitle = "Learn Javascript Programming";
delete bookTitle;
```

Then the **bookTitle** variable will disappear from memory.

Practical Section 6 : Functions

Functions are sub-programs that can be reused both within the program itself, and in other programs. A function in Javascript is an object. Because it has properties and also *methods*.

Step	Description
1	Create a new File named **function_javascript.html** and save it in the project folder
2	How to call a function in Javascript code is usually written with: **functionName();**

3	Type the following program code
	
	``<!DOCTYPE html>
	<html>
	<head>
	    <script>
	    // membuat fungsi
	    var sayHello = () => alert("Hello World!");
	    </script>
	</head>
	<body>
	    <!-- Memanggil fungsi saat link diklik -->
	    Klik Aku!
	</body>
	</html>```

4	Observe what appears in the browser
5	Record your observations (Question No. 15)

| 6 | A parameter is a variable that stores a value for a process inside a function.<br>How to call a parameter in javascript is:<br><br>```javascript
function kali(a, b){
    hasilKali = a * b;
    console.log("Hasil kali a*b = " + hasilKali);
}
``` |
| 7 | Type the following program code<br><br>```html
<html>
<head>
<script type="text/javascript">
function total(numberA,numberB)
{
return numberA + numberB
}
</script>
</head>
<body>
<script type="text/javascript">
document.write(total(2,3))
</script>

</body>
</html>
``` |
| 8 | Observe what appears in the browser |
| 9 | Record your observations
(Question No. 16)

5 |

	The web page displays the number 5.

Practical Section 7: Data Types

Data types are the types of data that we can store in variables. There are several types of data in Javascript programming:

- String (text)
- Integer or Number
- Float (number of Fractions)
- Boolean
- Object

Javascript is a *dynamic typing* language, which means that we don't have to write data types when creating variables like in C, C++, Java, etc. which are *static typing*. There are several rules for writing variables in Javascript:

- Variable naming **should not** use numbers in front of it.
  example:
  ```
  // wrong
  var 123name = "Polinema";

