Perulangan (Loop) di C++

Slide 1

- **Contoh Masalah Tanpa Perulangan:**

```
// Mencetak angka 1-10 tanpa loop
cout << "1" << endl;
cout << "2" << endl;
cout << "3" << endl;
cout << "4" << endl;
cout << "5" << endl;
cout << "6" << endl;
cout << "7" << endl;
cout << "8" << endl;
cout << "9" << endl;
cout << "10" << endl;
```

- **Solusi dengan Perulangan:**

```
// Mencetak angka 1-10 dengan loop
for (int i = 1; i <= 10; i++) {
    cout << i << endl;
}
```

Jenis-jenis Perulangan di C++

Slide 2

- **3 Jenis Perulangan Utama:**
  - **for loop**: Untuk pengulangan dengan jumlah yang sudah diketahui
  - **while loop**: Untuk pengulangan berdasarkan kondisi
  - **do-while loop**: Minimal menjalankan 1x, baru cek kondisi
- **For Loop - Struktur Dasar:**

```
for (inisialisasi; kondisi; increment/decrement) {
    // blok kode yang diulang
}

// Contoh:
for (int i = 0; i < 5; i++) {
    cout << "Perulangan ke-" << i << endl;
}
```

- **While Loop - Struktur Dasar:**

```
while (kondisi) {
    // blok kode yang diulang
    // jangan lupa update variabel kondisi!
}

// Contoh:
int i = 0;
while (i < 5) {
    cout << "Perulangan ke-" << i << endl;
    i++; // PENTING: increment agar tidak infinite loop
}
```

- **Do-While Loop - Struktur Dasar:**

```
do {
    // blok kode yang diulang (minimal 1x)
} while (kondisi);

// Contoh:
int pilihan;
do {
    cout << "Masukkan pilihan (1-3): ";
    cin >> pilihan;
} while (pilihan < 1 || pilihan > 3);
```

Kapan Menggunakan Jenis Perulangan yang Tepat?

Slide 3

- **Gunakan FOR LOOP ketika:**
  - Jumlah pengulangan sudah diketahui pasti
  - Menggunakan counter/index yang berurutan
  - Memproses array atau string
  - Membuat pola dengan ukuran tertentu

```
// Contoh yang tepat untuk FOR:
// 1. Mencetak angka 1-100
for (int i = 1; i <= 100; i++) {
    cout << i << " ";
}
```

- **Gunakan WHILE LOOP ketika:**
  - Kondisi berhenti tidak pasti/tergantung input

- Menunggu event tertentu
- Validasi input
- Membaca file sampai habis

```
// Contoh yang tepat untuk WHILE:
// 1. Validasi input
int umur;
cout << "Masukkan umur (0-150): ";
cin >> umur;
while (umur < 0 || umur > 150) {
    cout << "Umur tidak valid! Masukkan lagi: ";
    cin >> umur;
}

// 2. Menu program
char lanjut = 'y';
while (lanjut == 'y' || lanjut == 'Y') {
    // tampilkan menu
    cout << "Lanjut? (y/n): ";
    cin >> lanjut;
}
```

- **Gunakan DO-WHILE LOOP ketika:**

- Harus menjalankan minimal 1 kali
- Menu program yang harus tampil dulu
- Input validation yang perlu dicoba sekali

```
// Contoh yang tepat untuk DO-WHILE:
int password;
do {
    cout << "Masukkan password: ";
    cin >> password;
} while (password != 12345);
```

For Loop - Syntax dan Struktur Detail

Slide 4

- **Syntax For Loop:**

```
for (inisialisasi; kondisi; update) {
    // statement yang diulang
}
```

- **Penjelasan Setiap Bagian:**
  - **Inisialisasi**: Dijalankan sekali di awal (biasanya deklarasi variabel counter)
  - **Kondisi**: Dicek sebelum setiap iterasi (jika false, loop berhenti)
  - **Update**: Dijalankan setelah setiap iterasi (biasanya increment/decrement)

- **Contoh Step-by-Step Eksekusi:**

```
for (int i = 1; i <= 3; i++) {
    cout << "Nilai i: " << i << endl;
```

```
}

// Eksekusi:
// Step 1: int i = 1        (inisialisasi)
// Step 2: i <= 3? (1 <= 3) true ? jalankan cout
// Step 3: i++              (i menjadi 2)
// Step 4: i <= 3? (2 <= 3) true ? jalankan cout
// Step 5: i++              (i menjadi 3)
// Step 6: i <= 3? (3 <= 3) true ? jalankan cout
// Step 7: i++              (i menjadi 4)
// Step 8: i <= 3? (4 <= 3) false ? keluar dari loop
```

