

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 7  
STRUCT DAN ARRAY**



**Disusun Oleh :**  
**Mohammad Alfian Naraya / 2311102170**  
**S1-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**

### **A. ARRAY**

Array adalah tipe data dasar yang digunakan untuk menyimpan sekumpulan elemen dengan tipe data yang sama. Di bahasa Go, array memiliki ukuran tetap yang ditetapkan saat deklarasi.

Karakter Array:

#### **1. Ukuran Standar:**

Dibandingkan dengan tipe data lain seperti slice, ukuran array di Go ditentukan saat deklarasi dan tidak dapat diubah selama runtime.

#### **2. Data homogen terdiri dari:**

Semua elemen dalam array harus memiliki tipe data yang sama—misalnya, int, string, atau float64.

#### **3. Index dimulai dengan zero:**

Element awal array berada di indeks 0, elemen kedua berada di indeks 1, dan seterusnya.

#### **4. Akses ke elemen:**

Dengan menggunakan notasi `array[index]`, Anda dapat mengakses elemen dalam array dengan menggunakan indeks.

Syarat Array:

Ukuran dan tipe data array dapat dideklarasikan. Dalam Go, ada beberapa cara untuk mendeklarasikan dan menginisialisasi array.

1. Deklarasi Array Kosong
2. Inisialisasi Array dengan Nilai
3. Deklarasi Array dengan Panjang Otomatis
4. Penggunaan Array
5. Array Multidimensi

### **B. STRUCT**

Struktur sering digunakan untuk menunjukkan objek atau entitas dengan beberapa atribut. Ini adalah tipe data komposit yang memungkinkan kita mengelompokkan variabel dengan tipe yang berbeda dalam satu entitas.

Karakter Struktur:

#### 1. Tipe Data Heterogen:

Dalam struct, field dapat memiliki berbagai tipe data, yang memungkinkan penggunaan tipe data yang lebih kompleks.

#### 2. Nama Field:

Setiap field struct memiliki nama sendiri.

#### 3. Notasi Titik:

Kita menggunakan notasi titik (.) untuk mengakses atau mengubah area.

#### 4. Komposisi Struct:

Struct dapat menggabungkan struct lain karena mendukung komposisi.

#### Deklarasi Struktur:

Deklarasi struktur dimulai dengan kata kunci type, nama struktur, dan definisi field dalam kurung kurawal.

## **II. GUIDED**

### **1. Guided 1 Soal Studi Case**

Program sederhana untuk menghitung lama waktu parkir berdasarkan waktu kedatangan dan waktu pulang

## 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 // Konversi ke detik    dPulang
    =      wPulang.detik      +      wPulang.menit*60      +
    wPulang.jam*3600 // detik
            lParkir      =      dPulang
dParkir
//detik dari pulang-datang

    durasi.jam = lParkir / 3600    durasi.menit
    = lParkir % 3600 / 60    durasi.detik = lParkir
    % 3600 % 60 //17    fmt.Printf("Lama Parkir :
    %d jam %d menit %d detik", durasi.jam,
    durasi.menit, durasi.detik)
}
```

## Screenshoot Output

```
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\guided1.go"
7 3 0
10 45 15
Lama Parkir : 3 jam 42 menit 15 detik
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> █
```

### Deskripsi Program :

Program di atas merupakan program Go yang menghitung durasi parkir kendaraan berdasarkan waktu masuk dan waktu keluar. Program menggunakan struktur data `waktu` yang memiliki tiga komponen: jam, menit, dan detik. Program menerima input waktu parkir (masuk) dan waktu pulang (keluar) dari pengguna, kemudian mengkonversi kedua waktu tersebut ke dalam satuan detik untuk memudahkan perhitungan selisih waktu. Setelah mendapatkan selisih waktu dalam detik, program mengkonversi kembali ke format jam, menit, dan detik yang lebih mudah dibaca, lalu menampilkan hasil durasi parkir dalam format "Lama Parkir: X jam X menit X detik". Program ini berguna untuk menghitung berapa lama sebuah kendaraan parkir di suatu tempat dengan tepat hingga satuan detik.

## 2. Guided 2 Soal Studi Case

Program sederhana untuk validasi duplikasi nama pada daftar teman

### Sourcecode

