**Tugas 3**

**Nurul Karimah**

**Materi Looping (Perulangan) di JavaScript**

**looping** adalah salah satu konsep fundamental yang perlu kita pelajari. Ia sangat berguna untuk mengerjakan tugas-tugas yang berulang sehingga kita tidak perlu menulis kode yang sama berulang kali.

Looping adalah proses mengulang eksekusi sebuah blok kode beberapa kali.  
Looping sangat penting untuk tugas seperti:

* Melakukan iterasi pada nilai
* Melakukan iterasi pada struktur data
* Melakukan sebuah aksi berulang kali

**Apa Itu Looping?**

Looping adalah salah satu konsep fundamental dalam pemrograman yang digunakan untuk menjalankan serangkaian perintah atau kode secara berulang hingga kondisi tertentu terpenuhi. Dengan menggunakannya, kita dapat menghemat waktu dan usaha dalam menulis kode yang harus dijalankan berulang kali, terutama ketika jumlah perulangannya banyak atau tidak diketahui sebelumnya.

Looping memungkinkan programmer untuk mengotomatiskan tugas-tugas yang bersifat repetitif. Misalnya, Anda ingin mencetak angka 1 hingga 10 di layar. Daripada menulis perintah *console.log* sebanyak 10 kali, Anda cukup menulis kode sederhana dengan menggunakan loop. Selain itu, ia juga membantu meningkatkan efisiensi, meminimalkan kesalahan, dan membuat kode lebih mudah dikelola.

**Jenis – Jenis Looping**

Ada berbagai jenis yang bisa kita gunakan umumnya setiap bahasa pemrograman memiliki konsep-konsep looping yang kurang lebih sama, tetapi syntax-nya berbeda-beda. Contoh-contoh di bawah akan menggunakan bahasa pemrograman [Javascript](https://www.dicoding.com/academies/256).

**Jenis-jenis Loop di JavaScript**

| **Jenis Loop** | **Deskripsi** |
| --- | --- |
| for | Melakukan iterasi berdasarkan ekspresi dan nilai tertentu |
| while | Melakukan iterasi selama kondisi bernilai benar (true) |
| do...while | Sama dengan while, tetapi menjamin blok kode dijalankan minimal sekali |
| for...in | Melakukan iterasi pada properti milik sebuah **Object** |
| for...of | Melakukan iterasi pada nilai dari **objek iterable** (Array, String, Map, Set, dll.) |
| forEach() | Melakukan iterasi pada setiap elemen dalam sebuah Array |

**1. For Loop**

Digunakan saat jumlah perulangan sudah diketahui.

**Sintaks:**

for (exp1; exp2; exp3) {

// blok kode yang akan dijalankan

}

exp1: dijalankan **sekali** sebelum perulangan dimulai (inisialisasi variabel).

exp2: kondisi untuk menjalankan perulangan. Jika false, loop berhenti.

exp3: dijalankan setiap kali setelah blok kode selesai dieksekusi (biasanya increment).

**Contoh:**

for (let i = 0; i < 5; i++) {

console.log("Angka ke-" + i);

}

**2. While Loop**

Digunakan saat jumlah perulangan **belum diketahui**, tapi tergantung kondisi.

**Sintaks:**

while (kondisi) {

// blok kode yang akan dijalankan

}

**Contoh:**

let i = 0;

while (i < 5) {

console.log("Angka ke-" + i);

i++;

}

**Catatan:**  
Jika variabel kondisi tidak diubah (misalnya i++ lupa ditulis), maka perulangan tidak akan berhenti (**infinite loop**) dan bisa membuat browser crash.

**3. Do...While Loop**

Mirip dengan while, tetapi blok kode dijalankan minimal **satu kali**, meskipun kondisi salah.

**Sintaks:**

do {

// blok kode yang akan dijalankan

} while (kondisi);

**Contoh:**

let i = 0;

do {

console.log("Angka ke-" + i);

i++;

} while (i < 5);

**4. For...in Loop**

Digunakan untuk melakukan iterasi pada properti (key) dalam sebuah **Object**.

