SMK N2
KATINGAN

# Belajar Coding Dasar

# Tujuan Pembelajaran

1. Memahami konsep percabangan **Switch-Case** dalam JavaScript.
2. Mengetahui **perbedaan Switch-Case dan If-Else** serta kapan harus menggunakannya.
3. Memahami konsep **perulangan (looping)** dalam JavaScript.
4. Mampu menggunakan **for loop** dan **while loop** untuk mengulang kode secara otomatis.
5. Memahami **fungsi (function)** dalam JavaScript untuk membuat kode lebih rapi, efisien, dan mudah digunakan kembali.

# Switch-Case

# Apa Itu Switch-case

Bayangkan kamu sedang main game, dan kamu harus memilih karakter

1. Zilong

2. Miya

3. Franco

- Contoh Sederhana:
  - Jika tombol 1 ditekan, maka Zilong Terpilih

Nah, switch-case adalah "mesin" yang akan memilih sesuai angka yang kamu masukkan. Jadi kamu tidak perlu menulis "kalau ini, maka itu" berulang-ulang.

# Bentuk Umumnya:

SMK N2 KATINGAN

```
switch(nilai) {
case 1:
// aksi jika nilai adalah 1
break;
case 2:
// aksi jika nilai adalah 2
break;
default:
// aksi jika nilai tidak cocok dengan yang ada di atas
}
```

**break:**

- Bayangkan kamu lagi main game tanya-jawab. Setiap pertanyaan punya beberapa pilihan jawaban. Begitu kamu pilih jawaban yang benar, pertanyaan berhenti dan kamu lanjut ke soal berikutnya.

Nah, dalam pemrograman, break bekerja seperti itu. Kalau komputer sudah menemukan jawaban yang cocok, break memberitahu komputer untuk berhenti mengecek pilihan lain dan keluar dari bagian itu.

# Bentuk Umumnya:

SMK N2 KATINGAN

```
switch(nilai) {
case 1:
// aksi jika nilai adalah 1
break;
case 2:
// aksi jika nilai adalah 2
break;
default:
// aksi jika nilai tidak cocok dengan yang ada di atas
}
```

**default**:
Kata kunci default digunakan kalau tidak ada jawaban yang cocok.
Ibaratnya, kamu lagi nyari warna di daftar:

- Kalau warnanya merah, lakukan A
- Kalau biru, lakukan B
- Tapi kalau bukan dua-duanya, lakukan yang default → seperti pilihan "lainnya"

# Contoh Penggunaan

```
let jawaban = 1;

  switch (jawaban) {
    case 1:
      console.log("Kamu memilih Zilong!");
      break;
    case 2:
      console.log("Kamu memilih Miya!");
      break;
    case 3:
      console.log("Kamu memilih Franco!");
      break;
    default:
      console.log("Pilihan tidak tersedia!");
  }
```

# Kenapa pakai Switch-Case?

1. Jika menggunakan if-else bertingkat, kode akan menjadi panjang dan kurang rapi.
2. Dengan **switch-case**, setiap kondisi lebih **terstruktur dan mudah dibaca**.

```
if (pilihan == 1) {
  console.log("Zilong");
  }
else if (pilihan == 2) {
  console.log("Miya");
  }
else if (pilihan == 3) {
  console.log("Franco");
  }
```

```
switch (pilihan) {

  case 1:
   console.log("zilong");
   break;
  case 2:
   console.log("Miya™);
   break;
  case 3:
   console.log("Franco™);
   break;
  }
```

# Switch-Case dengan Banyak Nilai Sama

Kadang ada kondisi yang menghasilkan output sama. Kita bisa menggabungkan beberapa case.