```
package main
import (
    "fmt"
)

func sudahAda(daftarTeman []string, nama string)
bool {
    for _, teman := range daftarTeman {
        if teman == nama {
            return
            true
        }
    }
    return false
}
```

```

func main() {
    daftarTeman := []string{"Andi", "Budi",
"Cici"}
    namaBaru := []string{"Dewi", "Budi", "Eka"}

    for _, nama := range namaBaru {
        if !sudahAda(daftarTeman, nama) {
            daftarTeman = append(daftarTeman, nama)
        } else {
            fmt.Println("Nama", nama, "sudah ada
dalam daftar.")
        }
    }
    fmt.Println("Daftar Teman:", daftarTeman) }

```

## Screenshoot Output

```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\guided2.go"
Nama Budi sudah ada dalam daftar.
Daftar Teman: [Andi Budi Cici Dewi Eka]
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7>

```

## Deskripsi Program :

Program di atas adalah program Go yang mengelola daftar nama teman dengan mencegah duplikasi nama. Program memiliki fungsi `sudahAda` yang bertugas memeriksa apakah sebuah nama sudah terdapat dalam daftar teman yang ada. Program dimulai dengan daftar awal yang berisi tiga nama ("Andi", "Budi", "Cici"), kemudian mencoba menambahkan tiga nama baru ("Dewi", "Budi", "Eka"). Saat menambahkan nama baru, program memeriksa setiap nama menggunakan fungsi `sudahAda` - jika nama belum ada, maka nama tersebut akan ditambahkan ke daftar, namun jika nama sudah ada (seperti "Budi"), program akan menampilkan pesan bahwa nama tersebut sudah ada dalam daftar. Pada akhirnya, program menampilkan daftar teman yang sudah diperbarui,

yang hanya akan menambahkan nama-nama yang belum ada sebelumnya.

### 3. Guided 3 Soal Studi Case

Program sederhana untuk menampilkan daftar harga buah

#### Sourcecode

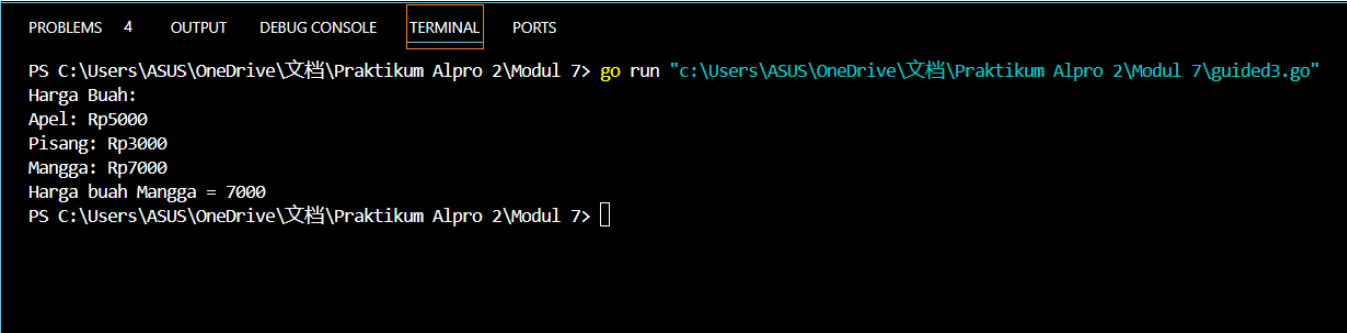
```
package main

import "fmt"

func main() {
    hargaBuah := map[string]int{
        "Apel": 5000,
        "Pisang": 3000,
        "Mangga": 7000,
    }
    fmt.Println("Harga Buah:")
    for buah, harga := range hargaBuah {
        fmt.Printf("%s: Rp%d\n", buah, harga)
    }