  // right
  var name123 = "Polinema";
  ```

- Variable naming **can** use the initial underscore.
  example:
  ```
  var _nama = "Polinema";
  ```

- Variable naming **is recommended** using `camelCase` if it consists of two syllables.
  Example:
  ```
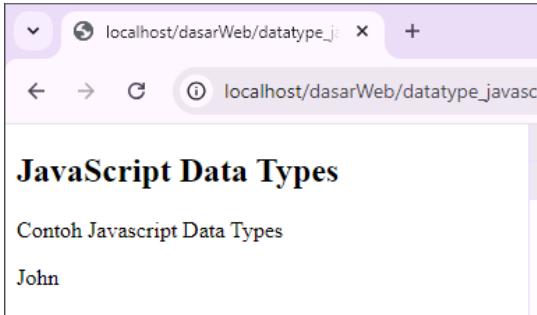  var _fullName = "Polinema";
  ```

- Variable naming **is recommended** using English
  Example:
  ```
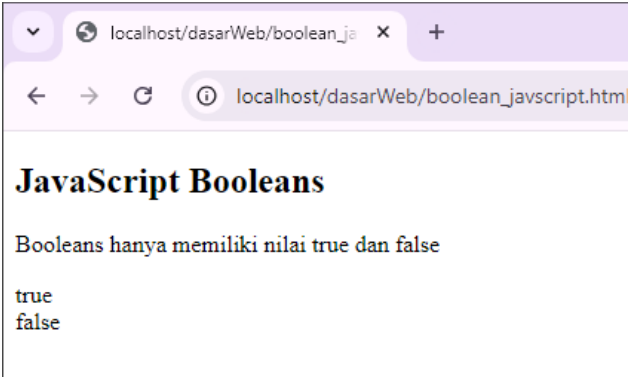  var _postTitle = "Javascript Tutorials";
  ```

Step	Description
1	Create a new File named `datatype_javascript.html` and save it in the project folder.
2	Type the following program code

| | ```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Data Types</h2>

<p>Contoh Javascript Data Types:</p>

<p id="demo"></p>

<script>
var x;          // Now x is undefined
x = 5;          // Now x is a Number
x = "John";     // Now x is a String

document.getElementById("demo").innerHTML = x;
</script>

</body>
</html>
``` |
3	Observe what appears in the browser
4	Record your observations (Question No. 17) localhost/dasarWeb/datatype_ja × + ← → C ⓘ localhost/dasarWeb/datatype_javasc **JavaScript Data Types** Contoh Javascript Data Types John The web page displays the text "John" in the paragraph with the id demo.
5	Type the program below and save it with the `string_javascript.html` name

```
<!DOCTYPE html>
<html>
<body>
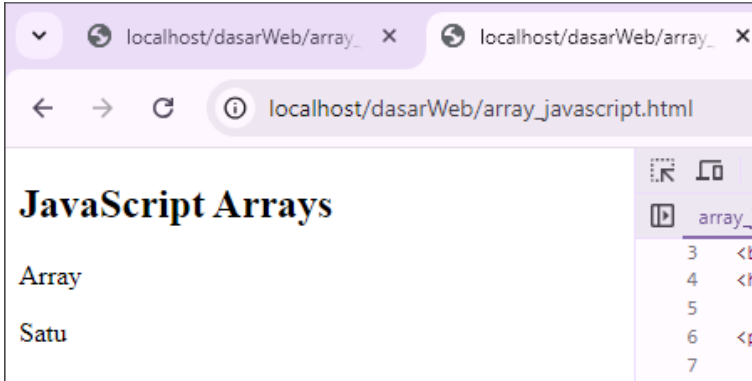
<h2>JavaScript Strings</h2>

<p>Membuat Javascript String</p>

<p id="demo"></p>

<script>
var answer1 = "It's alright";
var answer2 = "He is called 'Dilan'";
var answer3 = 'He is called "Dilan"';

document.getElementById("demo").innerHTML =
answer1 + "<br>" +
answer2 + "<br>" +
answer3;
</script>

</body>
</html>
```

6	Observe what appears in the browser
7	Record your observations

(Question No. 18)

> **localhost/dasarWeb/string_java** × +
>
> ← → C ⓘ localhost/dasarWeb/string_javascript.html#
>
> # JavaScript Strings
>
> Membuat JavaScript Srting
>
> It's alright
> He is called Dilan
> He is called Dilan

<mark>This code demonstrates how to create and manipulate string variables in JavaScript and how to update the content of an HTML element using document.getElementById().innerHTML</mark>

8	Type the program below and save it with the `boolean_javascript.html` name

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Booleans</h2>

<p>Booleans hanya memiliki nilai true dan false</p>

<p id="demo"></p>

<script>
var x = 5;
var y = 5;
var z = 6;
document.getElementById("demo").innerHTML =
(x == y) + "<br>" + (x == z);
</script>