Contohnya "nasi goreng" dan "mie goreng" — dua-duanya harganya 10 ribu. Kita tidak perlu menulis dua kali.

```
let makanan = "mie goreng";

switch (makanan) {
case "nasi goreng":
case "mie goreng":
  console.log("Harganya 10 ribu");
  break;
case "ayam bakar":
  console.log("Harganya 15 ribu");
  break;
default:
  console.log("Maaf, tidak tersedia");
}
```

# Switch-Case dengan Double Condition

Kadang ada kondisi yang memerlukan dua persyaratan agar bisa terpenuhi. Misalnya, nilai absen minimal 80 dan nilai ujian minimal 75 — keduanya harus dipenuhi. Kita perlu menuliskan dua syarat sekaligus dalam kode agar kondisi ini bisa berjalan sesuai aturan.

```
let nilai = 80;
let absen = 85;

switch (true) {
  case (nilai >= 75 && absen >= 80):
    console.log("Lulus");
    break;
  case (nilai >= 75 && absen < 80):
    console.log("Tidak lulus karena absen kurang");
    break;
  case (nilai < 75 && absen >= 80):
    console.log("Tidak lulus karena nilai kurang");
    break;
  default:
    console.log("Tidak lulus karena nilai dan absen kurang");
}
```

# TUGAS
# SWITCH-CASE

# 1 Menampilkan Nama Hari Berdasarkan Angka

```
let hari = 1;

switch (hari) {
    case 1:
        console.log("Hari ?");
        break;
    case 2:
        console.log("Hari ?");
        break;
    case 3:
        console.log("Hari ?");
        break;
    case 4:
        console.log("Hari ?");
        break;
    case 5:
        console.log("Hari ?");
        break;
    case 6:
        console.log("Hari ?");
        break;
    case 7:
        console.log("Hari ?");
        break;
    default:
        console.log("Maaf ?");
}
```

**Selesaikan kodingan dengan tanda ?**
**berdasarkan data berikut**

ANGKA 1 = SENIN , ANGKA 2 = SELASA
ANGKA 3 = RABU , ANGKA 4 = KAMIS
ANGKA 5 = JUMAT ,ANGKA 6 =  SABTU
ANGKA 7 = MINGGU

Default value = “Maaf Masukan Hari yang benar”

# 2. Menentukan Musim Berdasarkan Bulan

```
let bulan = "februari"

switch (bulan) {
    case 'desember':
    case 'januari':
    case '?':
    console.log("Musim ?");
    break;
    case 'maret':
    case 'april':
    case 'mei':
      console.log("Musim ?");
      break;
    case 'juni':
    case 'juli':
    case 'agustus':
      console.log("Musim ?");
      break;
    case 'september':
    case 'oktober':
    case '?':
      console.log("Musim ?");
      break;
    default:
      console.log("Nama bulan tidak valid! Masukkan nama bulan
          yang benar.");
}
```

# Selesaikan kodingan dengan tanda ?
# berdasarkan data berikut

Menentukan Musim Berdasarkan Bulan

- Jika bulan Desember - Februari, tampilkan "Musim Hujan".
- Jika bulan Maret - Mei, tampilkan "Musim Semi".
- Jika bulan Juni - Agustus, tampilkan "Musim Panas".
- Jika bulan September - November, tampilkan "Musim Gugur".

Default value = “Nama bulan tidak valid! Masukkan nama bulan yang benar“

# Contoh hasil outputnya

Output

```
Hari Minggu
Musim Semi
```

# For & While

# Apa itu Perulangan for?

Dalam kehidupan sehari-hari, kadang kita melakukan sesuatu berulang-ulang. Misalnya:

- Menyapu lantai 3 kali.
- Push-up 10 kali.
- Mengucap "Selamat pagi!" ke semua teman sekelas.

Nah, di dunia pemrograman, kita juga bisa menyuruh komputer melakukan sesuatu berulang-ulang. Ini disebut perulangan, dan salah satu jenisnya adalah perulangan for.

# Kenapa Harus Perulangan?

```
console.log("Aku harus belajar!");
console.log("Aku harus belajar!");
console.log("Aku harus belajar!");
console.log("Aku harus belajar!");
console.log("Aku harus belajar!");
```

```
for (let i = 1; i <= 5; i++) {
    console.log("Aku Harus belajar");
}
```

Kalau kamu ketik satu-satu, lama banget!
Tapi dengan perulangan, kamu hanya perlu suruh komputer:
“Tolong tulis kalimat itu 5 kali!”
Maka komputer akan melakukannya dengan cepat dan benar.