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

#### Screenshot Output



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\guided3.go"
Harga Buah:
Apel: Rp5000
Pisang: Rp3000
Mangga: Rp7000
Harga buah Mangga = 7000
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> █
```

#### Deskripsi Program :

Program di atas adalah program Go yang menggunakan struktur data map untuk menyimpan daftar harga buah-buahan. Program membuat sebuah map



bernama `hargaBuah` yang memetakan nama buah (string) dengan harganya (integer), di mana terdapat tiga jenis buah yaitu Apel seharga Rp5000, Pisang seharga Rp3000, dan Mangga seharga Rp7000. Program kemudian menampilkan seluruh daftar harga buah menggunakan perulangan `for range` yang mengiterasi setiap pasangan key-value dalam map, dan mencetak nama buah beserta harganya. Terakhir, program menampilkan harga spesifik untuk buah Mangga dengan mengakses langsung nilai dalam map menggunakan key "Mangga".

### III. UNGUIDED

#### 1. Unguided 1 Soal Studi Case

Suatu lingkaran didefinisikan dengan koordinat titik pusat ( $cx, cy$ ) dengan radius  $r$ . Apabila diberikan dua buah lingkaran, maka tentukan posisi sebuah titik sembarang ( $x, y$ ) berdasarkan dua lingkaran tersebut. Gunakan tipe bentukan titik untuk menyimpan koordinat, dan tipe bentukan lingkaran untuk menyimpan titik pusat lingkaran dan radiusnya.

**Masukan** terdiri dari beberapa tiga baris. Baris pertama dan kedua adalah koordinat titik pusat dan radius dari lingkaran 1 dan lingkaran 2, sedangkan baris ketiga adalah koordinat titik sembarang. Asumsi sumbu  $x$  dan  $y$  dari semua titik dan juga radius direpresentasikan dengan bilangan bulat.

**Keluaran** berupa string yang menyatakan posisi titik "Titik di dalam lingkaran 1 dan 2", "Titik di dalam lingkaran 1", "Titik di dalam lingkaran 2", atau "Titik di luar lingkaran 1 dan 2".

No	Masukan	Keluaran
1	1 1 5 8 8 4 2 2	Titik di dalam lingkaran 1
2	1 2 3 4 5 6 7 8	Titik di dalam lingkaran 2
3	5 10 15 -15 4 20 0 0	Titik di dalam lingkaran 1 dan 2
4	1 1 5 8 8 4 15 20	Titik di luar lingkaran 1 dan 2

Sourcecode

```
package main
import (      "bufio"
```

```

    "fmt"
    "math"
    "os"
    "strconv"
    "strings"
)

// Struct untuk menyimpan koordinat
titik type Point struct {      x, y
float64
}

// Struct untuk menyimpan informasi
lingkaran type Circle struct {      center
Point      radius float64
}

// Fungsi untuk menghitung jarak antara dua titik
func distance(p1, p2 Point) float64 {      dx :=
p1.x - p2.x      dy := p1.y - p2.y
return math.Sqrt(dx*dx + dy*dy)
}

// Fungsi untuk mengecek apakah titik berada di
dalam lingkaran
func isPointInCircle(p Point, c Circle) bool {
return distance(p, c.center) <= c.radius
}

// Fungsi untuk menentukan posisi titik terhadap
dua lingkaran
func determinePosition(p Point, c1, c2 Circle)
string {
    inCircle1 := isPointInCircle(p, c1)
    inCircle2 := isPointInCircle(p, c2)

    if inCircle1 && inCircle2 {
        return "Titik di dalam lingkaran 1 dan
2"
    } else if inCircle1 {
        return "Titik di dalam lingkaran 1"
    } else if inCircle2 {
        return "Titik di dalam lingkaran 2"
    }
}

```

```
return "Titik di luar lingkaran 1 dan 2"
```

```

}
func main() {
    scanner := bufio.NewScanner(os.Stdin)

    // Membaca data lingkaran 1
scanner.Scan()
    line1 := strings.Fields(scanner.Text())
x1, _ := strconv.ParseFloat(line1[0], 64)
y1, _ := strconv.ParseFloat(line1[1], 64)
r1, _ := strconv.ParseFloat(line1[2], 64)
circle1 := Circle{Point{x1, y1}, r1}

