Perulangan I

Perulangan atau *loop* adalah teknik dalam pemrograman yang memungkinkan eksekusi serangkaian perintah secara berulang hingga kondisi tertentu terpenuhi. Dengan menggunakan perulangan, kita dapat menulis kode yang lebih efisien dan mudah dibaca, menghindari penulisan ulang instruksi yang sama berulang kali.

Jenis-Jenis Perulangan

Berdasarkan sifatnya, perulangan dibagi menjadi dua jenis:

1. **Counted Loop**: Perulangan yang jumlah iterasinya sudah ditentukan sebelum eksekusi dimulai.
2. **Uncounted Loop**: Perulangan yang berjalan hingga kondisi tertentu terpenuhi, sehingga jumlah iterasi tidak diketahui sejak awal.

Komponen Perulangan

Setiap perulangan terdiri dari beberapa komponen utama:

1. **Inisialisasi**: Kode yang dieksekusi sekali sebelum perulangan dimulai, biasanya untuk menetapkan nilai awal *variabel kontrol*.
2. **Kondisi**: Ekspresi logika yang menentukan apakah perulangan akan terus berjalan atau berhenti.
3. **Inkremen/Dekremen**: Operasi yang mengubah nilai *variabel kontrol* dalam setiap iterasi.
4. **Statement**: Blok kode yang dieksekusi berulang selama kondisi masih terpenuhi.
5. **Terminasi**: Perintah yang menghentikan perulangan jika kondisi tertentu dipenuhi (bersifat opsional).

Catatan: *Variabel kontrol* adalah variabel yang digunakan untuk mengatur jalannya perulangan dan harus memiliki nilai awal.

1. Perulangan `for`

Perulangan `for` digunakan saat jumlah iterasi sudah diketahui sejak awal. Oleh karena itu, `for` termasuk dalam kategori *counted loop*.

Sintaks

```
for (inisialisasi; kondisi; inkremen/dekremen) {
    statement;
}
```

Note:

Dalam perulangan for, urutan eksekusi adalah sebagai berikut:

1. **Inisialisasi** dieksekusi satu kali sebelum perulangan dimulai.
2. **Cek kondisi**, jika bernilai `true`, maka lanjut ke langkah berikutnya. Jika `false`, perulangan berhenti.
3. **Statement** dalam blok perulangan dieksekusi.
4. **Inkremen/Dekremen** dijalankan untuk memperbarui nilai variabel kontrol.
5. Kembali ke langkah 2 (Looping).

Contoh Kode

```
#include <stdio.h>

int main() {
    for (int i = 0; i < 5; i++) {
        printf("%d\n", i);
    }
    return 0;
}
```

Alur Kode

Iterasi	Nilai `i` sebelum cetak	Kondisi (`i < 5`)?	Cetak Output (aksi)	Nilai `i` setelah inkremen
1	0	`true`	0	1
2	1	`true`	1	2
3	2	`true`	2	3
4	3	`true`	3	4

Iterasi	Nilai `i` sebelum cetak	Kondisi (`i < 5`)?	Cetak Output (aksi)	Nilai `i` setelah inkremen
5	4	`true`	4	5
6	5	`false`	-	-

Output

```
0
1
2
3
4
```

2. Perulangan `while`

Perulangan `while` termasuk *uncounted loop* karena jumlah iterasi bergantung pada suatu kondisi yang bisa berubah selama eksekusi.

Sintaks

```
inisialisasi;
while (kondisi) {
    statement;
    inkremen/dekremen;
}
```

Contoh Kode

```c
#include <stdio.h>

int main() {
    int i = 0;
    while (i < 5) {
```

```
        printf("%d\n", i);
        i++;
    }
    return 0;
}
```

Alur Kode

Iterasi	Nilai `i` sebelum cetak	Kondisi (`i < 5`)?	Cetak Output	Nilai `i` setelah inkremen
1	0	`true`	0	1
2	1	`true`	1	2
3	2	`true`	2	3
4	3	`true`	3	4
5	4	`true`	4	5
6	5	`false`	-	-

Output

```
0
1
2
3
4
```

3. Perulangan `do while`

Sama seperti `while` , perulangan `do while` adalah *uncounted loop*. Perbedaannya, `do while` selalu menjalankan perintah minimal satu kali sebelum memeriksa kondisi.

Sintaks

```
inisialisasi;
do {
    statement;
    inkremen/dekremen;
} while (kondisi);
```

Contoh Kode

```
#include <stdio.h>

int main() {
    int i = 0;
    do {
        printf("%d\n", i);
        i++;
    } while (i < 5);
    return 0;
}
```

Alur Kode

Iterasi	Nilai `i` sebelum cetak	Cetak Output	Nilai `i` setelah inkremen	Kondisi (`i < 5`)?	Lanjut?
1	0	0	1	`true`	Ya
2	1	1	2	`true`	Ya
3	2	2	3	`true`	Ya
4	3	3	4	`true`	Ya
5	4	4	5	`true`	Ya
6	5	-	-	`false`	Tidak

Output

```
0
1
2
3
4
```

Guided

Axel, seorang mahasiswa, ingin membuat program kalkulator sederhana yang memanfaatkan konsep perulangan, percabangan, dan autentikasi. Kalkulator ini hanya bisa digunakan setelah pengguna berhasil login.