# Bentuk Umumnya:

SMK N2 KATINGAN

```
for (awal; kondisi; perubahan) {
// yang mau diulang
}
```

**awal**
Contoh: let i = 1
Ini artinya mulai menghitung dari angka 1. Biasanya kita pakai variabel bernama i untuk hitung.

**kondisi**
Contoh: i <= 5
Ini artinya perulangan akan terus berjalan selama i masih kurang dari atau sama dengan 5.

**perubahan**
Contoh: i++
Ini artinya setiap kali selesai satu putaran, angka i akan bertambah 1.

# Contoh Penggunaan

```
for (let i = 1; i <= 3; i++) {
    console.log("Aku Harus belajar");
}
```

Kita mau mengulang tulisan "Aku harus belajar!" sebanyak 3 kali

**let i = 1; i<= 3;**
Mulai dari angka 1 (i = 1), selama angkanya belum lewat dari 3 (i <= 3),
**i++**
terus ulangi, dan setiap kali ulangi tambahkan 1 ke angka itu (i++)

**console.log("Aku harus belajar!");**
Tampilkan tulisan ke layar

# Modifikasi Perubahan

Biasanya **i++** artinya **tambah 1.**
Tapi kita bisa juga ubah jadi:

- **i += 2** → **tambah 2** setiap kali (jadi 2, 4, 6, 8, ...)
- **i--** → **kurang 1** setiap kali (jadi 5, 4, 3, ...)

## Contoh i++

```
1 for (let i = 1; i <= 5; i++) {
2   console.log(i);
3 }
```

## Contoh i+=2

```
1 for (let i = 2; i <= 10; i += 2) {
2   console.log(i);
3 }
```

## Contoh i--

```
1 for (let i = 5; i >= 1; i--) {
2   console.log(i);
3 }
```

# Menggunakan Nilai i di Dalam Loop (menggabungkan dengan teks)

```
1 for (let i = 1; i <= 5; i++) {
2     console.log("Aku belajar For , Perulangan ke-" + i);
3 }
```

console.log("Aku belajar For , Perulangan ke-" **+ i**);

**+ i** artinya menggabungkan (menyambung) angka i ke dalam kalimat.

- Jadi setiap kali perulangan berjalan, nilai i akan berubah otomatis (1, 2, 3, dst).
- Dengan begitu, kalimat yang dicetak akan berbeda angkanya sesuai hitungan perulangan.

# Kesalahan Umum Saat Membuat Perulangan for

### 1. Loop Tak Berhenti (Infinite Loop)

Apa itu?

Loop tak berhenti artinya perulangan terus berjalan tanpa henti, sehingga program tidak bisa selesai dan jadi macet.

```
1 for (let i = 1; i <= 5; // lupa i++ //) {
2     console.log(i);
3 }
```

## 2. Salah Tulis Kondisi

Apa itu?
Kalau kondisi di perulangan salah tulis, hasilnya bisa kurang tepat.
Misal Kita ingin melakukan perulangan hingga 5 tetapi menulis seperti berikut , maka hasilnya akan salah,

```
1 for (let i = 1; i < 5; i++) {
2     console.log(i);
3 }
```

**Dikarenakan i < 5 bukan i <=5** sehingga hanya berhenti di angka 4

# Tugas

# Perulangan For

**1.** buat kalimat “Aku semangat!” 4 kali. Ganti tanda ? dengan angka yang sesuai

```
for (let i = 1; i <= ?; i++) {
    console.log("Aku Semangat");
}
```

**2** .buat angka dari 10 sampai 1.
Ganti tanda ?? dengan tanda yang benar

```
6 for (let j = 10; j >= 1; j??) {
7   console.log(j);
8   }
```