    // Membaca data lingkaran 2
scanner.Scan()
    line2 := strings.Fields(scanner.Text())
x2, _ := strconv.ParseFloat(line2[0], 64)
y2, _ := strconv.ParseFloat(line2[1], 64)
r2, _ := strconv.ParseFloat(line2[2], 64)
circle2 := Circle{Point{x2, y2}, r2}

    // Membaca koordinat titik yang akan dicek
scanner.Scan()
    line3 := strings.Fields(scanner.Text())
x, _ := strconv.ParseFloat(line3[0], 64)
y, _ := strconv.ParseFloat(line3[1], 64)
point := Point{x, y}

    // Menentukan dan mencetak hasil      result
:= determinePosition(point, circle1, circle2)
    fmt.Println(result)
}

```

## Screenshoot Output

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\unguided1.go"
1 1 5
8 8 4
2 2
Titik di dalam lingkaran 1
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\unguided1.go"
1 2 3
4 5 6
7 8
Titik di dalam lingkaran 2
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7\unguided1.go"
5 10 15
-15 4 20
0 0
Titik di dalam lingkaran 1 dan 2
```

### Deskripsi Program :

Program di atas adalah program Go yang menentukan posisi sebuah titik relatif terhadap dua buah lingkaran. Program menggunakan dua struktur data yaitu `Point` untuk menyimpan koordinat titik (x,y) dan `Circle` untuk menyimpan informasi lingkaran (pusat dan jari-jari). Program ini dilengkapi dengan beberapa fungsi utama: `distance` untuk menghitung jarak antara dua titik menggunakan rumus Pythagoras, `isPointInCircle` untuk mengecek apakah suatu titik berada di dalam lingkaran, dan `determinePosition` untuk menentukan posisi titik terhadap kedua lingkaran. Program menerima input dari pengguna berupa koordinat pusat dan jari-jari untuk dua lingkaran, serta koordinat sebuah titik yang akan dicek posisinya. Setelah melakukan perhitungan, program akan menampilkan output berupa keterangan apakah titik tersebut berada di dalam lingkaran 1, di dalam lingkaran 2, di dalam kedua lingkaran, atau di luar kedua lingkaran.

## 2. Unguided 2 Soal Studi Case

Sebuah array digunakan untuk menampung sekumpulan bilangan bulat. Buatlah program yang digunakan untuk mengisi array tersebut sebanyak N elemen nilai. Asumsikan array memiliki kapasitas penyimpanan data sejumlah elemen tertentu. Program dapat menampilkan beberapa informasi berikut:

- Menampilkan keseluruhan isi dari array.
- Menampilkan elemen-elemen array dengan indeks ganjil saja.
- Menampilkan elemen-elemen array dengan indeks genap saja (asumsi indek ke-0 adalah genap).

- d. Menampilkan elemen-elemen array dengan indeks kelipatan bilangan x. x bisa diperoleh dari masukan pengguna.
- e. Menghapus elemen array pada indeks tertentu, asumsi indeks yang hapus selalu valid. Tampilkan keseluruhan isi dari arraynya, pastikan data yang dihapus tidak tampil
- f. Menampilkan rata-rata dari bilangan yang ada di dalam array.
- g. Menampilkan standar deviasi atau simpangan baku dari bilangan yang ada di dalam array tersebut.
- h. Menampilkan frekuensi dari suatu bilangan tertentu di dalam array yang telah diisi tersebut.