</body>
</html>
```

9	Observe what appears in the browser
10	Record your observations (Question No. 19) localhost/dasarWeb/boolean_ja × + ← → C ⓘ localhost/dasarWeb/boolean_javscript.html **JavaScript Booleans** Booleans hanya memiliki nilai true dan false true false ==This code demonstrates how JavaScript boolean values work and how to display the results of boolean expressions on a web page using document.getElementById().innerHTML.==

11	Type the program below and save it with the **array_javascript.html** name
	
	``
	<!DOCTYPE html>
	<html>
	<body>
	
	<h2>JavaScript Arrays</h2>
	
	<p>Array</p>
	
	<p id="demo"></p>
	
	<script>
	var cars = ["Satu","Dua","Tiga"];
	
	document.getElementById("demo").innerHTML = cars[0];
	</script>
	
	</body>
	</html>
	``

12	Observe what appears in the browser
13	Record your observations (Question No.20) **JavaScript Arrays** Array Satu The web page displays the text: Satu

Practical Section 8: Operator

An operator is a symbol used to perform operations on a value and variable. Operators in programming are divided into 6 types:

1. Arithmetic operator;
2. Assignment Operator;
3. relationship or comparison operators;
4. Logic Operators;
5. Bitwise Operator;
6. Ternary Operator;

An arithmetic operator is an operator to perform arithmetic operations such as addition, subtraction, division, multiplication, etc. Arithmetic operators consist of:

Operator Name	Symbol
Addition	+
Reduction	-
Multiplication	*
Appointment	**
Division	/
Leftover	%

Step	Description

1	Create a new File named **operator_javascript.html** and save it in the project folder.
2	Type the program below

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Operators</h2>

<p>x = 5, y = 2, menghitung z = x + y, dan tampil z:</p>

<p id="demo"></p>

<script>
var x = 5;
var y = 2;
var z = x + y;
document.getElementById("demo").innerHTML = z;
</script>

</body>
</html>
```

3	Observe what appears in the browser
4	Record your observations (Question No. 21)

JavaScript Operators

x= 5, y = 2 , menghitung z = x + y, dan tampil z :

7

The web page displays the text:
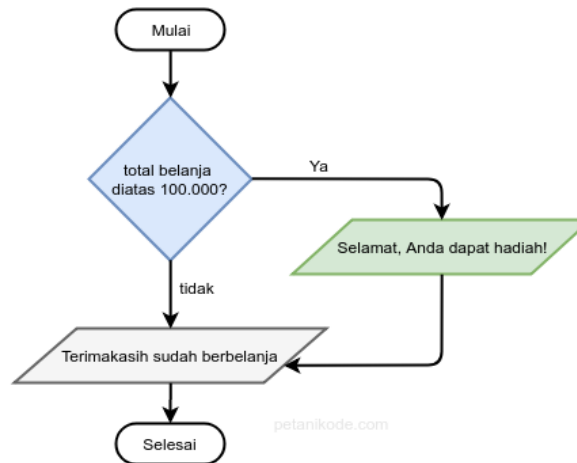
7

Practical Section 9: Branching

It can be said that branching and looping are one of the core methods in all programming languages worldwide. With branching and looping, a dynamic program can be created instead of a linear and static one. Since JavaScript is a method for client-side web programming, it also has this capability.

Some branching functions include:

- Use if to specify a block of code to be executed, if a specified condition is true
- Use else to specify a block of code to be executed, if the same condition is false
- Use else if to specify a new condition to test, if the first condition is false
- Use switch to specify many alternative blocks of code to be executed

❖ if Branching

if branching is a structure that only has one block of choice when the condition is true. Take a look at the following flowchart:

"If the total purchase is greater than Rp 100,000, then display the message: Congratulations, you won a prize."

What if it is below Rp 100,000?

Yes, the message is not displayed.

Step	Description
1	Create a new File named **if_javascript.html** and save it in the project folder

2	Type the program below
	
	 ``html
	<!DOCTYPE html>
	<html lang="en">
	<head>
	    <title>Percabangan if</title>
	</head>
	<body>
	    <script>
	        var totalBelanja = prompt("Total belanja?", 0);
	
	        if(totalBelanja > 30000){
	            document.write(" Selamat Anda dapat hadiah ");
	        }
	
	        document.write(" Terimakasih sudah berbelanja di toko kami ");
	    </script>
	</body>
	</html>
	``

