

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VII
STRUCT AND ARRAY**



Disusun Oleh :

Avrizal Setyo Aji Nugroho

2311102145

IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Struct adalah tipe data komposit yang memungkinkan pengelompokan berbagai atribut dengan tipe data yang berbeda ke dalam satu entitas. Struct mewakili objek atau data kompleks. Misalnya type Person Struct {Name string; Age int} mendefinisikan struct bernama Person yang memiliki dua atribut: Nama (string) dan Usia (integer). Struct mendukung metode (fungsi terkait) untuk meningkatkan modularitas dan keterbacaan kode dan membantu menyederhanakan pengelolaan data yang memiliki banyak properti sekaligus.

Array adalah koleksi elemen yang memiliki tipe data yang sama dan ukuran yang sama. Dengan menggunakan indeks berbasis nol, setiap elemen dalam array dapat diakses. Sebagai contoh, array integer yang terdiri dari lima elemen dapat dideklarasikan menggunakan `var numbers [5] int`. Saat deklarasi, ukuran array harus ditentukan, sehingga array di Golang bersifat statis. Untuk mengelola data terstruktur yang lebih kompleks, kombinasi antara struct dan array sering digunakan; ini termasuk array yang mengandung struct (seperti daftar pengguna) atau struct yang memiliki atribut yang menyerupai array (seperti nilai ujian dalam struct Student). Hal ini memungkinkan pembuatan aplikasi yang memiliki pola data yang dinamis dan terorganisir.

II. GUIDED

1. Map

Sourcecode

```
//Guided 2 - Map
Package main
import (
    "fmt"
)

func main() {
    // Membuat map dengan nama buah sebagai kunci dan
    harga sebagai nilai
    hargaBuah := map[string]int{
        "Apel": 5000,
        "Pisang": 3000,
        "Mangga": 7000,
    }

    // Menampilkan harga dari setiap buah
    fmt.Println("Harga Buah:")
    for buah, harga := range hargaBuah {
        fmt.Printf("%s: Rp%d\n", buah, harga)
    }

    fmt.Print("Harga buah Mangga = ",
    hargaBuah["Mangga"])
}
```

Screenshoot Output

```
PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Guided_map.go"
Harga Buah:
Apel: Rp5000
Pisang: Rp3000
Mangga: Rp7000
Harga buah Mangga = 7000
PS D:\praktikum alpro> 
```

Deskripsi Program

Program tersebut adalah implementasi bahasa Go yang sederhana untuk menyimpan data pasangan kunci-nilai. Map "hargaBuah" digunakan untuk menyimpan nama buah sebagai string dan harganya sebagai nilai. Program dimulai dengan membuat dan mengaktifkan map "hargaBuah" yang berisi harga buah-buahan seperti apel, pisang, dan mangga. Kemudian, program menggunakan loop "for range" untuk mencetak semua nama buah dan harganya. Akhirnya, program dapat mengakses harga buah mangga secara khusus dengan menggunakan kunci "Mangga". Output program berupa daftar nama buah dan harganya, serta informasi khusus tentang harga buah mangga.

2. Slice

Sourcecode

```
// Guided 2 - Slice
package main

import (
    "fmt"
)

// Fungsi untuk mengecek apakah nama sudah ada di dalam slice
func sudahAda(daftarTeman []string, nama string) bool {
    for _, teman := range daftarTeman {
        if teman == nama {
            return true
        }
    }
    return false
}

func main() {
```

```

// Slice awal untuk daftar teman dengan beberapa
data
daftarTeman := []string{"Andi", "Budi", "Cici"}

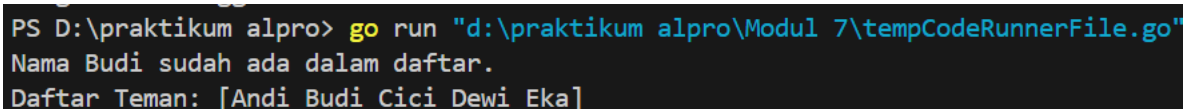
// Nama-nama baru yang ingin ditambahkan
namaBaru := []string{"Dewi", "Budi", "Eka"}

// Menambahkan nama baru hanya jika belum ada di
daftar
for _, nama := range namaBaru {
    if !sudahAda(daftarTeman, nama) {
        daftarTeman = append(daftarTeman, nama)
    } else {
        fmt.Println("Nama", nama, "sudah ada
dalam daftar.")
    }
}

// Menampilkan daftar teman akhir
fmt.Println("Daftar Teman:", daftarTeman)
}