3. buat kalimat aku belajar dengan nomor urut 1 sampai 10.

```
12 for (let k = 1; k <= 10; k++) {
13  console.log(? + ". Aku Belajar ");
14  }
```

# Output Tugas Seharusnya

```
Aku Semangat
Aku Semangat
Aku Semangat
Aku Semangat
10
9
8
7
6
5
4
3
2
1
1. Aku Belajar
2. Aku Belajar
3. Aku Belajar
4. Aku Belajar
5. Aku Belajar
6. Aku Belajar
7. Aku Belajar
8. Aku Belajar
9. Aku Belajar
10. Aku Belajar
```

# Apa itu Perulangan While?

Perulangan while adalah cara menyuruh komputer mengulang sesuatu **selama sebuah kondisi masih benar.**

Bayangkan kamu menyapu lantai selama lantainya masih kotor. Jadi, selama kondisi "lantai masih kotor" itu benar, kamu akan terus Menyapu

Begitu juga dengan while. Komputer akan terus mengulang selama kondisi masih **terpenuhi.**

# Bentuk Umumnya:

SMK N2 KATINGAN

```
while (kondisi) {
  // isi perulangan (apa yang mau diulang)
}
```

**Kondisi**

- Ini adalah syarat atau aturan kapan perulangan boleh jalan.
- Selama syarat ini benar (true), isi di dalam kurung { } akan terus berulang

**contoh Kondisi**

- i <= 5 → artinya selama i kurang dari atau sama dengan 5
- i !== 10 → artinya selama i tidak sama dengan 10

# Contoh Penggunaan

```
1  let i = 1;
2
3  while (i <= 3) {
4    console.log("Saya Belajar While");
5    i++;
6  }
```

**while** (**i <= 5**)
Selama **i** masih kurang dari atau sama dengan **5**, ulangi terus.

{ **console.log**(**...**) }
Ini isi perulangannya, yaitu mencetak tulisan ke layar.

**i++**
Tambah angka i setiap kali selesai satu putaran.

# Perbedaan While dan For

# Kapan Digunakan?

**for**

Kalau sudah tahu mau mengulang berapa kali

**while**

Kalau belum tahu pasti mau mengulang berapa kali

## Misal

**for itu seperti timer:**

Sudah tahu waktunya, langsung mulai dan berhenti sesuai rencana.

**while itu seperti alarm sensor:**

Akan terus jalan selama kondisi masih terjadi. Tidak tahu pasti kapan berhenti.

# Kesalahan Umum Saat Membuat Perulangan While

## 1. Loop Tak Berhenti (Infinite Loop)

Apa itu?

Loop tak berhenti artinya perulangan terus berjalan tanpa henti, sehingga program tidak bisa selesai dan jadi macet.

```
let i = 1;

while (i <= 5) {
  console.log(i);
  // lupa i++ → program jalan terus, tidak berhenti (infinite
      loop)
}
```

## 2. Salah Tulis Kondisi

Apa itu?

Kalau kondisi di perulangan salah tulis, hasilnya bisa kurang tepat.

Misal Kita ingin melakukan perulangan hingga 5 tetapi menulis seperti berikut , maka hasilnya akan salah

```
  // salah harusnya <= 5
let i = 1;

while (i < 5) {
  console.log(i);
  i++;
}
```

# LATIHAN

## Perulangan While

**1.** buat angka dari 1 sampai 10, Selesaikan kodingan di bawah , ubah tanda ? menjadi angka yang benar

```
1  let i = 1;
2
3  while (i <= ?) {
4     console.log(i);
5     i++;
6  }
```

**2.** buat kalimat “saya belajar while” sebanyak 3, Selesaikan kodingan di bawah , ubah tanda ? menjadi angka yang benar

```
let j = 1;

while (j <= 3) {
    console.log("?");
    j++;
}
```

**3.** buat Kalimat "Belajar itu seru" dengan nomor urut genap 2–10, Selesaikan kodingan di bawah , pastikan tanda ? terisi dengan benar

```
let k = 2;
while (k <= 10) {
  console.log(? + ". Belajar Itu Seru"); // variabel ?
  k+=?; // angka ?
}
```