3	Observe what appears in the browser

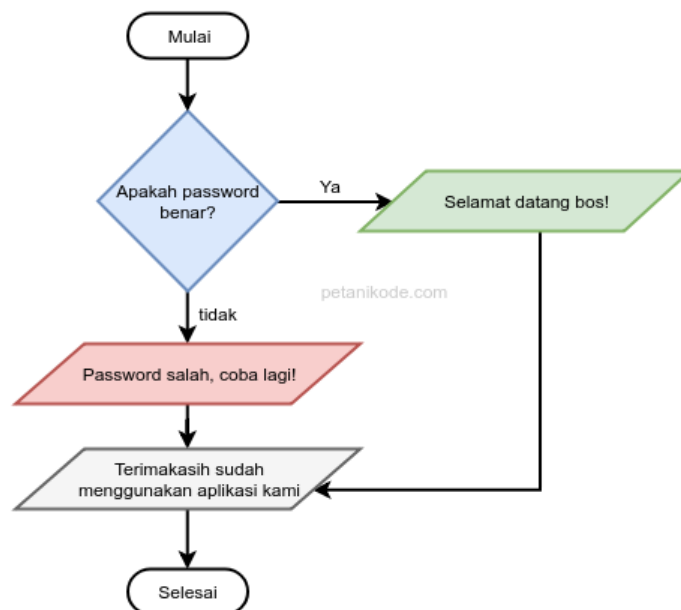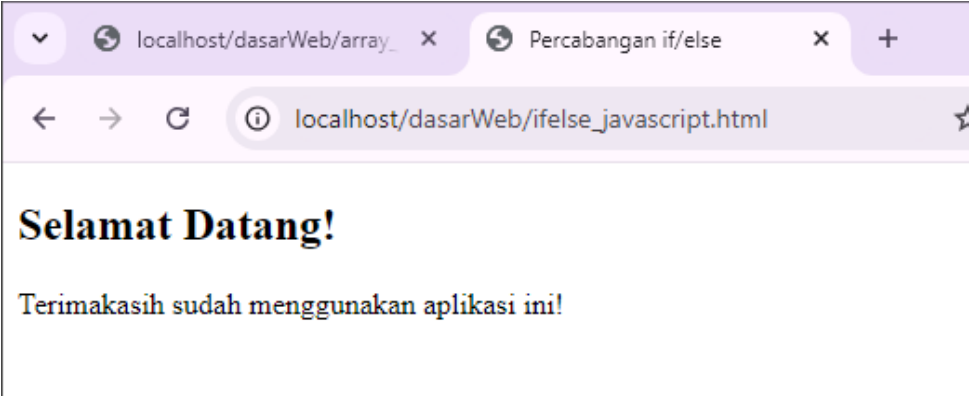| 4	Record your observations
	(Question No. 22)
	A prompt dialog box appears asking for the total amount spent. If the entered amount is greater than 30,000, the web page displays:
	Selamat Anda dapat hadiah
	The web page always displays:
	Terimakasih sudah berbelanja di toko kami

❖ if/else Branching

If/Else Branching is a structure that has **two blocks of choices**. The first choice is for when the **condition is true**, and the second choice is for when the **condition is false (else)**. Take a look at this flowchart:

This is a flowchart for checking the password. If the password is correct, the message in the green block will be displayed: **"Welcome, boss!"** However, if it is incorrect, the message in the red block will be shown: **"Incorrect password, please try again!"**

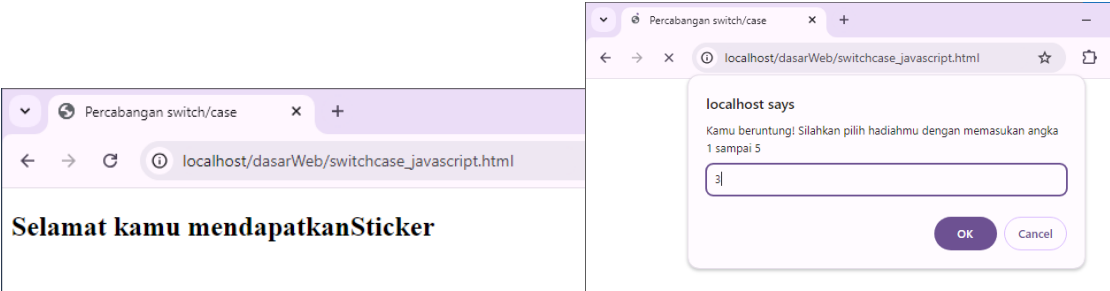| Step	Description
1	Create a new File named `ifelse_javascript.html` and save it in the project folder.