**Sintaks:**

for (let key in object) {

// blok kode

}

**Contoh:**

const person = { fname: "John", lname: "Doe", age: 25 };

for (let x in person) {

console.log(x + ": " + person[x]);

}

**5. For...of Loop**

Digunakan untuk melakukan iterasi pada **nilai** dari objek yang bisa diiterasi (Array, String, Map, Set, dll.).

**Sintaks:**

for (let value of iterable) {

// blok kode

}

**Contoh:**

const cars = ["BMW", "Volvo", "Saab"];

for (let car of cars) {

console.log(car);

}

**6. forEach() Method**

Digunakan khusus pada **Array** untuk menjalankan fungsi callback pada setiap elemen.

**Contoh:**

const numbers = [1, 2, 3];

numbers.forEach(function(num) {

console.log(num \* 2);

});

**Kontrol Alur dalam Loop**

* break → Menghentikan loop sepenuhnya.
* continue → Melompati iterasi saat ini, lalu melanjutkan ke iterasi berikutnya.

**Contoh:**

for (let i = 0; i < 5; i++) {

if (i === 2) continue; // skip angka 2

if (i === 4) break; // stop loop di angka 4

console.log(i);

}

**For...of Loop**

for...of adalah pernyataan JavaScript yang digunakan untuk melakukan perulangan pada **nilai** dari sebuah objek yang bersifat **iterable**.

Loop ini dapat digunakan pada struktur data seperti:

* Array
* String
* Map
* NodeList
* dan objek iterable lainnya

**Sintaks**

for (variable of iterable) {

// blok kode yang akan dijalankan

}

* **variable** → Pada setiap iterasi, nilai dari elemen berikutnya akan disimpan ke dalam variable. Variabel bisa dideklarasikan dengan const, let, atau var.
* **iterable** → Objek yang memiliki properti iterable (dapat diulang).

**Contoh: Looping pada Array**

const cars = ["BMW", "Volvo", "Mini"];

let text = "";

for (let x of cars) {

text += x + " ";

}

console.log(text); // Output: BMW Volvo Mini

**Contoh: Looping pada String**

let language = "JavaScript";

let text = "";

for (let x of language) {

text += x + " ";

}

console.log(text); // Output: J a v a S c r i p t

**Perulangan (Loops) dalam JavaScript**

Loop (perulangan) memberikan cara cepat dan mudah untuk melakukan sesuatu secara berulang. Bagian ini dari *JavaScript Guide* memperkenalkan berbagai macam pernyataan iterasi (*iteration statements*) yang tersedia di JavaScript.

Anda bisa membayangkan loop sebagai versi komputerisasi dari permainan di mana Anda menyuruh seseorang untuk melangkah X kali ke satu arah, kemudian Y kali ke arah lain.  
Misalnya, ide **"Melangkah lima kali ke timur"** bisa diekspresikan dalam bentuk loop seperti berikut:

for (let step = 0; step < 5; step++) {

// Dieksekusi 5 kali, dengan nilai step dari 0 sampai 4

console.log("Melangkah satu langkah ke timur");

}

Ada banyak jenis loop, tetapi intinya sama: **mengulang suatu aksi beberapa kali** (bisa saja jumlahnya nol).

Berbagai mekanisme loop menawarkan cara berbeda untuk menentukan titik awal dan akhir dari perulangan. Situasi tertentu lebih mudah ditangani dengan satu jenis loop dibandingkan yang lain.

**Jenis pernyataan loop di JavaScript**

* for statement
* do...while statement
* while statement
* labeled statement
* break statement
* continue statement
* for...in statement
* for...of statement

**1. for statement**

for loop akan mengulang sampai kondisi tertentu bernilai **false**. Sintaksnya mirip dengan Java dan C.