```

Screenshoot Output



```

PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\tempCodeRunnerFile.go"
Nama Budi sudah ada dalam daftar.
Daftar Teman: [Andi Budi Cici Dewi Eka]

```

Deskripsi Program

Program di atas adalah implementasi sederhana dari slice dalam bahasa Go yang berfungsi untuk mengelola daftar nama teman. Ini menggunakan algoritma pencarian linear untuk menentukan apakah sebuah nama sudah ada dalam daftar teman sebelum menambahkannya. Pada awalnya, terdapat slice "daftarTeman" yang berisi nama-nama teman yang sudah terdaftar dan nama-nama baru yang ingin ditambahkan. Fungsi "sudahAda" digunakan untuk mengecek apakah nama ada dalam slice

"namaBaru", dan program kemudian memproses setiap nama dalam "namaBaru"; jika nama belum ada di daftar, nama tersebut ditambahkan ke slice "daftarTeman" menggunakan fungsi "Hasil terakhir adalah daftar teman yang selalu diperbarui, yang ditampilkan ke output.

3. Durasi Parkir

Sourcecode

```
package main

import "fmt"

type waktu struct {
    jam, menit, detik int
}

func main() {
    var wParkir, wPulang, durasi waktu
    var dParkir, dPulang, lparkir int

    fmt.Scan(&wParkir.jam, &wParkir.menit,
&wParkir.detik)
    fmt.Scan(&wPulang.jam, &wPulang.menit,
&wPulang.detik)

    dParkir = wParkir.detik + wParkir.menit*60 +
wParkir.jam*3600
    dPulang = wPulang.detik + wPulang.menit*60 +
wPulang.jam*3600
    lparkir = dPulang - dParkir

    durasi.jam = lparkir / 3600
    durasi.menit = lparkir % 3600 / 60
    durasi.detik = lparkir % 3600 % 60
    fmt.Printf("LAMA PARKIR : %d jam %d menit %d detik",
durasi.jam, durasi.menit, durasi.detik)
```

```
}
```

Screenshoot Output

```
PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\tempCodeRunnerFile.go"
5 10 12
6 10 11
LAMA PARKIR : 0 jam 59 menit 59 detik
PS D:\praktikum alpro> █
```

Deskripsi Program

Program di atas menggunakan struktur "waktu" untuk menyimpan data dalam bentuk jam, menit, dan detik untuk menghitung durasi parkir. Untuk mempermudah perhitungan perbedaan waktu, pengguna memasukkan waktu masuk (wParkir) dan waktu keluar (wPulang), yang kemudian dikonversi menjadi total detik. Durasi parkir dalam detik (lparkir), yang kemudian dikonversi kembali ke format jam, menit, dan detik dengan menggunakan operasi pembagian dan modulus. Hasil akhirnya ditampilkan dalam format "X jam Y menit Z detik", memberikan detail tentang waktu parkir.

III. UNGUIDED

1. Sourcecode

```
package main

import (
    "fmt"
    "math"
)

type Circle_145 struct {
    x, y, radius float64
}

type Poin_145 struct {
    x, y float64
}

// Fungsi untuk menghitung jarak antara dua titik
func jarak(x1, y1, x2, y2 float64) float64 {
```

```

        return math.Sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2))
    }

    // Fungsi untuk mengecek apakah suatu titik berada di
    // dalam lingkaran
    func didalam(c Circle_145, p Poin_145) bool {
        return jarak(c.x, c.y, p.x, p.y) <= c.radius
    }

    func main() {
        var circle1, circle2 Circle_145
        fmt.Println("Masukkan koordinat pusat dan radius
        lingkaran 1 (cx1, cy1, r1):")
        fmt.Scan(&circle1.x, &circle1.y, &circle1.radius)

        fmt.Println("Masukkan koordinat pusat dan radius
        lingkaran 2 (cx2, cy2, r2):")
        fmt.Scan(&circle2.x, &circle2.y, &circle2.radius)

        var point Poin_145
        fmt.Println("Masukkan koordinat titik (x, y):")
        fmt.Scan(&point.x, &point.y)

        inCircle1 := didalam(circle1, point)
        inCircle2 := didalam(circle2, point)

        if inCircle1 && inCircle2 {
            fmt.Println("Titik di dalam lingkaran 1 dan
            2")
        } else if inCircle1 {
            fmt.Println("Titik di dalam lingkaran 1")
        } else if inCircle2 {
            fmt.Println("Titik di dalam lingkaran 2")
        } else {
            fmt.Println("Titik di luar lingkaran 1 dan 2")
        }
    }
}