2	Type the program below
	
	``html
	<!DOCTYPE html>
	<html>
	<head>
	    <title>Percabangan if/else</title>
	</head>
	<body>
	    <script>
	        var password = prompt("Password:");
	
	        if(password == "teh"){
	            document.write(" Selamat datang ! ");
	        } else {
	            document.write(" Password salah, coba lagi! ");
	        }
	
	        document.write(" Terima kasih sudah menggunakan aplikasi ini! ");
	
	    </script>
	</body>
	</html>
	``

3	Observe what appears in the browser
4	Record your observations (Question No. 23) **Selamat Datang!** Terimakasih sudah menggunakan aplikasi ini! ==A prompt dialog box appears asking for the password.== ==If the password is "teh", the web page displays:== ==Selamat Datang!== ==Terimakasih sudah menggunakan aplikasi ini!== ==If the password is not "teh", the web page displays:== ==Terimakasih sudah menggunakan aplikasi ini!== ==Terimakasih sudah menggunakan aplikasi ini!==

❖ switch/case Branching

switch/case branching is an alternative form of the **if/else/if** branching structure. In a **switch/case** statement, instead of evaluating multiple if conditions, the program evaluates the value of a variable or expression and compares it against multiple possible cases. Each case represents a potential value, and when a match is found, the corresponding block of code is executed. If no case matches, the **default** case is executed (if provided), similar to the **else** block in **if/else** statements.

The **switch/case** structure can make code more readable and organized, especially when dealing with multiple conditions based on a single variable. The structure looks like this:

```
switch(variabel){
    case <value>:
        // blok kode
        break;
    case <value>:
        // blok kode
        break;
    default:
        // blok kode
}
```
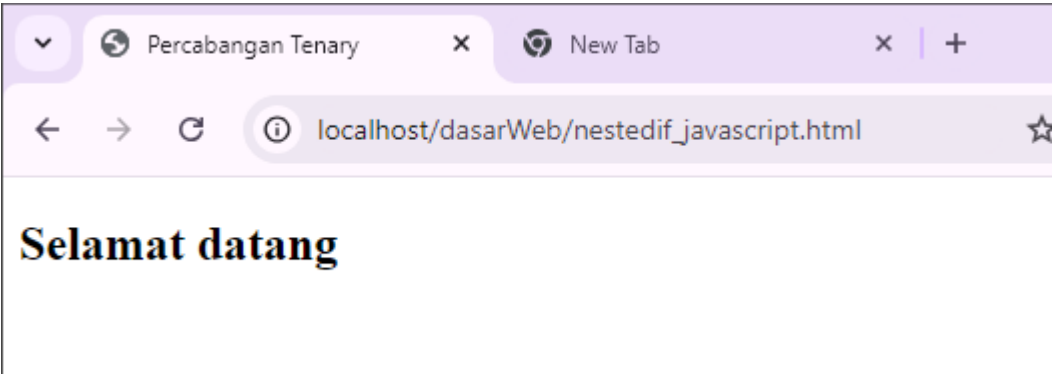
Step	Description
1	Create a new File named `switchcase_javascript.html` and save it in the project folder
2	Type the program below

```html
<!DOCTYPE html>
<html>
<head>
    <title>Percabangan switch/case</title>
</head>
<body>
    <script>
        var jawab = prompt("Kamu beruntung! Silahakn pilih hadiahmu dengan memasukan angka 1 sampai 5");
        var hadiah = "";
            switch(jawab){
            case "1":
                hadiah = "Tisu";
                break;
            case "2":
                hadiah = "1 Kotak Kopi";
                break;
            case "3":
                hadiah = "Sticker";
                break;
            case "4":
                hadiah = "Minyak Goreng";
                break;
            case "5":
                hadiah = "Uang Rp 50.000";
                break;
            default:
                document.write("<p>Opps! anda salah pilih</p>");}
        if(hadiah === ""){
            document.write("<p>Kamu gagal mendapat hadiah</p>");
        } else {
            document.write("<h2>Selamat kamu mendapatkan " + hadiah + "</h2>");
        }
    </script>
</body>
</html>
```

