Pertemuan 3: Perulangan dan Percabangan (Seleksi Kondisi) pada Bahasa Pemrograman Go”

1. Percabangan

Untuk mengambil keputusan berdasarkan kondisi tertentu.

If

```
nilai := 70

if nilai ⩾ 75 {
    fmt.Println("Lulus")
} else {
    fmt.Println("Tidak Lulus")
}
```

Output: Tidak Lulus

Nested if

```
umur := 20
status := "pelajar"

if umur ⩾ 18 {
    if status == "pelajar" {
        fmt.Println("Anda dewasa dan masih pelajar")
    } else {
        fmt.Println("Anda dewasa dan bukan pelajar")
    }
}
```

Output: Grade B

if, elif, else

```
if nilai ⩾ 90 {
    fmt.Println("Grade A")
} else if nilai ⩾ 80 {
    fmt.Println("Grade B")
} else if nilai ⩾ 70 {
    fmt.Println("Grade C")
} else {
    fmt.Println("Grade D")
}
```

Output: Anda dewasa dan masih pelajar

switch case

```
hari := "Senin"

switch hari {
case "Senin":
    fmt.Println("Hari kerja")
case "Sabtu", "Minggu":
    fmt.Println("Hari libur")
default:
    fmt.Println("Hari biasa")
}
```

Output: Hari kerja

Aspek	`if-else`	`switch-case`
Kegunaan	Untuk mengevaluasi ekspresi logika kompleks.	Untuk membandingkan satu nilai dengan banyak kemungkinan.
Kejelasan Kode	Bisa menjadi panjang dan tidak rapi jika banyak kondisi.	Lebih ringkas dan mudah dibaca untuk banyak pilihan.
Evaluasi Kondisi	Bisa mengevaluasi ekspresi logika, seperti `x > 5`.	Hanya cocok untuk membandingkan nilai tertentu.
Ekspresi Kompleks	Mendukung operator logika seperti `&&`, `	
Default	Gunakan `else` untuk menangani kondisi selainnya.	Gunakan `default` untuk kondisi selain case yang disebut.
Fallthrough	Tidak ada.	Mendukung `fallthrough` ke case berikutnya.

2. Perulangan

Untuk mengeksekusi blok kode berulang kali. Go hanya memiliki perulangan for.

```go
for i := 0; i < 3; i++ {
    fmt.Println(i)
}
```

Output:

0
1
2

```go
buah := []string{"Apel", "Jeruk", "Mangga"}

for index, item := range buah {
    fmt.Println("Index:", index, "Buah:", item)
}
```

Output : Index: 0 Buah: Apel
Index: 1 Buah: Jeruk
Index: 2 Buah: Mangga

Infinite Loop

```
i := 1
    for {
        fmt.Println("Perulangan ke-", i)
        i++
        if i > 3 {
            break // keluar dari loop
        }
    }
```

Perulangan ke- 1
Perulangan ke- 2
Perulangan ke- 3

3. Break, Continue, dan Counter

Konsep	Deskripsi
Break	Menghentikan perulangan secara paksa dan keluar dari loop, meskipun kondisi loop masih bernilai true.
Continue	Melewati sisa kode dalam iterasi saat ini dan langsung melanjutkan ke iterasi berikutnya.
Counter	Variabel penghitung yang digunakan untuk mencatat jumlah iterasi atau kejadian tertentu dalam loop.