- **Variasi For Loop:**

```
// 1. Countdown
for (int i = 10; i >= 1; i--) {
    cout << i << " ";
}

// 2. Step berbeda
for (int i = 0; i <= 20; i += 2) {
    cout << i << " ";  // 0 2 4 6 8 10 12 14 16 18 20
}

// 3. Multiple variables
for (int i = 0, j = 10; i < j; i++, j--) {
    cout << "i=" << i << " j=" << j << endl;
}
```

For Loop dengan Counter dan Step

Slide 5

- **Mengatur Counter dan Step:**

```
// Basic increment (+1)
for (int i = 1; i <= 5; i++) {
    cout << i << " ";  // Output: 1 2 3 4 5
}

// Basic decrement (-1)
for (int i = 5; i >= 1; i--) {
    cout << i << " ";  // Output: 5 4 3 2 1
}

// Custom step (+2)
for (int i = 0; i <= 10; i += 2) {
    cout << i << " ";  // Output: 0 2 4 6 8 10
}

// Custom step (-3)
for (int i = 15; i >= 0; i -= 3) {
    cout << i << " ";  // Output: 15 12 9 6 3 0
}
```

- **Aplikasi Praktis Counter:**

```
// 1. Membuat daftar nomor
cout << "Daftar Siswa:" << endl;
for (int i = 1; i <= 5; i++) {
    cout << i << ". Siswa ke-" << i << endl;
}

// 2. Tabel perkalian
int angka = 7;
cout << "Tabel Perkalian " << angka << ":" << endl;
for (int i = 1; i <= 10; i++) {
    cout << angka << " x " << i << " = " << (angka * i) << endl;
}

// 3. Menghitung jumlah bilangan genap
int jumlah = 0;
for (int i = 2; i <= 20; i += 2) {
    jumlah += i;
    cout << "Menambah " << i << ", jumlah sekarang: " << jumlah << endl;
}
```

Nested For Loop (Perulangan Bersarang)

Slide 6

- **Konsep Nested Loop:**
  - Loop di dalam loop
  - Loop luar mengontrol baris, loop dalam mengontrol kolom
  - Sangat berguna untuk membuat pola 2D
- **Struktur Dasar:**

```
for (int i = 0; i < baris; i++) {        // Loop luar (baris)
    for (int j = 0; j < kolom; j++) {    // Loop dalam (kolom)
        // kode yang dieksekusi setiap kombinasi i,j
    }
}
```

- **Contoh Sederhana - Tabel Perkalian:**

```
cout << "Tabel Perkalian 3x3:" << endl;
for (int i = 1; i <= 3; i++) {          // Baris 1,2,3
    for (int j = 1; j <= 3; j++) {      // Kolom 1,2,3
        cout << (i * j) << "\t";        // \t = tab
    }
    cout << endl;                        // Pindah baris setelah selesai 1 baris
}
// Output:
// 1    2    3
// 2    4    6
// 3    6    9
```

- **Contoh Pattern - Segitiga Bintang:**

```
cout << "Segitiga Bintang:" << endl;
for (int i = 1; i <= 5; i++) {          // 5 baris
    for (int j = 1; j <= i; j++) {      // Bintang sebanyak nomor baris
        cout << "* ";
    }
    cout << endl;
}
// Output:
// *
// * *
// * * *
// * * * *
```

```
// * * * * *
```

- **Contoh Pattern - Persegi Angka:**

```
cout << "Persegi Angka 4x4:" << endl;
for (int i = 1; i <= 4; i++) {
    for (int j = 1; j <= 4; j++) {
        cout << j << " ";
    }
    cout << endl;
}
// Output:
// 1 2 3 4
// 1 2 3 4
// 1 2 3 4
// 1 2 3 4
```

While Loop - Syntax dan Struktur

Slide 7

- **Struktur While Loop:**

```
while (kondisi) {
    // blok kode yang diulang
    // PENTING: pastikan ada perubahan yang membuat kondisi menjadi false
}
```