3	Observe what appears in the browser
4	Record your observations

(Question No. 24)

Selamat kamu mendapatkanSticker

A prompt dialog box appears asking for a number between 1 and 5.

If a valid number is entered, the web page displays a message indicating the prize won.

If an invalid number is entered, the web page displays "Opps! anda salah pilih" and "Kamu gagal mendapat hadiah".

❖ Nested Branching

Nested Branching refers to a condition where one branching statement (such as `if`, `else`, `switch`, etc.) is placed inside another branching statement. This allows for more complex decision-making processes where multiple conditions need to be evaluated at different levels. In nested branching, the outcome of one condition can depend on the result of another, providing more fine-grained control over the program flow.

For example, you can nest an `if` statement inside another `if` statement to first check one condition and then, based on that, check a second condition within the first block.

Step	Description
1	Create a new File named `nestedif_javascript.html` and save it in the project folder

2	Type the program below
	
	``html
	<!DOCTYPE html>
	<html>
	<head>
	    <title>Percabangan Ternary</title>
	</head>
	<body>
	    <script>
	        var username = prompt("Username:");
	        var password = prompt("Password:");
	
	        if(username == "mahasiswa"){
	            if(password == "kopi"){
	                document.write(" Selamat datang ");
	            } else {
	                document.write(" Password salah, coba lagi! ");
	            }
	        } else {
	            document.write(" Anda tidak terdaftar! ");
	        }
	    </script>
	</body>
	</html>
	``

3	Observe what appears in the browser
4	Record your observations (Question No. 25) localhost/dasarWeb/nestedif_javascript.html **Selamat datang**

Practical Section 10: Loops

Loops help us execute code repeatedly, as many times as we want. There are five types of loops in JavaScript. Generally, these loops are categorized into two types: counted loops and uncounted loops. The difference is as follows:

✓ **Counted Loops** are loops where the number of iterations is **known** and **definite**.

✓ **Uncounted Loops**, on the other hand, are loops where the number of iterations is **not predetermined**.

The loops that fall under **Counted Loops** are:

1. For Loop
2. Foreach Loop
3. Repeat Loop

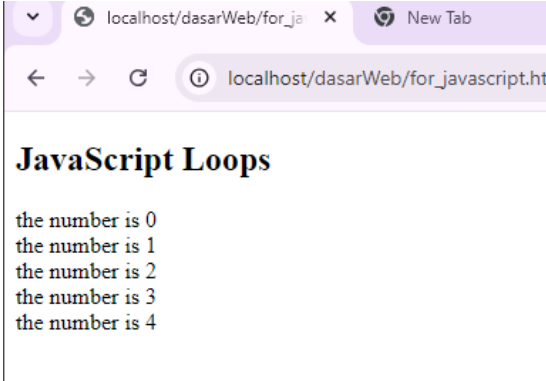The loops that fall under **Uncounted Loops** are:

1. While Loop
2. Do/While Loop

❖ **For loops in Javascript**

A for loop is a loop that is included in a *couted loop*, because it is clear how many times it will repeat. It looks like this:

```javascript
for(let i = 0; i < 10; i++){
    document.write("<p>Perulangan ke-" + i + "</p>")
}
```

