

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

Syalfa Oktifa Nanda

NIM

23051430023

KELAS / ROMBEL

TI-A 2023

TANGGAL Pengerjaan

27-Feb-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 sebaiknya pakai `const` karena dari awal memang tidak dirancang untuk berubah. Kode mesin itu seperti identitas unik, jadi kalau sampai berubah bisa bikin data jadi tidak konsisten. Kalau pakai `const` kita jadi lebih aman karena nilainya tidak bisa diganti sembarangan.

SCREENSHOT OUTPUT CONSOLE

```

> const kodeMesin = "CNC-02";
  let tahunPembuatan = 2024;
  let statusMesin = "stop";
  let kapasitasProduksi = 540;
  let presentasePengguna = 87;

console.log("===PROFIL MESIN===");
console.log("Kode Mesin:", kodeMesin);
console.log("Tahun Pembuatan:", tahunPembuatan);
console.log("Status Mesin:", statusMesin);
console.log("Kapasitas Produksi:", kapasitasProduksi, "unit");
console.log("Presentase Pengguna:", presentasePengguna + "%");

===PROFIL MESIN===
Kode Mesin: CNC-02
Tahun Pembuatan: 2024
Status Mesin: stop
Kapasitas Produksi: 540 unit
Presentase Pengguna: 87%
  
```

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` bernilai 0, maka terjadi pembagian dengan nol yang menyebabkan `biayaPerUnit` menjadi Infinity dan secara matematis tidak valid.

SCREENSHOT OUTPUT CONSOLE

```

> const hargaPerKg = 15000;
  const kebutuhanPerUnit = 0.35;
  const jumlahProduksi = 220;

const biayaPerUnit = hargaPerKg * kebutuhanPerUnit;
const totalBiayaMaterial = biayaPerUnit * jumlahProduksi;
const totalBeratMaterial = kebutuhanPerUnit * jumlahProduksi;

console.log("Biaya material per unit: Rp", biayaPerUnit);
console.log("Total biaya material hari ini: Rp", totalBiayaMaterial);
console.log("Total berat material:", totalBeratMaterial, "kg");

Biaya material per unit: Rp 5250
Total biaya material hari ini: Rp 1155000
Total berat material: 77 kg
  
```

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.

Urutan else if penting karena program mengecek dari atas ke bawah dan berhenti saat menemukan kondisi yang benar. Jika urutannya salah, hasil yang muncul bisa tidak sesuai logika yang diinginkan.

SCREENSHOT OUTPUT CONSOLE — UJI MINIMAL 2 SKENARIO

```
> console.log("=== SKENARIO 1 ===");
let skenario1 = [
  { totalProduksi: 1000, jumlahReject: 5 },
  { totalProduksi: 1000, jumlahReject: 20 }
];
for (let i = 0; i < skenario1.length; i++) {
  let total = skenario1[i].totalProduksi;
  let reject = skenario1[i].jumlahReject;
  let rate = (reject / total) * 100;

  let kategori, tindakan;

  if (rate < 1) {
    kategori = "Excellent";
    tindakan = "Tidak ada tindakan";
  }
  else if (rate >= 1 && rate <= 3) {
    kategori = "Acceptable";
    tindakan = "Monitor lebih ketat";
  }

  console.log("Reject Rate:", rate.toFixed(2) + "%");
  console.log("Kategori:", kategori);
  console.log("Tindakan:", tindakan);
  console.log("-----");
}

=== SKENARIO 1 ===
Reject Rate: 0.50%
Kategori: Excellent
Tindakan: Tidak ada tindakan
-----
Reject Rate: 2.00%
Kategori: Acceptable
Tindakan: Monitor lebih ketat
```

Screenshot · 27/2/2026

```
> console.log("=== SKENARIO 2 ===");
let skenario2 = [
  { totalProduksi: 1000, jumlahReject: 40 },
  { totalProduksi: 1000, jumlahReject: 80 }
];
for (let i = 0; i < skenario2.length; i++) {
  let total = skenario2[i].totalProduksi;
  let reject = skenario2[i].jumlahReject;
  let rate = (reject / total) * 100;

  let kategori, tindakan;

  if (rate > 3 && rate <= 5) {
    kategori = "Warning";
    tindakan = "Investigasi proses";
  }
  else if (rate > 5) {
    kategori = "Critical";
    tindakan = "Hentikan produksi, lakukan RCA";
  }

  console.log("Reject rate:", rate.toFixed(2) + "%");
  console.log("Kategori:", kategori);
  console.log("Tindakan:", tindakan);
  console.log("-----");
}

=== SKENARIO 2 ===
Reject rate: 4.00%
Kategori: Warning
Tindakan: Investigasi proses
-----
Reject rate: 8.00%
Kategori: Critical
Tindakan: Hentikan produksi, lakukan RCA
```

Screenshot · 27/2/2026



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.