```

Screenshoot Output

```

PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Unguided1.go"
Masukkan koordinat pusat dan radius lingkaran 1 (cx1, cy1, r1):
1 1 5
Masukkan koordinat pusat dan radius lingkaran 2 (cx2, cy2, r2):
2 3 4
Masukkan koordinat titik (x, y):
2 1
Titik di dalam lingkaran 1 dan 2
PS D:\praktikum alpro>

```

Deskripsi Program

Program di atas digunakan untuk menentukan apakah sebuah titik termasuk dalam satu atau dua lingkaran tertentu. Program ini menggunakan dua struktur data: "Circle_145" menunjukkan lingkaran dengan atribut pusat "(x, y)" dan "radius", dan "Poin_145" menunjukkan koordinat titik. Program bekerja dengan menerima input untuk kedua lingkaran ("Circle1" dan "Circle2"), serta koordinat pusat dan radius untuk satu titik. Untuk mengetahui apakah jarak antara titik yang diuji dan pusat lingkaran lebih kecil atau sama dengan radius lingkaran, fungsi "jarak" digunakan. Program menunjukkan apakah titik berada di dalam salah satu atau kedua lingkaran atau di luar keduanya berdasarkan hasil cek dua lingkaran. Outputnya membantu mengetahui posisi relatif titik terhadap lingkaran.

2. Sourcecode

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var n int
    fmt.Print("Masukkan jumlah elemen array: ")
    fmt.Scan(&n)

    array_145 := make([]int, n)
    fmt.Println("Masukkan elemen array:")
    for i := 0; i < n; i++ {
        fmt.Printf("Elemen ke-%d: ", i)
        fmt.Scan(&array_145[i])
    }

    //Menampilkan seluruh isi array
    fmt.Println("Seluruh isi array:", array_145)

    // Menampilkan indeks ganjil
    fmt.Print("Elemen dengan indeks ganjil: ")
    for i := 1; i < len(array_145); i += 2 {
        fmt.Print(array_145[i], " ")
    }
    fmt.Println()

    // Menampilkan indeks genap
    fmt.Print("Elemen dengan indeks genap: ")
    for i := 0; i < len(array_145); i += 2 {
```

```

        fmt.Print(array_145[i], " ")
    }
    fmt.Println()

    // Menampilkan elemen indeks kelipatan x
    var x int
    fmt.Print("Masukkan bilangan x untuk kelipatan
indeks: ")
    fmt.Scan(&x)
    fmt.Print("Elemen dengan indeks kelipatan ", x, ":
")
    for i := 0; i < len(array_145); i++ {
        if i%x == 0 {
            fmt.Print(array_145[i], " ")
        }
    }
    fmt.Println()

    // Menghapus array pada indeks tertentu
    var index int
    fmt.Print("Masukkan indeks elemen yang akan dihapus:
")
    fmt.Scan(&index)
    if index >= 0 && index < len(array_145) {
        array_145 = append(array_145[:index],
array_145[index+1:]...)
        fmt.Println("Isi array setelah dihapus:",
array_145)
    } else {
        fmt.Println("Indeks tidak valid!")
    }

    // Menampilkan rata-rata array
    var sum int
    for _, v := range array_145 {
        sum += v
    }
    rataRata := float64(sum) / float64(len(array_145))
    fmt.Printf("f. Rata-rata elemen array: %.2f\n",
rataRata)

    // Menampilkan standar deviasi elemen array
    var variance float64
    for _, v := range array_145 {
        variance += math.Pow(float64(v)-rataRata, 2)
    }
    stdDev := math.Sqrt(variance /
float64(len(array_145)))
    fmt.Printf("Standar deviasi elemen array: %.2f\n",
stdDev)