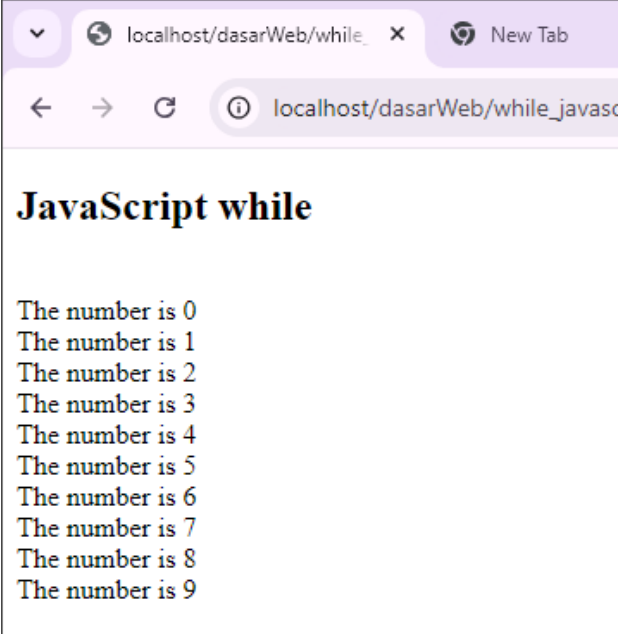| Step	Description
1	Create a new File named **for_javascript.html** and save it in the project folder
2	Type the program below

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Loops</h2>

<p id="demo"></p>

<script>
var text = "";
var i;
for (i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

3	Observe what appears in the browser
4	Record your observations (Question No. 26) JavaScript Loops the number is 0 the number is 1 the number is 2 the number is 3 the number is 4 uses a for loop to iterate from 0 to 4 and appends the text to the text variable

❖ While loops in Javascript

The while loop is categorized as an uncounted loop. However, the while loop can also function as a counted loop by including a counter within it.

Step	Description
1	Create a new File named **while_javascript.html** and save it in the project folder
2	Type the program below

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript while</h2>

<p id="demo"></p>

<script>
var text = "";
var i = 0;
while (i < 10) {
  text += "<br>The number is " + i;
  i++;
}
document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

3	Observe what appears in the browser
4	Record your observations (Question No. 27) JavaScript while The number is 0 The number is 1 The number is 2 The number is 3 The number is 4 The number is 5 The number is 6 The number is 7 The number is 8 The number is 9 It uses a while loop to iterate from 0 to 9, appending the text to the text variable.