**Sintaks**

for (initialization; condition; afterthought)

statement

**Proses eksekusi**

1. **initialization** → dieksekusi sekali di awal (biasanya untuk inisialisasi variabel penghitung).
2. **condition** → diperiksa, jika true, maka blok kode dijalankan. Jika false, loop berhenti. (Jika condition dihilangkan, nilainya dianggap true).
3. **statement** → blok kode yang akan dijalankan. Jika lebih dari satu baris, bungkus dengan { }.
4. **afterthought** → dijalankan setiap kali satu putaran loop selesai (biasanya untuk increment/decrement).
5. Kembali lagi ke langkah 2.

**Contoh**

Hitung berapa banyak opsi yang dipilih dari sebuah <select>:

<form name="selectForm">

<label for="musicTypes">Pilih beberapa genre musik:</label>

<select id="musicTypes" name="musicTypes" multiple>

<option selected>R&B</option>

<option>Jazz</option>

<option>Blues</option>

<option>New Age</option>

<option>Klasik</option>

<option>Opera</option>

</select>

<button id="btn" type="button">Berapa yang dipilih?</button>

</form>

function countSelected(selectObject) {

let numberSelected = 0;

for (let i = 0; i < selectObject.options.length; i++) {

if (selectObject.options[i].selected) {

numberSelected++;

}

}

return numberSelected;

}

const btn = document.getElementById("btn");

btn.addEventListener("click", () => {

const musicTypes = document.selectForm.musicTypes;

console.log(`Anda memilih ${countSelected(musicTypes)} opsi.`);

});

**2. do...while statement**

do...while mengulang sampai kondisi bernilai **false**, tetapi **pasti dijalankan minimal sekali**.

**Sintaks**

do

statement

while (condition);

**Contoh**

let i = 0;

do {

i += 1;

console.log(i);

} while (i < 5);

**3. while statement**

while menjalankan blok kode selama kondisi bernilai **true**.

**Sintaks**

while (condition)

statement

**Contoh 1**

let n = 0;

let x = 0;

while (n < 3) {

n++;

x += n;

}

console.log(x); // Output: 6

**Contoh 2 (loop tak terbatas - jangan lakukan ini!)**

while (true) {

console.log("Hello, world!");

}

**4. labeled statement**

Label memberi nama pada sebuah pernyataan agar bisa direferensikan di tempat lain, misalnya dengan break atau continue.

**Sintaks**

label:

statement

**5. break statement**

Digunakan untuk **menghentikan loop** atau keluar dari switch.

* break; → menghentikan loop/switch terdekat.
* break label; → menghentikan loop/labeled statement tertentu.

**Contoh 1**

for (let i = 0; i < a.length; i++) {

if (a[i] === theValue) {

break;

}

}

**Contoh 2 (break dengan label)**

labelCancelLoops: while (true) {

// kode ...

if (x === 10 && z === 10) {

break labelCancelLoops;

}

}

**6. continue statement**

Digunakan untuk **melewatkan satu iterasi** dan lanjut ke iterasi berikutnya.

* continue; → skip iterasi sekarang, lanjut ke berikutnya.
* continue label; → skip iterasi dari loop tertentu yang diberi label.

**Contoh 1**

let i = 0;

let n = 0;

while (i < 5) {

i++;

if (i === 3) {

continue; // lewati iterasi saat i=3

}

n += i;

console.log(n);

}

// Output: 1, 3, 7, 12

**7. for...in statement**

for...in digunakan untuk mengulang **property names (keys)** dari sebuah objek.

**Sintaks**

for (variable in object)

statement

**Contoh**

const person = {fname:"John", lname:"Doe", age:25};

for (const key in person) {

console.log(key, "=", person[key]);

}

**Catatan:**  
for...in bisa juga mengulang array, tapi hasilnya termasuk properti tambahan. Jadi, lebih baik gunakan for atau for...of untuk array.

**8. for...of statement**

for...of digunakan untuk mengulang **nilai** dari objek iterable (Array, Map, Set, String, dll).

**Sintaks**

for (variable of iterable)

statement

**Perbedaan for...in vs for...of**

const arr = [3, 5, 7];

arr.foo = "hello";

for (const i in arr) {

console.log(i);

}

// Output: "0", "1", "2", "foo"

for (const i of arr) {

console.log(i);

}

// Output: 3, 5, 7