

LOGBOOK SUPLEMEN PRAKTIKUM — DASAR PEMROGRAMAN JAVASCRIPT

Mata Kuliah: Aplikasi Web dan Mobile | Program Studi Teknik Industri | Universitas Negeri Yogyakarta
Persiapan Pertemuan 4 | Semester Genap 2025/2026

SUPLEMEN PRAKTIKUM · TEKNIK INDUSTRI UNY

Logbook Dasar Pemrograman JavaScript

Identitas Mahasiswa

NAMA LENGKAP

Maessa Andrea Vallenia

NIM

23051430044

KELAS / ROMBEL

TI-A 2023

TANGGAL Pengerjaan

27/02/2026



DOSEN PENGAMPU

Dr. Eng. Ir. Aji Ery Burhandenny, S.T., M.AIT.

BAGIAN A

Checklist Materi Suplemen

Centang setiap topik setelah Anda membaca, memahami, dan mencoba contoh kodenya di Browser Console.



Bagian 1 & 2 — Mengapa JavaScript & cara kerja di browser

Konsep

Saya memahami peran JS dalam tiga pilar web dan dapat membuka Browser Console (F12)



Bagian 3 — Variabel: let, const, dan aturan penamaan

Praktik

Saya sudah mencoba mendeklarasikan variabel dan memahami kapan memakai let vs const



Bagian 4 — Tipe Data: Number, String, Boolean

Praktik

Saya memahami jebakan String + Number dan cara konversi dengan Number()



Bagian 5 — Operator: Aritmatika, Perbandingan (===), Logika (&&, ||)

Praktik

Saya sudah mencoba minimal satu perhitungan dan satu ekspresi perbandingan



**Bagian 6 — Control Flow: if, else if, switch, ternary**

Praktik

Saya memahami urutan else if dan kenapa break diperlukan di switch**Bagian 7 — Studi Kasus Kalkulator OEE (membaca dan memahami seluruh kode)**

Analisis

Saya dapat menjelaskan setiap variabel dan setiap blok if dalam kode OEE tersebut**Bagian 9 — Membaca seluruh daftar Kesalahan Umum Pemula**

Review

Saya sudah mengidentifikasi minimal dua kesalahan yang pernah atau mungkin saya lakukan

BAGIAN A2

Bukti Pengerjaan Latihan Mandiri

Untuk setiap latihan: (1) centang setelah selesai, (2) jawab pertanyaan uji pemahaman singkat, dan (3) unggah screenshot output Console sebagai bukti pengerjaan yang akan tercetak di PDF.

Level 1 — Latihan Dasar

**Latihan 1.1 — Variabel Profil Mesin**

Level 1

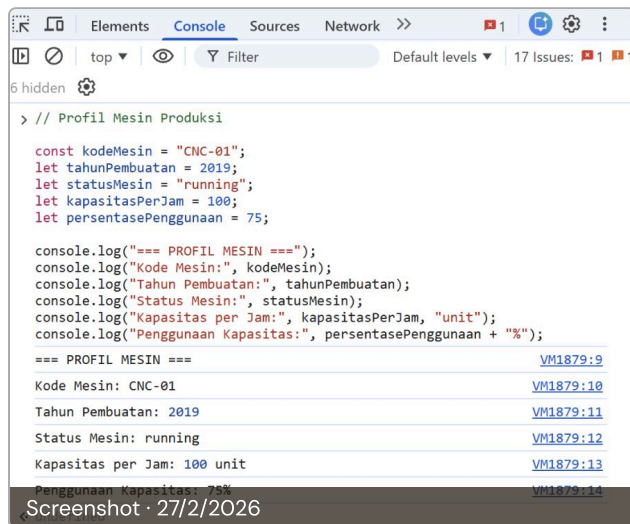
Membuat variabel dengan tipe tepat (let/const) dan menampilkan semua dengan console.log()

UJI PEMAHAMAN SINGKAT

Mengapa variabel "kode mesin" sebaiknya dideklarasikan dengan `const` bukan `let`? Jelaskan singkat.