# Output Tugas Seharusnya

```
1
2
3
4
5
6
7
8
9
10
Saya Belajar While
Saya Belajar While
Saya Belajar While
2. Belajar Itu Seru
4. Belajar Itu Seru
6. Belajar Itu Seru
8. Belajar Itu Seru
10. Belajar Itu Seru
```

# Fungsi
# (Function)

# Apa itu Fungsi (Function)

Function adalah sekumpulan perintah yang diberi nama, agar bisa digunakan berulang kali.

Misalnya kamu sering membuat kopi. Daripada menjelaskan langkahnya berulang-ulang, kamu cukup bilang:
**“Saya bikin kopi.”**

Nah, **bikin kopi itu ibarat function**. Semua langkah-langkahnya disimpan di dalamnya, tinggal panggil kalau dibutuhkan.

# Kenapa Pakai Function?

SMK N2 KATINGAN

1. **Menghemat penulisan**
2. **Kode jadi rapi dan teratur**
3. **Bisa dipakai berulang-ulang**
4. **Lebih mudah dibaca**

SMK N2
KATINGAN

# perbandingan menggunakan function dan tidak menggunakan function

```
1  console.log("Halo, Budi!");
2  console.log("Halo, Sinta!");
3  console.log("Halo, Andi!");
```

```
1  function sapa(nama) {
2    console.log("Halo, " + nama + "!");
3  }
4
5  sapa("Budi");
6  sapa("Sinta");
7  sapa("Andi");
```

SMK N2 KATINGAN

# Bentuk Umum Tanpa Parameter:

```
function namaFunction() {
  // isi perintah

}

namaFunction();
```

**Function**
Menandakan kita sedang membuat function

**namaFunction**
Ini adalah nama function-nya. Kamu boleh menamai apa saja (asal tidak pakai spasi dan bukan nama yang sudah dipakai).
Contoh nama yang boleh:

- salam
- tampilkanPesan
- cetakNama

SMK N2 KATINGAN

# Bentuk Umum Tanpa Parameter:

```
function namaFunction() {
// isi perintah

}

namaFunction();
```

**Tanda Kurung kurawal** {}

Tanda kurung kurawal {} menandakan isi function, yaitu perintah-perintah yang akan dijalankan saat function dipanggil.

**dan namaFunction**() diakhir menandakan pemanggilan dari function itu sendiri

# Contoh Penggunaan

```
1 function salam() {
2   console.log("Halo, selamat datang di kelas Gen-IT!");
3 }
4
5 salam();
```

**function salam**()
→ kita buat function bernama salam

Di dalamnya ada **console.log**(**...**) → ini adalah isi perintah
Tapi function ini belum berjalan kalau belum dipanggil

Sehingga Kita memerlukan pemanggilan function yang dimanaa adalah **salam**();

SMK N2
KATINGAN

# Bentuk Umum Dengan Paramater:

```
1 function namaFunction(parameter) {
2
3  // isi perintah yang memakai parameter
4  }
5  namaFunction();
6
```

Bentuk Umum dengan parameter dan tanpa parameter hanya dibedakan dari

(**parameter**) yang merupakan
“Kotak masuk” yang dapat disi data dari luar
(contohnya nama, angka, dll)

# Contoh Penggunaan

```
1 function sapa(nama) {
2  console.log("Halo " + nama + " ,selamat datang di kelas Gen
       -IT!");
3 }
4
5 sapa("Rina");
6 sapa("Budi");
7 sapa("Tono");
```

**function Sapa**
kita buat function bernama **Sapa**

(**nama**) adalah parameter — seperti tempat kosong untuk diisi nama siapa pun

**console.log**("**Halo,** " **+ nama +** "**!**");
adalah isi perintah: menyapa dengan nama itu

Kita mengguanakan **Sapa**(**“Nama”**) untuk memanggil fungsinya

# Tugas Function Dengan Parameter

**1.** Buat fungsi tampilkanData(nama, kelas) yang menampilkan:
Nama: Nama kalian
Kelas: kelas kalian

```
function tampilkanData(nama, kelas) {
  console.log("Nama: " + nama);
  console.log("Kelas: " + kelas);
}

tampilkanData("Budi", "XII RPL"); // ubah dengan nama dan kelas masing masing
```