### Sourcecode

```
package main
import (
    "fmt"
    "math"
)

type ArrayOperations struct {
    data []float64    size int
}

// Membuat array baru
func NewArrayOperations(capacity int)
*ArrayOperations {
return &ArrayOperations{
    data: make([]float64, 0, capacity),
    size: 0,
}
}

// Menambah elemen ke array
func (a *ArrayOperations) Add(value float64) {
if a.size < cap(a.data) {
    a.data = append(a.data, value)
    a.size++
}
}

// a. Menampilkan seluruh isi array func
(a *ArrayOperations) DisplayAll() {
    fmt.Println("Seluruh isi array:")
}
```





```

        for i, v := range a.data {
            fmt.Printf("Index %d: %.2f\n", i, v)
        }
    }

    // b. Menampilkan elemen dengan indeks ganjil
    func (a *ArrayOperations) DisplayOddIndices() {
        fmt.Println("Elemen dengan indeks ganjil:")
        for i, v := range a.data {
            if i%2 != 0 {
                fmt.Printf("Index %d: %.2f\n", i, v)
            }
        }
    }

    // c. Menampilkan elemen dengan indeks genap
    func (a *ArrayOperations) DisplayEvenIndices() {
        fmt.Println("Elemen dengan indeks genap:")
        for i, v := range a.data {
            if i%2 == 0 {
                fmt.Printf("Index %d: %.2f\n", i, v)
            }
        }
    }

    // d. Menampilkan elemen dengan indeks kelipatan
    x
    func (a *ArrayOperations)
    DisplayMultipleIndices(x int) {
        fmt.Printf("Elemen dengan indeks kelipatan
        %d:\n", x)
        for i, v := range a.data {
            if i%x == 0 {
                fmt.Printf("Index %d: %.2f\n", i, v)
            }
        }
    }

    // e. Menghapus elemen pada indeks tertentu
    func (a *ArrayOperations) DeleteAtIndex(index int) {
        if index >= 0 && index < a.size {
            a.data = append(a.data[:index],
            a.data[index+1:]...)
            a.size--
        }
    }

```

}



```

// f. Menghitung rata-rata
func (a *ArrayOperations) CalculateAverage()
float64 {
    if a.size == 0 {
return 0
    }
    sum
:= 0.0
    for _, v := range a.data {
sum += v
    }
    return sum / float64(a.size)
}

// g. Menghitung standar deviasi
func (a *ArrayOperations)
CalculateStandardDeviation() float64 {
if a.size < 2 {
    return 0
}
    mean := a.CalculateAverage()
sumSquaredDiff := 0.0
    for _,
v := range a.data {
        diff
:= v - mean
        sumSquaredDiff += diff * diff
    }
    variance := sumSquaredDiff /
float64(a.size-
1)
    return math.Sqrt(variance)
}

// h. Menghitung frekuensi suatu bilangan
func (a *ArrayOperations)
CalculateFrequency(number float64) int {
frequency := 0
    for _, v := range a.data {
if v == number {
frequency++
        }
    }
    return frequency
}

func main() {

```

```
// Membuat array dengan kapasitas  
10    arr := NewArrayOperations(10)
```



```

        // Menambahkan beberapa nilai
arr.Add(1.0)      arr.Add(2.0)
arr.Add(3.0)      arr.Add(4.0)
arr.Add(5.0)      arr.Add(2.0)
arr.Add(3.0)

        // Demonstrasi semua operasi
fmt.Println("\n=== Demonstrasi Operasi Array
===")

        // a. Menampilkan semua elemen
arr.DisplayAll()

        fmt.Println()
        // b. Menampilkan indeks ganjil
arr.DisplayOddIndices()

        fmt.Println()
        // c. Menampilkan indeks genap
arr.DisplayEvenIndices()

        fmt.Println()
        // d. Menampilkan kelipatan 2
arr.DisplayMultipleIndices(2)

        fmt.Println()
        // e. Menghapus elemen index 2
fmt.Println("Menghapus elemen index 2:")
arr.DeleteAtIndex(2)      arr.DisplayAll()

        fmt.Println()
        // f. Menampilkan rata-rata
                                fmt.Printf("Rata-rata:
%.2f\n",
arr.CalculateAverage())

        // g. Menampilkan standar deviasi
fmt.Printf("Standar deviasi: %.2f\n",
arr.CalculateStandardDeviation())

        // h. Menampilkan frekuensi angka 2.0
fmt.Printf("Frekuensi angka 2.0: %d\n",
arr.CalculateFrequency(2.0))

```

```
}
```

### Screenshot Output

```
PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Elemen dengan indeks ganjil:
Index 1: 2.00
Index 3: 4.00
Index 5: 2.00

Elemen dengan indeks genap:
Index 0: 1.00
Index 2: 3.00
Index 4: 5.00
Index 6: 3.00

Elemen dengan indeks kelipatan 2:
Index 0: 1.00
Index 2: 3.00
Index 4: 5.00
Index 6: 3.00

Menghapus elemen index 2:
Seluruh isi array:
Index 0: 1.00
Index 1: 2.00
Index 2: 4.00
Index 3: 5.00
Index 4: 2.00
Index 5: 3.00

Rata-rata: 2.83
Standar deviasi: 1.47
Frekuensi angka 2.0: 2
```

