

PERULANGAN II

Pada bagian sebelumnya, kita telah mempelajari dasar-dasar perulangan (looping) dalam bahasa C, termasuk `for`, `while`, dan `do while`. Sekarang, kita akan mempelajari tiga konsep lanjutan dalam perulangan, yaitu `break`, `continue`, dan `nested loop`.

1. `break`

`break` digunakan untuk menghentikan perulangan secara paksa, bahkan jika kondisi perulangan masih bernilai true. `break` sering digunakan dalam situasi tertentu seperti ketika sebuah kondisi khusus terjadi dalam perulangan.

Sintaks

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

Contoh Kode

```
#include <stdio.h>  
  
int main() {  
    for (int i = 1; i <= 10; i++) {  
        if (i == 5) {  
            break; // Menghentikan perulangan saat i = 5  
        }  
        printf("%d ", i);  
    }  
    return 0;  
}
```

Alur Iterasi

Iterasi	Nilai <code>i</code>	<code>i == 5</code> (Break?)	Output
1	1	Tidak	1
2	2	Tidak	2
3	3	Tidak	3
4	4	Tidak	4
5	5	Ya (<code>break</code>)	(Stop)

Output

```
1 2 3 4
```

Penjelasan

- Perulangan for dimulai dari `i = 1` hingga `i <= 10`.
- Ketika `i == 5`, `break` dieksekusi, sehingga perulangan berhenti sebelum mencetak angka 5.

2. `continue`

`continue` digunakan untuk melewati iterasi tertentu dalam perulangan dan langsung melanjutkan ke iterasi berikutnya. Ini berguna jika kita ingin melewati perintah tertentu tanpa menghentikan seluruh perulangan.

Sintaks

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

Contoh Kode

```
#include <stdio.h>

int main() {
    for (int i = 1; i <= 10; i++) {
        if (i == 5) {
            continue; // Melewati iterasi saat i = 5
        }
        printf("%d ", i);
    }
    return 0;
}
```

Alur Iterasi

Iterasi	Nilai i	i == 5 (Continue?)	Output
1	1	Tidak	1
2	2	Tidak	2
3	3	Tidak	3
4	4	Tidak	4
5	5	Ya (continue)	(Lewat)
6	6	Tidak	6
7	7	Tidak	7
8	8	Tidak	8
9	9	Tidak	9
10	10	Tidak	10

Output

1 2 3 4 6 7 8 9 10

Penjelasan

- Saat `i == 5`, `continue` dieksekusi sehingga perintah `printf` dilewati untuk iterasi ini.
- Perulangan tetap berjalan untuk nilai `i` lainnya.

3. nested loop

Nested loop adalah perulangan yang berada di dalam perulangan lain. Konsep ini memungkinkan kita membuat struktur berulang yang lebih kompleks, seperti tabel, pola bintang, atau perulangan multidimensi.

Sintaks

```
for (inisialisasi_1; kondisi_1; inkremen_1) {  
    for (inisialisasi_2; kondisi_2; inkremen_2) {  
        statement;  
    }  
}
```

Contoh Kode

```
#include <stdio.h>  
  
int main() {  
    int i, j;  
  
    for (i = 1; i <= 3; i++) {  
        for (j = 1; j <= 3; j++) {  
            printf("%d ", i * j);  
        }  
        printf("\n");  
    }  
}
```

```
    return 0;  
}
```

Alur Iterasi

Iterasi i	Iterasi j	i * j	Output
1	1	1×1=1	1
1	2	1×2=2	2
1	3	1×3=3	3
			\n
2	1	2×1=2	2
2	2	2×2=4	4
2	3	2×3=6	6
			\n
3	1	3×1=3	3
3	2	3×2=6	6
3	3	3×3=9	9
			\n

Output

```
1 2 3  
2 4 6  
3 6 9
```

Penjelasan

- Perulangan luar (i) mengontrol jumlah baris.

- Perulangan dalam (j) mencetak angka di setiap baris.
- Setelah perulangan dalam (j) selesai, `printf("\n")` mencetak baris baru.