Variabel kode mesin lebih tepat menggunakan `const` daripada `let` karena nilainya tidak berubah selama program berjalan. Jika menggunakan `let`, nilai variabel masih bisa diubah, padahal kode mesin berfungsi sebagai identitas tetap. Dengan menggunakan `const`, kita memastikan bahwa nilai tersebut tidak dapat dimodifikasi, sehingga program menjadi lebih aman dan mengurangi risiko kesalahan akibat perubahan data yang tidak disengaja.

SCREENSHOT OUTPUT CONSOLE



```
> // Profil Mesin Produksi

const kodeMesin = "CNC-01";
let tahunPembuatan = 2019;
let statusMesin = "running";
let kapasitasPerJam = 100;
let persentasePenggunaan = 75;

console.log("=== PROFIL MESIN ===");
console.log("Kode Mesin:", kodeMesin);
console.log("Tahun Pembuatan:", tahunPembuatan);
console.log("Status Mesin:", statusMesin);
console.log("Kapasitas per Jam:", kapasitasPerJam, "unit");
console.log("Penggunaan Kapasitas:", persentasePenggunaan + "%");

=== PROFIL MESIN ===
Kode Mesin: CNC-01
Tahun Pembuatan: 2019
Status Mesin: running
Kapasitas per Jam: 100 unit
Penggunaan Kapasitas: 75%
```

Screenshot · 27/2/2026



Latihan 1.2 — Kalkulator Biaya Material

Level 1

Menghitung biaya per unit, total biaya, dan berat total material; tampilkan di console



UJI PEMAHAMAN SINGKAT

Jika jumlahProduksi diubah menjadi 0, apa yang terjadi pada biayaPerUnit? Apa masalah matematis yang muncul?

Jika jumlahProduksi diubah menjadi 0, maka biayaPerUnit tetap tidak berubah, karena perhitungannya berasal dari hargaBahanPerKg × kebutuhanPerUnitKg dan tidak bergantung pada jumlah produksi.

Namun, masalah matematis akan muncul apabila dilakukan perhitungan seperti:

$\text{biayaPerUnit} = \text{totalBiayaProduksi} / \text{jumlahProduksi}$

Jika jumlahProduksi = 0, maka terjadi pembagian dengan nol, yang secara matematis tidak terdefinisi. Dalam JavaScript, kondisi ini dapat menghasilkan nilai Infinity atau NaN. Oleh karena itu, diperlukan validasi agar jumlahProduksi tidak bernilai nol sebelum melakukan operasi pembagian.

SCREENSHOT OUTPUT CONSOLE

```
> {  
  // Perhitungan Biaya Material  
  
  const hargaBahanPerKg = 12000;  
  const kebutuhanPerUnitKg = 0.5;  
  const jumlahProduksi = 200;  
  
  let biayaPerUnit = hargaBahanPerKg * kebutuhanPerUnitKg;  
  let totalBiayaProduksi = biayaPerUnit * jumlahProduksi;  
  let totalBeratMaterial = kebutuhanPerUnitKg * jumlahProduksi;  
  
  console.log("=== PERHITUNGAN BIAYA ===");  
  console.log("Biaya per Unit: Rp", biayaPerUnit);  
  console.log("Total Biaya Produksi: Rp", totalBiayaProduksi);  
  console.log("Total Berat Material:", totalBeratMaterial, "kg");  
}  
=== PERHITUNGAN BIAYA === VM1904:12  
Biaya per Unit: Rp 6000 VM1904:13  
Total Biaya Produksi: Rp 1200000 VM1904:14  
Total Berat Material: 100 kg VM1904:15
```

Screenshot · 27/2/2026

Level 2 — Latihan Menengah



Latihan 2.1 — Sistem Klasifikasi Reject

Level 2

Logika if/else if/else menentukan kategori (Excellent/Acceptable/Warning/Critical) dari reject rate



UJI PEMAHAMAN SINGKAT

Jika Anda membalik urutan kondisi (cek ≥ 5 dulu, baru ≥ 3), apakah hasilnya berbeda? Jelaskan mengapa urutan kondisi `else if` sangat penting.