Upah/jam = $3.500.000 \div 173$
= 20.231

Jam 1–3 = $3 \times (1,5 \times 20.231)$
= 3×30.346
= 91.038

Jam 4–5 = $2 \times (2 \times 20.231)$
= 2×40.462
= 80.924

Total = $91.038 + 80.924$
= $171.962 \approx 171.965$

SCREENSHOT OUTPUT CONSOLE — UJI MINIMAL 2 SKENARIO

```
> console.log("=== SKENARIO 1 ===");  
  
let gajiPokok = 350000;  
let upahPerJam = gajiPokok / 173;  
let N = 3;  
  
let totalLembur = N * (1.5 * upahPerJam);  
  
console.log("Upah per Jam:", upahPerJam.toFixed(2));  
console.log("Jam Lembur:", N);  
console.log("Total Upah Lembur: Rp", totalLembur.toFixed(0));  
  
=== SKENARIO 1 ===  
Upah per Jam: 2023.12  
Jam Lembur: 3  
Total Upah Lembur: Rp 9104
```

Screenshot · 27/2/2026

```
> console.log("=== SKENARIO 2 ===");  
  
let gajiPokok = 350000;  
let upahPerJam = gajiPokok / 173;  
let N = 6;  
  
let jamAwal = 3 * (1.5 * upahPerJam);  
let jamSisa = (N - 3) * (2 * upahPerJam);  
let totalLembur = jamAwal + jamSisa;  
  
console.log("Upah per Jam:", upahPerJam.toFixed(2));  
console.log("Jam Lembur:", N);  
console.log("Total Upah Lembur: Rp", totalLembur.toFixed(0));  
  
=== SKENARIO 2 ===  
Upah per Jam: 2023.12  
Jam Lembur: 6  
Total Upah Lembur: Rp 21243
```

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.

let digunakan untuk menyimpan data yang nilainya bisa berubah selama program berjalan. Sedangkan const digunakan untuk menyimpan data yang nilainya tetap dan tidak boleh diubah setelah dibuat.

Dalam konteks industri, misalnya pada sistem produksi, jumlah barang yang dihasilkan setiap jam bisa berubah tergantung kondisi mesin atau permintaan. Data seperti itu cocok menggunakan let karena nilainya dinamis. Sebaliknya, standar target produksi harian yang sudah ditetapkan perusahaan biasanya tidak berubah setiap saat. Nilai seperti ini lebih tepat menggunakan const agar tidak bisa diubah secara tidak sengaja di dalam program.

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 === mengecek nilai sekaligus tipe datanya. Sedangkan == hanya membandingkan nilai dan bisa mengubah tipe data secara otomatis agar terlihat sama.

Contohnya, angka 0 jika dibandingkan dengan false menggunakan == akan dianggap sama, padahal sebenarnya tipe datanya berbeda. Jika menggunakan ===, hasilnya tidak sama karena tipe data number dan boolean memang berbeda.

Karena itu, dalam penulisan kode sebaiknya menggunakan === supaya tidak terjadi kesalahan logika akibat perbedaan tipe data yang tidak terlihat.

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 control flow, terutama if...else if dan penggunaan && serta ||. Saya sempat bingung karena urutan kondisi sangat berpengaruh dan program akan berhenti di kondisi pertama yang bernilai true.

Untuk mengatasinya, saya mencoba menuliskan logikanya dalam bentuk kalimat dulu lalu menguji tiap kondisi di console dengan berbagai nilai sampai benar-benar paham polanya.

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.

```
console.log("=== SKENARIO 2 ===");
```

```
let gajiPokok = 350000;
```

```
let upahPerJam = gajiPokok / 173;
```

```
let N = 6;
```

```
let jamAwal = 3 * (1.5 * upahPerJam);
```

```
let jamSisa = (N - 3) * (2 * upahPerJam);
```

```
let totalLembur = jamAwal + jamSisa;
```

```
console.log("Upah per Jam:", upahPerJam.toFixed(2));
```

```
console.log("Jam Lembur:", N);
```

```
console.log("Total Upah Lembur: Rp", totalLembur.toFixed(0));
```

Saya menggunakan logika percabangan (if-else) untuk menghitung lembur sesuai aturan tarifnya. Pertama, program mengecek jumlah jam lembur (N). Jika lembur sampai 3 jam atau kurang, maka semua jam langsung dihitung 1,5 kali upah per jam. Namun jika lembur lebih dari 3 jam, maka perhitungannya saya bagi menjadi dua bagian: tiga jam pertama tetap dihitung 1,5x, sedangkan sisa jam berikutnya dihitung 2x. Setelah itu, kedua hasil tersebut dijumlahkan untuk mendapatkan total upah lembur.

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?

Yang berkesan dari suplemen ini adalah penjelasannya yang detail namun tetap mudah diikuti, terutama bagian kesalahan umum pemula. Bagian itu membuat saya sadar bahwa kesalahan kecil seperti salah menggunakan `=` dan `===` bisa berdampak besar pada hasil program. Sebelum Pertemuan 4, saya akan membaca ulang bagian yang belum saya pahami dan mencoba kembali beberapa contoh di console agar lebih terbiasa dengan sintaksnya.

Syalfa Oktifa Nanda
23051430023

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