    // Menampilkan frekuensi suatu bilangan dalam array
    var target int
    fmt.Print("Masukkan bilangan untuk menghitung
frekuensinya: ")
    fmt.Scan(&target)

```

```

    frekuensi := 0
    for _, v := range array_145 {
        if v == target {
            frekuensi++
        }
    }
    fmt.Printf("Frekuensi bilangan %d: %d kali\n",
target, frekuensi)
}

```

Screenshoot Output

```

PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Unguided2.go"
Masukkan jumlah elemen array: 5
Masukkan elemen array:
Elemen ke-0: 1 2 3 4 5
Elemen ke-1: Elemen ke-2: Elemen ke-3: Elemen ke-4: Seluruh isi array: [1 2 3 4 5]
Elemen dengan indeks ganjil: 2 4
Elemen dengan indeks genap: 1 3 5
Masukkan bilangan x untuk kelipatan indeks: 3
Elemen dengan indeks kelipatan 3: 1 4
Masukkan indeks elemen yang akan dihapus: 2
Isi array setelah dihapus: [1 2 4 5]
f. Rata-rata elemen array: 3.00
Standar deviasi elemen array: 1.58
Masukkan bilangan untuk menghitung frekuensinya: 1
Frekuensi bilangan 1: 1 kali
PS D:\praktikum alpro>

```

Deskripsi Program

Program di atas dapat melakukan banyak hal pada array, seperti menampilkan isi array, menghitung rata-rata dan standar deviasi, menghitung frekuensi, dan menampilkan elemen dengan indeks ganjil atau genap, kelipatan, atau dihapus. Selanjutnya, program meminta jumlah elemen dalam array dari pengguna, dan kemudian mengisi array dengan input pengguna. Terakhir, program menghitung berapa kali angka tertentu muncul dalam array. Output program menampilkan hasil dari setiap operasi berdasarkan input pengguna.

3. Sourcecode

```

package main

import (
    "fmt"
    "strings"
)

```

```

type Pertandingan_145 struct {
    clubA string
    clubB string
    scoreA int
    scoreB int
}

func main() {
    var clubA, clubB string
    var matches []Pertandingan_145
    var winners []string

    fmt.Print("Masukkan nama Klub A: ")
    fmt.Scanln(&clubA)
    fmt.Print("Masukkan nama Klub B: ")
    fmt.Scanln(&clubB)

    round := 1
    for {
        var scoreA, scoreB int
        fmt.Printf("Pertandingan %d - skor: ", round)
        fmt.Scan(&scoreA, &scoreB)

        if scoreA < 0 || scoreB < 0 {
            fmt.Println("Input skor selesai.")
            break
        }

        matches = append(matches,
Pertandingan_145{clubA, clubB, scoreA, scoreB})

        if scoreA > scoreB {
            winners = append(winners, clubA)
        } else if scoreB > scoreA {
            winners = append(winners, clubB)
        }

        round++
    }

    fmt.Println("\nHasil Pertandingan:")
    for i, match := range matches {
        fmt.Printf("Pertandingan %d: %s %d - %d %s\n",
i+1, match.clubA, match.scoreA, match.scoreB,
match.clubB)
    }

    fmt.Println("\nDaftar Klub yang Menang:")
    for _, winner := range winners {
        fmt.Println(winner)
    }

    countA, countB := 0, 0
    for _, winner := range winners {
        if strings.EqualFold(winner, clubA) {

```

```

        countA++
    } else if strings.EqualFold(winner, clubB) {
        countB++
    }
}

fmt.Printf("\nJumlah kemenangan:\n%s: %d\n%s: %d\n",
clubA, countA, clubB, countB)
}

```

Screenshoot Output

```

PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Unguided3.go"
Masukkan nama Klub A: Mu

```

```

Masukkan nama Klub B: Inter
Pertandingan 1 - skor:
2 0
Pertandingan 2 - skor: 1 2
Pertandingan 3 - skor: 2 2
Pertandingan 4 - skor: 0 1
Pertandingan 5 - skor: 3 2
Pertandingan 6 - skor: 1 0
Pertandingan 7 - skor: 5 2
Pertandingan 8 - skor: 2 3
Pertandingan 9 - skor: -1 2
Input skor selesai.

```