Ya, hasilnya bisa berbeda. Misalnya jika nilai `rejectRate` = 6% dan kondisi ditulis ≥ 3 terlebih dahulu, maka 6% sudah memenuhi syarat ≥ 3 sehingga langsung dikategorikan sebagai Warning. Padahal seharusnya 6% masuk kategori ≥ 5 yaitu Critical. Hal ini terjadi karena `else if` dibaca dari atas ke bawah dan akan berhenti pada kondisi pertama yang bernilai benar. Oleh karena itu, urutan kondisi sangat penting agar tidak terjadi kesalahan klasifikasi.

SCREENSHOT OUTPUT CONSOLE — UJI MINIMAL 2 SKENARIO

```
> {
  function klasifikasiReject(totalProduksi, jumlahReject) {
    const rejectRate = (jumlahReject / totalProduksi) * 100;

    console.log("Reject Rate:", rejectRate.toFixed(2) + "%");

    if (rejectRate < 1) {
      console.log("Kategori: EXCELLENT");
    } else if (rejectRate < 3) {
      console.log("Kategori: ACCEPTABLE");
    } else if (rejectRate < 5) {
      console.log("Kategori: WARNING");
    } else {
      console.log("Kategori: CRITICAL");
    }

    console.log("-----");
  }

  klasifikasiReject(500, 2); // 0.4% → Excellent
  klasifikasiReject(500, 10); // 2% → Acceptable
}
```

```
Reject Rate: 0.40% VM1925:5
Kategori: EXCELLENT VM1925:8
----- VM1925:17
Reject Rate: 2.00% VM1925:5
Kategori: ACCEPTABLE VM1925:10
```

Screenshot · 27/2/2026

VM1925:17

```
> {
  function klasifikasiReject(totalProduksi, jumlahReject) {
    const rejectRate = (jumlahReject / totalProduksi) * 100;

    console.log("Reject Rate:", rejectRate.toFixed(2) + "%");

    if (rejectRate < 1) {
      console.log("Kategori: EXCELLENT");
    } else if (rejectRate < 3) {
      console.log("Kategori: ACCEPTABLE");
    } else if (rejectRate < 5) {
      console.log("Kategori: WARNING");
    } else {
      console.log("Kategori: CRITICAL");
    }

    console.log("-----");
  }

  klasifikasiReject(500, 20); // 4% → Warning
  klasifikasiReject(500, 40); // 8% → Critical
}
```

```
Reject Rate: 4.00% VM1930:5
Kategori: WARNING VM1930:12
----- VM1930:17
Reject Rate: 8.00% VM1930:5
Kategori: CRITICAL VM1930:14
```

Screenshot · 27/2/2026

VM1930:17



Latihan 2.2 — Kalkulator Lembur

Level 2

Menghitung total upah lembur berdasarkan jam lembur, dengan tarif 1.5x dan 2x



UJI PEMAHAMAN SINGKAT

Berapa total lembur (Rp) untuk operator dengan gaji pokok Rp 3.500.000 yang lembur 5 jam? Tulis perhitungan manual Anda.

Gaji pokok = Rp3.500.000

Jam kerja = 173

Upah per jam = $3.500.000 / 173$

= Rp20.231

Jam 1–3 = $1,5 \times 20.231 \times 3$

= Rp91.039

Jam 4–5 = $2 \times 20.231 \times 2$

= Rp80.924

Total = $91.039 + 80.924$

= Rp171.963

Jadi total lembur untuk 5 jam adalah Rp171.963.

SCREENSHOT OUTPUT CONSOLE — UJI MINIMAL 2 SKENARIO

```
> {
  const gajiPokok = 3500000;
  const jamLembur = 2; // ≤ 3 jam

  const upahPerJam = gajiPokok / 173;
  const totalLembur = 1.5 * upahPerJam * jamLembur;

  console.log("=== SKENARIO ≤ 3 JAM ===");
  console.log("Upah per jam = Rp", upahPerJam.toFixed(0));
  console.log("Total lembur (1.5x saja) = Rp", totalLembur.toFixed(0));
}

=== SKENARIO ≤ 3 JAM ===
Upah per jam = Rp 20231
Total lembur (1.5x saja) = Rp 60694
```

