

Soal

1. Struktur Kontrol

```
package main

import (
    "fmt"
)

func main() {
    // 1. Struktur kondisional if-else
    nilai := 85
    fmt.Println("Contoh if-else:")
    if nilai >= 90 {
        fmt.Println("Nilai A ") // (1) Lengkapi bagian ini untuk
        mencetak "Nilai A"
    } else if nilai >= 80 {
        fmt.Println("Nilai B")
    } else if nilai >= 70 {
        fmt.Println("Nilai C") // (2) Lengkapi bagian ini untuk
        mencetak "Nilai C"
    } else if nilai >= 60 {
        fmt.Println("Nilai D")
    } else {
        fmt.Println("Nilai E") // (3) Lengkapi bagian ini untuk mencetak
        "Nilai E"
    }

    // 2. Struktur perulangan for (seperti while)
    fmt.Println("\nContoh for sebagai while:")
    counter := 1
    for counter <= 5 {
        fmt.Printf("iterasi ke-%d\n", counter) // (4) Lengkapi bagian ini
        untuk mencetak "Iterasi ke-X"
        counter++
    }
}
```

```

    }

// 3. Struktur perulangan for dengan range
fmt.Println("\nContoh for dengan range:")
buah := []string{"Apel", "Mangga", "Jeruk", "Pisang"}
for _, item := range buah { // (5) Lengkapi bagian ini agar mencetak
indeks dan nama buah
    fmt.Println("Buah : ", item) // (6) Lengkapi bagian ini untuk
mencetak "Buah pada index X adalah Y"
}

// 4. Struktur switch-case
fmt.Println("\nContoh switch-case:")
hari := "Senin"
switch hari {
case "Senin":
    fmt.Println("Hari kerja")// (7) Lengkapi bagian ini untuk
mencetak "Hari kerja"
case "Selasa":
    fmt.Println("Hari kerja")
case "Rabu":
    fmt.Println("Hari kerja lengkap")// (8) Lengkapi bagian ini agar
hari kerja lengkap
case "Kamis":
    fmt.Println("Hari kerja")
case "Jumat":
    fmt.Println("Hari kerja") // (9) Lengkapi bagian ini untuk
mencetak "Hari kerja"
case "Sabtu", "Minggu":
    fmt.Println("Hari libur") // (10) Lengkapi bagian ini untuk
mencetak "Hari libur"
default:
    fmt.Println("Hari tidak valid")
}
}

```

2. Fungsi

```
package main

import (
    "fmt"
    "math"
)

// Fungsi dengan parameter dan return value
func hitungLuasLingkaran(jariJari float64) float64 {
    return math.Pi * jariJari * jariJari // (1) Lengkapi rumus luas lingkaran
}

// Fungsi dengan multiple return values
func minMax(angka []int) (int, int) {
    if len(angka) == 0 {
        return 0, 0 // (2) Pastikan return value yang benar jika array
        kosong
    }

    min := angka[0]
    max := angka[0]

    for _, nilai := range angka {
        if nilai < min {
            min = nilai // (3) Lengkapi agar min selalu mendapat nilai
            terkecil
        }
        if nilai > max {
            max = nilai // (4) Lengkapi agar max selalu mendapat
            nilai terbesar
        }
    }
}
```

```

        return min, max
    }

// Fungsi dengan named return values
func hitungStatistik(angka []float64) (min, max, avg float64) {
    if len(angka) == 0 {
        return 0, 0, 0
    }

    min = angka[0]
    max = angka[0]
    var total float64 = 0

    for _, nilai := range angka {
        if nilai < min {
            min = nilai // (5) Lengkapi agar min selalu mendapat
nilai terkecil
        }
        if nilai > max {
            mmax = nilai // (6) Lengkapi agar max selalu mendapat
nilai terbesar
        }
        total += nilai
    }

    avg = total / float64(len(angka)) // (7) Lengkapi perhitungan rata-rata
    return // implisit return untuk named return values
}

// Fungsi dengan variadic parameter
func jumlahkan(angka ...int) int {
    total := 0
    for _, nilai := range angka {
        total += nilai // (8) Lengkapi proses penjumlahan
    }
}

```

```

    }
    return total
}

```

```

func main() {
    // Contoh penggunaan fungsi dengan return value
    radius := 7.0
    luas := hitungLuasLingkaran(radius float64)
    // (9) Panggil fungsi hitungLuasLingkaran dengan parameter yang benar
    fmt.Printf("Luas lingkaran dengan jari-jari %.1f adalah %.2f\n",
radius, luas)

    // Contoh penggunaan fungsi dengan multiple return values
    data := []int{23, 45, 12, 67, 34, 8}
    minimal, maksimal := _____ // (10) Panggil
fungsi minMax dengan parameter yang benar
    fmt.Printf("Nilai minimum: %d, Nilai maksimum: %d\n", minimal,
maksimal)
}