### Deskripsi Program :

Program di atas adalah implementasi Go yang membuat sistem pengelolaan array dengan berbagai operasi matematika dan manipulasi data menggunakan struktur data `ArrayOperations`. Program ini menyediakan berbagai fungsi untuk memanipulasi dan menganalisis array floating-point, termasuk: menambah elemen baru (`Add`), menampilkan seluruh isi array (`DisplayAll`), menampilkan elemen dengan indeks ganjil (`DisplayOddIndices`) dan genap (`DisplayEvenIndices`), menampilkan elemen dengan indeks kelipatan tertentu (`DisplayMultipleIndices`), menghapus elemen pada indeks tertentu (`DeleteAtIndex`), menghitung



rata-rata (`CalculateAverage`), menghitung standar deviasi (`CalculateStandardDeviation`), dan menghitung frekuensi kemunculan suatu bilangan (`CalculateFrequency`). Program menggunakan konsep struct dan method untuk mengorganisasi kode, dan dalam fungsi `main`, program mendemonstrasikan penggunaan semua operasi tersebut dengan membuat array contoh dan menjalankan setiap operasi secara berurutan.

### 3. Unguided 3 Soal Studi Case

Sebuah program digunakan untuk menyimpan dan menampilkan nama-nama klub yang memenangkan pertandingan bola pada suatu grup pertandingan. Buatlah program yang digunakan untuk merekap skor pertandingan bola 2 buah klub bola yang berlaga. Pertama-tama program meminta masukan nama-nama klub yang bertanding, kemudian program meminta masukan skor hasil pertandingan kedua klub tersebut. Yang disimpan dalam array adalah nama-nama klub yang menang saja. Proses input skor berhenti ketika skor salah satu atau kedua klub tidak valid (negatif). Di akhir program, tampilkan daftar klub yang memenangkan pertandingan. Perhatikan sesi interaksi pada contoh berikut ini (teks bergaris bawah adalah input/read)

```
Klub A : MU
Klub B : Inter
Pertandingan 1 : 2 0 // MU = 2 sedangkan Inter = 0
Pertandingan 2 : 1 2
Pertandingan 3 : 2 2
Pertandingan 4 : 0 1
Pertandingan 5 : 3 2
Pertandingan 6 : 1 0
Pertandingan 7 : 5 2
Pertandingan 8 : 2 3
Pertandingan 9 : -1 2
Hasil 1 : MU
Hasil 2 : Inter
Hasil 3 : Draw
Hasil 4 : Inter
Hasil 5 : MU
Hasil 6 : MU
Hasil 7 : MU
Hasil 8 : Inter
Pertandingan selesai
```

### Sourcecode

```
package main
import (      "bufio"
    "fmt"
    "os"
    "strings"
)
```

```

func main() {
    scanner := bufio.NewScanner(os.Stdin)
    var winners []string

    // Input nama klub
    fmt.Print("Klub A : ")
    scanner.Scan()
    clubA := scanner.Text()

    fmt.Print("Klub B : ")
    scanner.Scan()
    clubB := scanner.Text()

    // Proses input skor pertandingan
    matchNumber := 1    for {
        fmt.Printf("Pertandingan %d : ",
matchNumber)          scanner.Scan()
        input := scanner.Text()

        // Split input menjadi dua skor
        scores := strings.Split(input, " ")
        if len(scores) != 2 {                continue
        }

        // Convert string ke integer
        var scoreA, scoreB int
        _, err1 := fmt.Sscanf(scores[0], "%d",
&scoreA)
        _, err2 := fmt.Sscanf(scores[1], "%d",
&scoreB)

        // Cek apakah skor valid
        if err1 != nil || err2 != nil || scoreA
< 0 || scoreB < 0 {                fmt.Printf("\nHasil
pertandingan:\n")                for i, winner :=
range winners {                    fmt.Printf("Hasil
%d : %s\n", i+1, winner)                }
        fmt.Println("Pertandingan selesai")
        break
    }
}