Guided

Pada project sebelumnya, Axel sudah membuat program kalkulator sederhana. Kali ini, Axel ingin menambah fitur pada program kalkulator sederhananya agar lebih sempurna. Kali ini, ia ingin menambahkan konsep `break`, `continue`, dan `nested loop` pada programnya untuk membuat tabel perkalian.

Spesifikasi Program

1. Menu utama

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

2. Fitur login

- Mirip dengan sebelumnya, hanya pengguna yang memasukkan username dan password yang benar dapat mengakses fitur lainnya.

3. Perkalian loop

- Sama seperti sebelumnya, menggunakan penjumlahan berulang.

4. Tabel perkalian

- Menggunakan `nested loop` untuk mencetak tabel perkalian berdasarkan input pengguna.
- Jika pengguna memasukkan angka lebih dari 10, gunakan `continue` untuk meminta ulang input.
- Jika pengguna ingin keluar dari menu ini, mereka bisa memasukkan `0`, yang akan menggunakan `break`.

5. Fitur Keluar

- Jika pengguna memilih `0`, program berhenti.

Implementasi Program

Note: komentar tidak perlu ditulis

main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

typedef char String[100];

int main() {
    bool run = true;
    bool authenticated = false;
    int menu;

    String secret_name = "Axel"; // ganti dengan nama panggilan praktikan
    String secret_password = "230712306"; // ganti dengan NPM praktikan
    String name, password;

    int a, b, hasil;
    int i, j, n;

    while(run) {
        system("cls");
        puts("=== Kalkulator Sederhana ===");
        puts("1. Login");
        puts("2. Perkalian Loop");
        puts("3. Tabel Perkalian");
        puts("0. Exit");

        printf(">> ");
        scanf("%d", &menu);
```

```
fflush(stdin);

switch(menu) {
    case 1:
        do {
            system("cls");
            printf("Name: ");
            fflush(stdin);
            gets(name);
            printf("Password: ");
            fflush(stdin);
            gets(password);

            if (strcmp(name, secret_name) != 0 || strcmp(password, secret_password) != 0) {
                puts("[!] Invalid Credentials");
                getch();
            }
        } while(strcmp(name, secret_name) != 0 || strcmp(password, secret_password) != 0);

        authenticated = true;
        puts("[!] Logged in successfully");
        break;

    case 2:
        if (!authenticated) {
            puts("[!] Login required");
        } else {
            printf("Masukkan angka pertama (a): ");
            scanf("%d", &a);
            printf("Masukkan angka kedua (b): ");
            scanf("%d", &b);

            hasil = 0;
        }
    }
}
```



```

    for (i = 0; i < a; i++) {
        hasil += b;
    }

    printf("\nHasil %d x %d = %d", a, b, hasil);
}
break;

```

case 3:

```

if (!authenticated) { // Mengecek apakah pengguna sudah login atau belum
    puts("[!] Login required");
} else {
    do {
        // Meminta input angka untuk membuat tabel perkalian
        printf("Masukkan angka (maks 10, 0 untuk keluar): ");
        scanf("%d", &n);

        if (n == 0) { // Jika pengguna memasukkan 0, keluar dari loop
            break; // menghentikan loop `do while`
        }

        if (n > 10) { // Jika angka lebih dari 10, tampilkan pesan dan l

```

anjut ke iterasi berikutnya

```

        puts("[!] Angka terlalu besar, coba lagi.");
        continue; // lanjut ke iterasi berikutnya, kode for loop di b

```

awah dilangkahi

```

    }

    // Membuat tabel perkalian n x n
    for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++) {
            printf("%d\t", i * j); // Mencetak hasil perkalian i * j
        }
        printf("\n"); // Pindah ke baris baru setelah satu baris tab
    }
}

```

el selesai

```

        } while (n != 0); // Loop terus berjalan sampai pengguna memasukkan
0
    }
    break;

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

    default:
        puts("[!] Invalid menu");
        break;
}

getch();
}

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!