❖ Do/While Loops in Javascript

The **do/while loop** is a variation of the while loop in JavaScript. The main difference between them is that the do/while loop will always execute the code inside the loop **at least once**, regardless of whether the
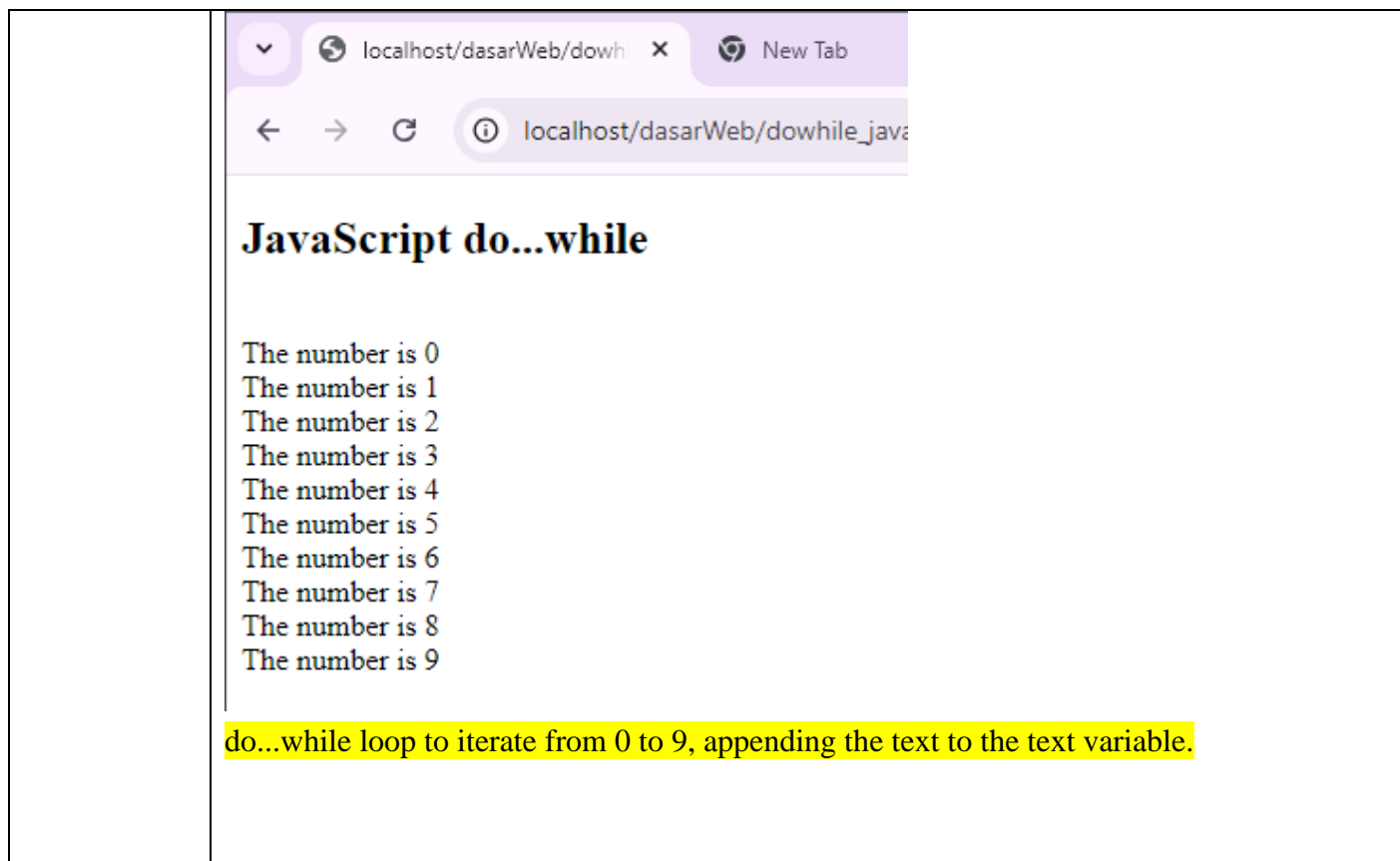
condition is true or false. This is because the condition is evaluated **after** the code block is executed, not before, as in the standard while loop.:

```
do {
    // blok kode yang akan diulang
} while (<kondisi>);
```

Key Characteristics:
- The code inside the do block runs first, and then the condition is checked.
- If the condition is true, the loop repeats; if false, the loop stops.
- This type of loop ensures that the code inside the loop executes at least once, even if the condition is false from the beginning.

Step	Description
1	Create a new File named **dowhile_javascript.html** and save it in the project folder

2	Type the program below
	
	``html
	<!DOCTYPE html>
	<html>
	<body>
	
	<h2>JavaScript do ... while</h2>
	
	<p id="demo"></p>
	
	<script>
	var text = ""
	var i = 0;
	
	do {
	  text += "
	The number is " + i;
	  i++;
	}
	while (i < 10);
	
	document.getElementById("demo").innerHTML = text;
	</script>
	
	</body>
	</html>
	``

3	Observe what appears in the browser
4	Record your observations (Question No. 28)

JavaScript do...while

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9

do...while loop to iterate from 0 to 9, appending the text to the text variable.

Reference:

1) Jason Beaird, The principles of Beautiful Web Design
2) Rian Ariona, Learn HTML and CSS (Fundamental Tutorial in Learning HTML and CSS)
3) Adi Hadisaputra, HTML and CSS Fundamentals from the Roots to the Leaves of John Duckett, HTML and CSS design and build websites