```

```

        // Tentukan pemenang
var winner string
        if
scoreA > scoreB {
winner = clubA
        } else if scoreB > scoreA {
winner = clubB
        } else {
            winner = "Draw"
        }

        // Tambahkan pemenang ke array
winners = append(winners, winner)
matchNumber++
    }
}

```

### Screenshot Output

```

PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "c:\Users\
Klub A : MU
Klub B : Inter
Pertandingan 1 : 2 0
Pertandingan 2 : 1 2
Pertandingan 3 : 2 2
Pertandingan 4 : 0 1
Pertandingan 5 : 3 2
Pertandingan 6 : 1 0
Pertandingan 7 : 5 2
Pertandingan 8 : 2 3
Pertandingan 9 : -1 2

Hasil pertandingan:
Hasil 1 : MU
Hasil 2 : Inter
Hasil 3 : Draw
Hasil 4 : Inter
Hasil 5 : MU
Hasil 6 : MU
Hasil 7 : MU
Hasil 8 : Inter
Pertandingan selesai

```

### Deskripsi Program :

Program di atas adalah program Go yang mencatat dan mengelola hasil pertandingan antara dua klub olahraga. Program dimulai dengan meminta input nama kedua klub (Klub A dan Klub B), kemudian secara berulang meminta input skor pertandingan dalam format "skor\_A skor\_B". Program akan terus meminta input skor pertandingan hingga pengguna memasukkan input yang tidak valid (seperti skor negatif atau format yang salah). Setiap kali skor dimasukkan, program menentukan pemenang berdasarkan skor (klub dengan skor lebih tinggi menang, atau "Draw" jika skor sama) dan menyimpannya dalam slice `winners`. Ketika input tidak valid diterima, program akan menampilkan seluruh hasil pertandingan yang telah dicatat sebelumnya dan mengakhiri program dengan pesan "Pertandingan selesai".

### 1. Unguided 4 Soal Studi Case

Sebuah array digunakan untuk menampung sekumpulan karakter, Anda diminta untuk membuat sebuah subprogram untuk melakukan membalikkan urutan isi array dan memeriksa apakah membentuk palindrom. Lengkapi potongan algoritma berikut ini!

```
package main
import "fmt"
const NMAX int = 127
type tabel [NMAX]rune
    tab : tabel
    m : integer

func isiArray(t *tabel, n *int)
/*I.S. Data tersedia dalam piranti masukan
  F.S. Array t berisi sejumlah n karakter yang dimasukkan user,
  Proses input selama karakter bukanlah TITIK dan n <= NMAX */

func cetakArray(t tabel, n int)
/*I.S. Terdefinisi array t yang berisi sejumlah n karakter
  F.S. n karakter dalam array muncul di layar */

func balikanArray(t *tabel, n int)
/*I.S. Terdefinisi array t yang berisi sejumlah n karakter
  F.S. Urutan isi array t terbalik */

func main(){
    var tab tabel
    var m int
    // si array tab dengan memanggil prosedur isiArray

    // Balikian isi array tab dengan memanggil balikanArray

    // Cetak is array tab
}
```

Perhatikan sesi interaksi pada contoh berikut ini (teks bergaris bawah adalah Input/read)

Teks	: <b>S E N A N G .</b>
Reverse teks	: G N A N E S
Teks	: <b>K A T A K .</b>
Reverse teks	: K A T A K

Modifikasi program tersebut dengan menambahkan fungsi palindrom. Tambahkan instruksi untuk memanggil fungsi tersebut dan menampilkan hasilnya pada program utama.