Spesifikasi Program

1. **Menu utama**
    - Program menampilkan menu utama saat dijalankan:

```
=== Kalkulator Sederhana ===
1. Login
2. Perkalian Loop
0. Exit
>>
```

2. **Fitur Login**
    - Pengguna harus memasukkan username dan password yang sudah ditentukan.
    - Jika salah, muncul pesan `[!] Invalid Credentials` dan pengguna diminta mencoba lagi.
    - Jika benar, muncul pesan `[!] Logged in successfully` , dan pengguna dapat mengakses fitur kalkulator.
3. **Perkalian dengan Perulangan**
    - Hanya bisa diakses jika pengguna sudah login.
    - Menghitung hasil perkalian dua angka menggunakan penjumlahan berulang.
4. **Fitur Exit**
    - Program berhenti jika pengguna memilih opsi `0` .

Implementasi Program

Note: komentar tidak wajib ditulis

main.c

```c
#include <stdio.h>   // Library standar input-output, digunakan untuk printf, scanf, dll.
#include <stdlib.h>  // Library standar untuk fungsi seperti system("cls") untuk membersihkan layar.
#include <stdbool.h> // Library untuk menggunakan tipe data boolean (true/false).

// Mendeklarasikan tipe data String sebagai array karakter berukuran 100.
typedef char String[100];

int main(int argc, char *argv[]) {

    // Variabel `run` mengontrol jalannya program. Selama `run = true`, program akan terus berjalan.
    bool run = true;

    // Variabel `authenticated` menandakan apakah pengguna sudah login atau belum.
    bool authenticated = false;

    // Variabel `menu` digunakan untuk menyimpan pilihan menu dari pengguna.
    int menu;

    // Menyimpan username dan password yang digunakan untuk login.
    String secret_name = "Axel";        // Ganti dengan nama panggilan praktikan.
    String secret_password = "230712306"; // Ganti dengan NPM praktikan.

    // Variabel untuk menyimpan input username dan password dari pengguna.
    String name, password;

    // Variabel untuk operasi matematika (perkalian dengan loop).
```

```
    int a, b, hasil;

    // Variabel kontrol untuk loop `for`.
    int i;

    // Perulangan `while` untuk menampilkan menu terus-menerus sampai user keluar.
    while(run) {
        system("cls"); // Membersihkan layar agar tampilan lebih rapi setiap kali menu di tampilkan.

        // Menampilkan menu utama.
        puts("=== Kalkulator Sederhana ===");
        puts("1. Login");
        puts("2. Perkalian Loop");
        puts("0. Exit");

        // Menerima input pilihan menu dari user.
        printf(">> ");
        scanf("%d", &menu);
        fflush(stdin); // Membersihkan buffer agar input tidak bermasalah.

        // Menggunakan `switch` untuk menangani pilihan user.
        switch(menu) {
            case 1:
                // Perulangan `do while` untuk meminta login sampai berhasil.
                do {
                    system("cls"); // Membersihkan layar.

                    // Meminta input username.
                    printf("Name: ");
                    fflush(stdin);  // Membersihkan buffer input agar tidak ada data yang tertinggal.
                    gets(name);     // Mengambil input sebagai string.

                    // Meminta input password.
```

```
                printf("Password: ");
                fflush(stdin);
                gets(password);

                // Mengecek apakah username dan password cocok dengan yang disimpan.
                if (strcmp(name, secret_name) != 0 || strcmp(password, secret_passwor
d) != 0) {
                    // Jika salah, tampilkan pesan error.
                    puts("[!] Invalid Credentials");
                    getch(); // Menunggu user menekan tombol sebelum melanjutkan.
                }
            } while(strcmp(name, secret_name) != 0 || strcmp(password, secret_passwor
d) != 0);

            // Jika login berhasil, ubah status `authenticated` menjadi `true`.
            authenticated = true;
            puts("[!] Logged in successfully");

            break;

        case 2:
            // Mengecek apakah user sudah login sebelum menggunakan fitur perkalian.
            if (!authenticated) {
                puts("[!] Login required"); // Jika belum login, tampilkan pesan peri
ngatan.
            } else {
                // Meminta user memasukkan angka pertama (a) dan angka kedua (b).
                printf("Masukkan angka pertama (a): ");
                scanf("%d", &a);
                printf("Masukkan angka kedua (b): ");
                scanf("%d", &b);

                // Inisialisasi hasil perkalian.
                hasil = 0;
```

```
                    // Menggunakan `for loop` untuk melakukan perkalian dengan penjumlaha
n berulang.
                    for (i = 0; i < a; i++) {
                        hasil += b; // Menambahkan b ke hasil sebanyak a kali.
                    }

                    // Menampilkan hasil perkalian.
                    printf("\nHasil %d x %d = %d", a, b, hasil);
                }

                break;

            case 0:
                run = false;
                puts("Good Bye...");

                break;
            default:
                // Jika user memasukkan angka yang tidak ada di menu
                puts("[!] Invalid menu");
                break;
        }

        getch(); // Menunggu user menekan tombol sebelum kembali ke menu.
    }

    return 0;
}
```

Ketentuan Pengerjaan

- Ekstensi file harus .c
- Komentar tidak wajib

- Folder hasil pekerjaan di-zip dengan format:

  GD_X_YYYYY.zip

  X = Kelas mahasiswa

  Y = Lima digit terakhir NPM mahasiswa

Contact

Jika ada pertanyaan, silakan hubungi saya (Axel Liang Gono) melalui Teams atau WhatsApp.

Terima kasih, selamat belajar!