```

Hasil Pertandingan:
Pertandingan 1: Mu 2 - 0 Inter
Pertandingan 2: Mu 1 - 2 Inter
Pertandingan 3: Mu 2 - 2 Inter
Pertandingan 4: Mu 0 - 1 Inter
Pertandingan 5: Mu 3 - 2 Inter
Pertandingan 6: Mu 1 - 0 Inter
Pertandingan 7: Mu 5 - 2 Inter
Pertandingan 8: Mu 2 - 3 Inter

```

Daftar Klub yang Menang:

```

Mu
Inter
Inter
Mu
Mu
Mu
Inter

```

Jumlah kemenangan:

Mu: 4

Inter: 3

```

PS D:\praktikum alpro> █

```

Deskripsi Program

Program ini mencatat hasil pertandingan antara dua klub sepak bola, mengetahui pemenang, dan menampilkan statistik kemenangan masing-masing klub. Mula-mula, program meminta user memasukkan nama dua klub: Klub A dan Klub B. Kemudian mereka dapat memasukkan skor pertandingan, masing-masing Klub A dan Klub B. Setelah input pertandingan selesai, program menampilkan hasil semua pertandingan yang sudah dimainkan, daftar pemenang setiap pertandingan, dan jumlah total kemenangan dari masing-masing klub. Setiap skor yang dimasukkan ke dalam daftar pertandingan (slice struct) dicatat, dan pemenang dari setiap pertandingan ditentukan berdasarkan skor tertinggi. Untuk membantu user memahami hasil kompetisi, program ini memberikan output yang jelas.

4. Sourcecode

```
package main

import (
    "fmt"
)

const NMAX = 127

type tabel_145 struct {
    tab [NMAX]rune
}

func isiArray(t *tabel_145, n *int) {
    var input string
    fmt.Println("Masukkan teks (akhiri dengan titik '.') : ")
    fmt.Scanln(&input)

    for i, char := range input {
        if char == '.' || i >= NMAX {
            break
        }
        t.tab[i] = char
        *n++
    }
}
```

```

func cetakArray(t tabel_145, n int) {
    for i := 0; i < n; i++ {
        fmt.Printf("%c", t.tab[i])
    }
    fmt.Println()
}

func balikanArray(t tabel_145, n int) tabel_145 {
    var reversed tabel_145
    for i := 0; i < n; i++ {
        reversed.tab[i] = t.tab[n-i-1]
    }
    return reversed
}

func palindrom(t tabel_145, n int) bool {
    for i := 0; i < n/2; i++ {
        if t.tab[i] != t.tab[n-i-1] {
            return false
        }
    }
    return true
}

func main() {
    var tab tabel_145
    var n int

    isiArray(&tab, &n)

    fmt.Print("Teks: ")
    cetakArray(tab, n)

    reversedTab := balikanArray(tab, n)
    fmt.Print("Reverse teks: ")
    cetakArray(reversedTab, n)

    if palindrom(tab, n) {
        fmt.Println("Palindrom: true")
    } else {
        fmt.Println("Palindrom: false")
    }
}

```

Screenshot Output

```
PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Unguided4.go"
Masukkan teks (akhiri dengan titik '.'):
KATAK.
Teks: KATAK
Reverse teks: KATAK
Palindrom: true
PS D:\praktikum alpro> go run "d:\praktikum alpro\Modul 7\Unguided4.go"
Masukkan teks (akhiri dengan titik '.'):
SENANG.
Teks: SENANG
Reverse teks: GNANES
Palindrom: false
PS D:\praktikum alpro> █
```

Deskripsi Program

Program di atas adalah program yang berfungsi untuk membaca teks dari pengguna, membalik urutan teks tersebut, dan memeriksa apakah teks tersebut adalah palindrom. Program dimulai dengan meminta pengguna memasukkan teks, yang harus diakhiri dengan tanda titik (`. `). Fungsi "isiArray" digunakan untuk mengisi array dengan teks yang dimasukkan, "balikanArray" digunakan untuk membalik urutan karakter dalam array, dan "palindrom" digunakan untuk mengetahui apakah teks tersebut adalah palindrom (teks yang sama jika dibaca dari depan maupun belakang). Program mencetak teks asli, teks yang telah dibalik, dan hasil untuk mengetahui apakah teks adalah palindrom setelah memproses input. Program ini memastikan bahwa setiap tugas dilakukan dengan efisien dan bahwa outputnya mudah dipahami.