```

3. Prosedur

```
package main
```

```
import (  
    "fmt"  
)
```

```
// Prosedur sederhana tanpa parameter
```

```
func tampilkanHeader() {  
    fmt.Print("=====") // (1) Lengkapi untuk  
    mencetak garis atas  
    fmt.Println("    PROGRAM MAHASISWA    ")  
    fmt.Print("=====") // (2) Lengkapi untuk  
    mencetak garis bawah  
}
```

```
// Prosedur dengan parameter value
```

```
func tampilkanInfo(nama string, nim string, jurusan string) {  
    fmt.Println("Informasi Mahasiswa:")  
    fmt.Printf("Nama   : %s\n", nama)  
    fmt.Printf("NIM    : %s\n", nim) // (3) Lengkapi agar mencetak NIM  
    dengan format yang benar  
    fmt.Printf("Jurusan : %s\n", jurusan)  
}
```

```
// Prosedur dengan parameter pointer
```

```
func ubahNilai(nilai *int) {  
    *nilai += 10  
    fmt.Printf("Nilai setelah diubah : %d\n", nilai) // (4) Lengkapi agar  
    mencetak nilai setelah diubah  
}
```

```
// Prosedur dengan struct parameter
```

```

type Mahasiswa struct {
    Nama    string
    NIM     string
    Jurusan string
    Nilai   map[string]int
}

func tampilkanNilai(mhs Mahasiswa) {
    fmt.Printf("Nilai mahasiswa %s:\n", mhs.Nama)
    for matkul, nilai := range mhs.Nilai {
        fmt.Printf("Nama Mata kuliah : %s , dengan nilai %d\n ",
matkul, nilai) // (5) Lengkapi agar mencetak nama mata kuliah dan nilai
    }
}

// Prosedur dengan slice parameter
func tampilkanDaftarMahasiswa(daftar []string) {
    fmt.Println("Daftar Mahasiswa:")
    for i, nama := range daftar {
        fmt.Printf("Nomor %d.Nama Mahasiswa %s\n", i+1, nama)//
(6) Lengkapi agar mencetak nomor dan nama mahasiswa
    }
}

func main() {
    // Memanggil prosedur tanpa parameter
    tampilkanHeader()// (7) Lengkapi agar memanggil prosedur
tampilkanHeader

    // Memanggil prosedur dengan parameter value
    tampilkanInfo() // (8) Lengkapi agar memanggil prosedur
tampilkanInfo dengan data yang sesuai

    // Memanggil prosedur dengan parameter pointer
    nilai := 75

```

```

    fmt.Printf("Nilai awal: %d\n", nilai)
    ubahNilai(&nilai) // (9) Lengkapi agar memanggil prosedur ubahNilai
dengan parameter yang benar
    fmt.Printf("Nilai akhir: %d\n", nilai)

// Memanggil prosedur dengan struct parameter
mhs := Mahasiswa{
    Nama:   "Ani Wijaya",
    NIM:    "87654321",
    Jurusan: "Sistem Informasi",
    Nilai: map[string]int{
        "Algoritma":      85,
        "Basis Data":      90,
        "Pemrograman Web": 78,
        "Struktur Data":   82,
    },
}

tampilkanNilai(mhs Mahasiswa) // (10) Lengkapi agar memanggil
prosedur tampilkanNilai dengan parameter yang sesuai
}

```


4. Rekursif

```
package main

import (
    "fmt"
)

// Rekursif untuk menghitung faktorial
func faktorial(n int) int {
    // Basis/kondisi penghentian rekursi
    if n == 0 || n == 1 {
        return 1
    }
    // Langkah rekursif
    return n * faktorial(n-1) // (1) Lengkapi bagian ini
}

// Rekursif untuk menghitung bilangan Fibonacci
func fibonacci(n int) int {
    if n <= 1 {
        return n
    }
    return fibonacci(n-1) + fibonacci(n-2) // (2) Lengkapi bagian ini
}

// Rekursif untuk menghitung pangkat
func pangkat(base int, eksponen int) int {
    if eksponen == 0 {
        return 1
    }
    return base * pangkat(base, eksponen-1) // (3) Lengkapi bagian ini
}

// Rekursif untuk mengecek palindrome
```

```

func isPalindrome(s string) bool {
    if len(s) <= 1 {
        return true
    }
    if s[0] != s[len(s)-1] {
        return false
    }
    return isPalindrome(s[1] : s[len(s)-1]) // (4) Lengkapi bagian ini
}

// Rekursif dengan helper function (untuk menghitung jumlah elemen array)
func sum(arr []int) int {
    return sumHelper(arr, 0) // (5) Lengkapi bagian ini
}

func sumHelper(arr []int, index int) int {
    if index >= len(arr) {
        return 0
    }
    return arr[index] + sumHelper(arr, index+1) // (6) Lengkapi bagian ini
}

func main() {
    // Contoh penggunaan rekursif faktorial
    fmt.Printf("Faktorial 5 = %d\n", faktorial(5))

    // Contoh penggunaan rekursif fibonacci
    fmt.Println("Deret Fibonacci:")
    for i := 0; i < 10; i++ {
        fmt.Printf("%d ", fibonnaci(i)) // (7) Lengkapi bagian ini
    }
    fmt.Println()

    // Contoh penggunaan rekursif pangkat

```

```
fmt.Printf("2 pangkat 8 = %d\n", pangkat(2,8)) // (8) Lengkapi bagian  
ini
```

```
// Contoh penggunaan rekursif palindrome  
kata1 := "katak"  
kata2 := "mobil"  
fmt.Printf("Apakah '%s' palindrome? %t\n", kata1,  
isPalindrome(kata1)) // (9) Lengkapi bagian ini  
fmt.Printf("Apakah '%s' palindrome? %t\n", kata2,  
isPalindrome(kata2)) // (10) Lengkapi bagian ini
```

```
// Contoh penggunaan rekursif dengan helper function  
angka := []int{1, 2, 3, 4, 5}  
fmt.Printf("Jumlah elemen array = %d\n", sum(angka))  
}
```