- **Karakteristik While Loop:**
  - Dicek kondisi dulu, baru eksekusi (bisa jadi tidak pernah dieksekusi)
  - Cocok ketika jumlah iterasi tidak diketahui pasti
  - Harus ada mekanisme untuk mengubah kondisi
- **Contoh Dasar While Loop:**

```
// Countdown sederhana
int countdown = 5;
cout << "Countdown dimulai:" << endl;
while (countdown > 0) {
    cout << countdown << "..." << endl;
    countdown--;  // PENTING: decrement agar tidak infinite loop
}
cout << "Selesai!" << endl;
```

- **While Loop vs For Loop - Contoh Sama:**

```
// Menggunakan for loop
cout << "For loop:" << endl;
for (int i = 1; i <= 3; i++) {
    cout << "Iterasi " << i << endl;
}

// Menggunakan while loop
cout << "While loop:" << endl;
int i = 1;
while (i <= 3) {
    cout << "Iterasi " << i << endl;
    i++;  // Jangan lupa increment!
}
```

While Loop vs For Loop - Kapan Pakai Yang Mana?

Slide 8

- **Gunakan WHILE LOOP untuk:**
  - Input validation yang tidak tahu berapa kali salah
  - Menu program yang bisa dipilih berkali-kali
  - Membaca data sampai kondisi tertentu
  - Event-driven programming
- **Contoh Kasus While Loop Yang Tepat:**

```
// 1. Input validation - tidak tahu berapa kali user salah input
int password;
cout << "Masukkan password (4 digit): ";
cin >> password;
while (password < 1000 || password > 9999) {
    cout << "Password harus 4 digit! Coba lagi: ";
    cin >> password;
}
```

```
cout << "Password benar!" << endl;

// 2. Menu program yang berulang
char pilihan;
while (true) {  // Infinite loop dengan break
    cout << "\n=== MENU UTAMA ===" << endl;
    cout << "1. Tambah Data" << endl;
    cout << "2. Lihat Data" << endl;
    cout << "3. Keluar" << endl;
    cout << "Pilihan Anda: ";
    cin >> pilihan;

    if (pilihan == '1') {
        cout << "Menambah data..." << endl;
    } else if (pilihan == '2') {
        cout << "Menampilkan data..." << endl;
    } else if (pilihan == '3') {
        cout << "Terima kasih!" << endl;
        break;  // Keluar dari loop
    } else {
        cout << "Pilihan tidak valid!" << endl;
    }
}
```

- **Contoh Kasus For Loop Yang Tepat:**

```
// 1. Memproses array (jumlah elemen diketahui)
int nilai[5] = {85, 90, 78, 92, 88};
for (int i = 0; i < 5; i++) {
    cout << "Nilai " << (i+1) << ": " << nilai[i] << endl;
}

// 2. Membuat pattern dengan ukuran pasti
for (int baris = 1; baris <= 5; baris++) {
    for (int kolom = 1; kolom <= baris; kolom++) {
        cout << "* ";
    }
    cout << endl;
}
```

- **Rule of Thumb:**
  - **Tahu pasti berapa kali loop** → gunakan **FOR**
  - **Tidak tahu berapa kali, tergantung kondisi** → gunakan **WHILE**

Do-While Loop

Slide 9

- **Struktur Do-While Loop:**

```
do {
    // blok kode yang diulang
    // PASTI dijalankan minimal 1 kali
} while (kondisi);
```

- **Perbedaan dengan While Loop:**
  - **While**: Cek kondisi dulu, baru eksekusi (bisa tidak pernah dieksekusi)
  - **Do-While**: Eksekusi dulu, baru cek kondisi (minimal 1x eksekusi)
- **Perbandingan Langsung:**

```
// WHILE LOOP - bisa tidak pernah dieksekusi
int x = 10;
while (x < 5) {
```

```
    cout << "Ini tidak akan tampil" << endl;
    x++;
}

// DO-WHILE LOOP - minimal 1x dieksekusi
int y = 10;
do {
    cout << "Ini akan tampil sekali" << endl;
    y++;
} while (y < 5);
```