**2.**Buat fungsi nilaiSiswa(nama, nilai) yang menampilkan:

nama kalian , mendapat nilai 69

```
function nilaiSiswa(nama, nilai) {
    console.log(nama + " mendapat nilai " + nilai);
}

nilaiSiswa("Budi", 85);
```

# Contoh Output tugas

```
Nama: Budi
Kelas: XII RPL
Budi mendapat nilai 85

=== Code Execution Successful ===
```

SMK N2 KATINGAN

# Bentuk Umum Dengan Default Paramater:

```
function namaFunction(parameter=nilaiDefault) {
// isi perintah yang memakai parameter
}

namaFunction(nilai);
namaFunction();
```

**Default Parameter itu apa?**

Kadang, kita ingin membuat parameter di function yang jika tidak diisi, tetap punya nilai standar (default). Jadi, kalau kita lupa memberikan nilai, function tetap bisa jalan dengan nilai yang sudah kita tentukan sebelumnya.

# Contoh Penggunaan

```
1  function sapa(nama = "Teman") {
2    console.log("Halo, " + nama + "!");
3  }
4
5  sapa("Ani");   // Output: Halo, Ani!
6  sapa();        // Output: Halo, Teman!
```

**Penjelasannya:**

- Kalau kamu panggil **sapa**("**Ani**"), parameter nama berisi "Ani", jadi muncul "**Halo, Ani!**"
- Kalau kamu panggil **sapa**() **tanpa isi nama,** karena ada default "Teman", maka yang muncul adalah "**Halo, Teman!**"

SMK N2
KATINGAN

# Fungsi yang Mengembalikan Nilai (Return)

**Apa itu Fungsi yang Mengembalikan Nilai (Return)?**

Fungsi yang mengembalikan nilai adalah fungsi yang menghasilkan sebuah jawaban, dan kamu bisa menyimpan atau menggunakan jawaban itu.

Bayangkan kamu tanya ke kalkulator:
"Kalau 5 + 3, hasilnya berapa?"
Kalkulator akan kasih kamu jawaban: 8
Nah, fungsi dengan return itu seperti kalkulator:

- Kamu beri input (misalnya angka 5 dan 3)
- Fungsi mengolah
- Lalu mengembalikan hasilnya

# Bentuk Dasar Fungsi dengan Return:

```
function namaFungsi() {
return nilai; // nilai dikembalikan ke pemanggil fungsi
}
```

Sama Seperti Bentuk Fungsi Pada umumnya tetapi Terdapat **return nilai** yang dimana dapat disimpan dan dikembalikan ke pemanggil fungsi

# Contoh Fungsi Return Dengan Parameter

```javascript
function luasPersegi(sisi) {
    return sisi * sisi;
}

let luas = luasPersegi(4);
console.log("Luas: " + luas); // Output: Luas: 16
```

**Apa yang terjadi:**

- Buat fungsi bernama luasPersegi yang terima satu angka (sisi).
- Fungsi ini hitung sisi x sisi (luas persegi) dan beri hasilnya kembali.
- Kita panggil fungsi dengan angka 4, berarti hitung 4 x 4 = 16.
- Hasil 16 disimpan di luas.
- Tampilkan kata "Luas: 16" di layar.

# Tugas
# Function Return

**1.** Buat fungsi konversiJam(jam) yang mengembalikan total menit.

**Rumus: Jam x 60**

Contoh: 2 jam = 120 menit, ubah tanda ? menjadi nilai yang benar

```
function konversiJam(jam) {
  return jam * ?;
}

let totalMenit = konversiJam(2);
console.log("Total menit: " + totalMenit);
```