Screenshot · 27/2/2026

```
> {
  const gajiPokok = 3500000;
  const jamLembur = 5; // > 3 jam

  const upahPerJam = gajiPokok / 173;

  const jamAwal = 3;
  const sisaJam = jamLembur - 3;

  const lembur15x = 1.5 * upahPerJam * jamAwal;
  const lembur2x = 2 * upahPerJam * sisaJam;

  const totalLembur = lembur15x + lembur2x;

  console.log("=== SKENARIO > 3 JAM ===");
  console.log("Upah per jam = Rp", upahPerJam.toFixed(0));
  console.log("Lembur 1-3 jam (1.5x) = Rp", lembur15x.toFixed(0));
  console.log("Lembur 4-5 jam (2x) = Rp", lembur2x.toFixed(0));
  console.log("Total lembur (campuran) = Rp", totalLembur.toFixed(0));
}

=== SKENARIO > 3 JAM ===
Upah per jam = Rp 20231
Lembur 1-3 jam (1.5x) = Rp 91040
Lembur 4-5 jam (2x) = Rp 80925
```

Screenshot · 27/2/2026

BAGIAN B

Uji Pemahaman Kode

Prediksi output kode berikut *tanpa menjalankannya* terlebih dahulu, lalu klik "Periksa".

SOAL B-1 · OPERATOR & TIPE DATA

```
let a = 10;  
let b = "5";  
let c = a + Number(b);  
let d = a + b;  
console.log(c);           // Jawaban 1  
console.log(d);           // Jawaban 2  
console.log(typeof c);    // Jawaban 3
```

JAWABAN 1 — CONSOLE.LOG(C)

JAWABAN 2 — CONSOLE.LOG(D)

JAWABAN 3 — TYPEOF C

SOAL B-2 · CONTROL FLOW

```
let reject = 8;  
let total  = 200;  
let rate   = (reject / total) * 100;  
if (rate < 1) {  
  console.log("Excellent");  
} else if (rate < 3) {  
  console.log("Acceptable");  
} else if (rate < 5) {  
  console.log("Warning");  
} else {  
  console.log("Critical");  
}
```

NILAI RATE (%)

OUTPUT DI CONSOLE

SOAL B-3 · SWITCH & LOGIKA

```
let shift = 2;
let isWeekend = true;
let bonus = 0;
switch (shift) {
  case 3: bonus = 50000; break;
  case 2: bonus = 25000; break;
  default: bonus = 0;
}
if (isWeekend && shift === 2) {
  bonus = bonus * 2;
}
console.log(bonus);
```

OUTPUT — CONSOLE.LOG(BONUS)

50000

BAGIAN C

Refleksi Per Topik

Tuliskan refleksi jujur untuk setiap topik. Minimal 40 karakter per jawaban.

C-1

Jelaskan dengan kata-kata Anda sendiri: apa perbedaan let dan const? Berikan satu contoh nyata dari konteks industri untuk masing-masing.

Petunjuk: pikirkan data apa yang berubah vs data apa yang tetap dalam sistem produksi.

Perbedaan let dan const terletak pada perubahan nilainya. let digunakan untuk variabel yang nilainya bisa berubah selama program berjalan, sedangkan const digunakan untuk variabel yang nilainya tetap dan tidak boleh diubah. Dalam konteks industri, jumlah produksi harian bisa menggunakan let karena jumlahnya dapat berubah setiap hari. Sedangkan kode mesin lebih tepat menggunakan const karena bersifat tetap sebagai identitas mesin.

C-2

Mengapa menggunakan `===` lebih aman daripada `==`? Tuliskan contoh kode singkat yang menunjukkan perbedaan perilaku keduanya.

Petunjuk: coba bandingkan angka 0 dengan boolean false menggunakan keduanya di Console.

Operator `===` lebih aman dibandingkan `==` karena `===` membandingkan nilai sekaligus tipe data, sedangkan `==` hanya membandingkan nilai dan dapat melakukan konversi tipe otomatis. Contohnya:

```
console.log(0 == false); // true
console.log(0 === false); // false
```

Pada `==`, angka 0 dianggap sama dengan false karena terjadi konversi tipe data. Sedangkan `===` tidak mengubah tipe data sehingga hasilnya lebih akurat dan aman.