- **Contoh Aplikasi Do-While:**

```
// 1. Menu program (harus tampil minimal sekali)
int pilihan;
do {
    cout << "\n=== KALKULATOR ===" << endl;
    cout << "1. Penjumlahan" << endl;
    cout << "2. Pengurangan" << endl;
    cout << "3. Keluar" << endl;
    cout << "Pilih menu (1-3): ";
    cin >> pilihan;

    if (pilihan == 1) {
        cout << "Mode Penjumlahan" << endl;
    } else if (pilihan == 2) {
        cout << "Mode Pengurangan" << endl;
```

```
    } else if (pilihan != 3) {
        cout << "Pilihan tidak valid!" << endl;
    }

} while (pilihan != 3);
cout << "Program selesai!" << endl;
```

```
// 2. Input validation (user harus coba minimal sekali)
int umur;
do {
    cout << "Masukkan umur Anda (1-120): ";
    cin >> umur;
    if (umur < 1 || umur > 120) {
        cout << "Umur tidak valid! Coba lagi." << endl;
    }
} while (umur < 1 || umur > 120);
cout << "Umur Anda: " << umur << " tahun" << endl;
```

- **Kapan Menggunakan Do-While:**
  - Menu program yang harus tampil minimal sekali
  - Input yang harus diminta minimal sekali
  - Game loop yang harus jalan minimal satu putaran

- Konfirmasi yang perlu ditanya minimal sekali

Break dan Continue Statement

Slide 10

- **Break Statement:**
  - Menghentikan loop secara paksa
  - Keluar dari loop dan lanjut ke statement setelah loop
  - Berguna untuk exit condition khusus

```
for (int i = 1; i <= 100; i++) {
    cout << i << " ";

    if(i == 5){
        break;
    }
}
cout << "Loop selesai." << endl;
```

- **Continue Statement:**

- Melewati sisa kode dalam iterasi saat ini
- Lanjut ke iterasi berikutnya
- Berguna untuk skip kondisi tertentu

```
// Contoh continue - print angka ganjil saja
cout << "Angka ganjil dari 1-10:" << endl;
for (int i = 1; i <= 10; i++) {
    if (i % 2 == 0) {  // Jika genap
        continue;      // Skip ke iterasi berikutnya
    }
    cout << i << " ";  // Hanya angka ganjil yang diprint
}
cout << endl;
// Output: 1 3 5 7 9
```

- **Contoh Kombinasi Break dan Continue:**

```
// Program mencari bilangan prima dari 2-50
cout << "Bilangan prima dari 2-50:" << endl;
for (int num = 2; num <= 50; num++) {
    bool prima = true;
```

```
    // Cek apakah num adalah bilangan prima
    for (int i = 2; i < num; i++) {
        if (num % i == 0) {
            prima = false;
            break;  // Tidak perlu cek lebih lanjut
        }
    }

    if (!prima) {
        continue;  // Skip jika bukan prima
    }

    cout << num << " ";
}
cout << endl;
```

- **Break dalam Nested Loop:**

```
// Break hanya keluar dari loop terdekat
for (int i = 1; i <= 3; i++) {
    cout << "Loop luar: " << i << endl;

    for (int j = 1; j <= 5; j++) {
        if (j == 3) {
            break;  // Hanya keluar dari loop dalam (j)
        }
        cout << "  Loop dalam: " << j << endl;
    }
```

```
}
// Loop luar tetap berlanjut sampai selesai
```

Latihan Perulangan - Level Mudah (Soal 1-5)

Slide 11

Soal 1: Deret Angka Sederhana

Buatlah program yang mencetak angka dari 1 sampai 10 secara berurutan.

Sample Output:

```
1 2 3 4 5 6 7 8 9 10
```

Soal 2: Hitung Mundur

Buatlah program yang mencetak angka dari 10 sampai 1 secara menurun.

Sample Output:

```
10 9 8 7 6 5 4 3 2 1
```

Soal 3: Tabel Perkalian

Buatlah program yang mencetak tabel perkalian untuk angka 5 (dari 5x1 sampai 5x10).

Sample Output:

```
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

Soal 4: Bilangan Genap

Buatlah program yang mencetak semua bilangan genap dari 2 sampai 20.

Sample Output:

```
2 4 6 8 10 12 14 16 18 20
```

Soal 5: Pola Bintang Sederhana

Buatlah program yang mencetak pola bintang sebanyak 5 baris, dimana baris ke-i memiliki i buah bintang.

Sample Output:

```
*
**
***
****
*****
```

Latihan Perulangan - Level Menengah (Soal 6-10)

Soal 6: Faktorial

Buatlah program yang menghitung faktorial dari angka 6 (6!). Tampilkan proses perhitungannya.

Sample Output:

```
6! = 6 x 5 x 4 x 3 x 2 x 1 = 720
```

Soal 7: Bilangan Prima Check

Buatlah program yang mengecek apakah angka 17 adalah bilangan prima atau bukan. Tampilkan proses pengecekannya.

Sample Output:

```
Mengecek apakah 17 prima...
17 % 2 = 1 (bukan 0)
```

```
17 % 3 = 2 (bukan 0)
17 % 4 = 1 (bukan 0)
17 % 5 = 2 (bukan 0)
17 % 6 = 5 (bukan 0)
17 % 7 = 3 (bukan 0)
17 % 8 = 1 (bukan 0)
17 % 9 = 8 (bukan 0)
17 % 10 = 7 (bukan 0)
17 % 11 = 6 (bukan 0)
17 % 12 = 5 (bukan 0)
17 % 13 = 4 (bukan 0)
17 % 14 = 3 (bukan 0)
17 % 15 = 2 (bukan 0)
17 % 16 = 1 (bukan 0)
17 adalah bilangan prima!
```

Soal 8: Jumlah Bilangan

Buatlah program yang menghitung jumlah semua bilangan dari 1 sampai 100.

Sample Output:

```
1 + 2 + 3 + ... + 100 = 5050
```

Soal 9: Pola Angka Piramida

Buatlah program yang mencetak pola angka berbentuk piramida setinggi 4 baris.

Sample Output:

```
   1
  123
 12345
1234567
```

Soal 10: Menu Sederhana

Buatlah program menu sederhana yang terus berulang sampai user memilih keluar. Menu memiliki 3 pilihan: 1=Halo, 2=Selamat, 3=Keluar.

Sample Output:

```
=== MENU ===
1. Tampilkan Halo
2. Tampilkan Selamat
3. Keluar
```

```
Pilihan: 1
Halo!

=== MENU ===
1. Tampilkan Halo
2. Tampilkan Selamat
3. Keluar
Pilihan: 2
Selamat!

=== MENU ===
1. Tampilkan Halo
2. Tampilkan Selamat
3. Keluar
Pilihan: 3
Terima kasih!
```

Latihan Perulangan - Level Lanjut (Soal 11-15)

Slide 13

Soal 11: Pattern Diamond

Buatlah program yang mencetak pola diamond (berlian) dengan tinggi 7 baris.

Sample Output:

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

Soal 12: Bilangan Fibonacci

Buatlah program yang mencetak 10 angka pertama dalam deret Fibonacci.

Sample Output:

```
Deret Fibonacci 10 angka pertama:
0 1 1 2 3 5 8 13 21 34
```

Soal 13: Cari Angka Terbesar dari Input

Buatlah program yang meminta user memasukkan 8 angka, lalu mencari angka terbesar. Tampilkan proses perbandingannya tanpa menyimpan semua angka.

Sample Output:

```
Masukkan 8 angka:
Angka ke-1: 23
Terbesar sementara: 23
Angka ke-2: 45
45 > 23, terbesar baru: 45
Angka ke-3: 12
12 <= 45, terbesar tetap: 45
Angka ke-4: 67
67 > 45, terbesar baru: 67
Angka ke-5: 34
34 <= 67, terbesar tetap: 67
```

```
Angka ke-6: 89
89 > 67, terbesar baru: 89
Angka ke-7: 56
56 <= 89, terbesar tetap: 89
Angka ke-8: 78
78 <= 89, terbesar tetap: 89
Angka terbesar adalah: 89
```

Soal 14: Kalkulator Rata-rata dengan Validasi

Buatlah program yang meminta user memasukkan 5 angka (harus antara 0-100), lalu hitung rata-ratanya. Jika input salah, minta input ulang. Hitung total dan rata-rata tanpa menyimpan semua angka.

Sample Output:

```
Masukkan 5 angka (0-100):
Angka ke-1: 85
Total sementara: 85
Angka ke-2: 150
Input tidak valid! Masukkan angka 0-100.
Angka ke-2: 92
Total sementara: 177
Angka ke-3: 78
Total sementara: 255
```

```
Angka ke-4: 95
Total sementara: 350
Angka ke-5: 88
Total sementara: 438

Jumlah total: 438
Rata-rata: 87.6
```

Soal 15: Segitiga Pascal

Buatlah program yang mencetak segitiga Pascal setinggi 5 baris.

Sample Output:

```
    1
   1 1
  1 2 1
 1 3 3 1
1 4 6 4 1
```