C-3

Dari seluruh materi suplemen, konsep mana yang paling sulit Anda pahami? Jelaskan apa yang membuat konsep tersebut sulit dan bagaimana Anda mencoba mengatasinya.

Konsep yang paling sulit saya pahami adalah perbedaan antara operator `==` dan `===`, terutama ketika membandingkan tipe data seperti angka dan boolean. Awalnya saya tidak memahami mengapa `0 == false` bisa menghasilkan nilai true. Hal tersebut cukup membingungkan karena secara logika keduanya terlihat berbeda. Untuk mengatasinya, saya mencoba menjalankan beberapa contoh kode di Console dan membaca kembali penjelasan tentang konversi tipe data otomatis pada operator `==`. Setelah mencoba langsung dan melihat hasilnya, saya menjadi lebih memahami perbedaannya.

TINGKAT KESULITAN MATERI (PILIH SATU)

 Mudah dipahami

 Butuh usaha

 Cukup menantang

 Sangat sulit

C-4

Dari latihan mandiri Bagian 8, pilih satu soal yang sudah Anda kerjakan. Tulis ulang kode solusi Anda dan jelaskan logika yang Anda gunakan.

Petunjuk: salin kode dari VS Code / Console Anda ke sini, lalu jelaskan baris-baris kuncinya.

```
function klasifikasiReject(totalProduksi, jumlahReject) {  
  const rejectRate = (jumlahReject / totalProduksi) * 100;  
  
  console.log("Reject Rate:", rejectRate.toFixed(2) + "%");  
  
  if (rejectRate < 1) {  
    console.log("Kategori: EXCELLENT");  
  } else if (rejectRate < 3) {  
    console.log("Kategori: ACCEPTABLE");  
  } else if (rejectRate < 5) {  
    console.log("Kategori: WARNING");  
  } else {  
    console.log("Kategori: CRITICAL");  
  }  
}  
  
klasifikasiReject(500, 20);
```

Penjelasan:

Pada kode tersebut, saya membuat sebuah function bernama `klasifikasiReject` yang menerima dua parameter, yaitu total produksi dan jumlah reject. Program menghitung persentase reject terlebih dahulu dengan rumus $(\text{jumlahReject} / \text{totalProduksi}) \times 100$. Setelah itu, digunakan percabangan if-else untuk menentukan kategori berdasarkan batas persentase. Urutan kondisi disusun secara bertahap agar setiap nilai masuk ke kategori yang sesuai. Function ini kemudian dipanggil dengan contoh data untuk melihat hasil klasifikasinya.

Bagian D — Refleksi Akhir & Rencana Belajar

Tulis secara jujur: apa yang paling berkesan dari suplemen ini, dan apa yang akan Anda lakukan sebelum Pertemuan 4 untuk memastikan diri Anda siap?

Hal yang paling berkesan dari suplemen ini adalah saya menjadi lebih memahami bagaimana JavaScript membaca dan mengeksekusi kode secara berurutan, terutama pada penggunaan if-else, switch, serta perbedaan antara `==` dan `===`. Awalnya saya mengira semua kondisi akan diperiksa, tetapi ternyata program berhenti pada kondisi pertama yang bernilai benar. Selain itu, saya juga baru menyadari pentingnya tipe data dalam menentukan hasil output.

Sebelum Pertemuan 4, saya akan mengulang kembali materi tentang control flow dan operator perbandingan, serta mencoba beberapa latihan tambahan di Console tanpa langsung melihat jawaban. Dengan cara tersebut, saya berharap bisa lebih terbiasa memprediksi output kode dan memahami logikanya dengan lebih baik.

Maessa Andrea Vallenia

23051430044

Diperiksa oleh Dosen Pengampu**Dr. Eng. Ir. Aji Ery Burhandenny, S.T., M.AIT.**

27 Februari 2026

Dokumen ini dicetak dari Logbook Digital Suplemen Praktikum — Aplikasi Web dan Mobile, Program Studi Teknik Industri, Universitas Negeri Yogyakarta