**\*Palindrom adalah teks yang dibaca dari awal atau akhir adalah sama, contoh: KATAK, APA, KASUR\_RUSAK.**

```
func palindrom(t tabel, n int) bool
/* Mengembalikan true apabila susunan karakter di dalam t membentuk palindrom,
dan false apabila sebaliknya. Petunjuk: Manfaatkan prosedur balikanArray */
```

Perhatikan sesi interaksi pada contoh berikut ini (teks bergaris bawah adalah Input/read)

Teks	: <b>K A T A K</b>
Palindrom	? true
Teks	: <b>S E N A N G</b>
Palindrom	? false

### Sourcecode

```
package main

import (
    "bufio"
    "fmt"
    "os"
    "strings"
)

const NMAX int = 127

type tabel [NMAX]rune

// Fungsi untuk mengisi array dengan
karakter func isiArray(t *tabel, n *int) {
    scanner := bufio.NewScanner(os.Stdin)
    fmt.Print("Teks : ")      scanner.Scan()
    text := scanner.Text()

    // Menghilangkan spasi dan tanda baca
    text = strings.ReplaceAll(text, " ", "")
    text = strings.ReplaceAll(text, ".", "")
    text = strings.ReplaceAll(text, ",", "")

    *n = 0
    // Mengisi array dengan karakter dari input
```

```

        for _, char := range text {
            if *n < NMAX {
                t[*n] = char
                *n++
            }
        }
    }

    // Fungsi untuk mencetak isi array
    func cetakArray(t tabel, n int) {
        fmt.Print("Reverse teks : ")      for
        i := 0; i < n; i++ {
            fmt.Printf("%c", t[i])
        }
        fmt.Println()
    }

    // Fungsi untuk membalikkan array func
    balikArray(t *tabel, n int) {
        for i := 0; i < n/2; i++ {
            t[i], t[n-1-i] = t[n-1-i], t[i]
        }
    }

    // Fungsi untuk mengecek apakah array membentuk
    palindrom
    func palindrom(t tabel, n int) bool {
        // Membuat salinan array untuk dibalik
        var temp tabel      copy(temp[:], t[:n])

        // Balik array salinan
        for i := 0; i < n/2; i++ {
            temp[i], temp[n-1-i] = temp[n-1-i],
temp[i]
        }

        // Bandingkan array asli dengan array yang
        sudah dibalik
        for i := 0; i < n; i++ {
            if t[i] != temp[i] {
                return false
            }
        }

        return true
    }

```

```

}
func main() {
var tab tabel
var m int

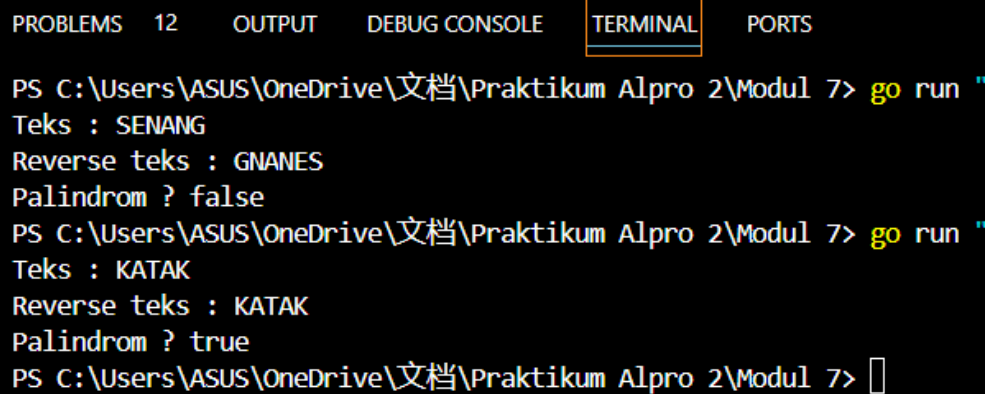
    // Input dan proses array
isiArray(&tab, &m)

    // Tampilkan array yang dibalik
var reversed tabel
copy(reversed[:], tab[:m])
balikanArray(&reversed, m)
cetakArray(reversed, m)

    // Cek palindrom
fmt.Print("Palindrom ? ")      if
palindrom(tab, m) {
fmt.Println("true")
} else {
    fmt.Println("false")
}
}

```

### Screenshot Output



```

PROBLEMS  12  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "
Teks : SENANG
Reverse teks : GNANES
Palindrom ? false
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> go run "
Teks : KATAK
Reverse teks : KATAK
Palindrom ? true
PS C:\Users\ASUS\OneDrive\文档\Praktikum Alpro 2\Modul 7> 

```



### **Deskripsi Program :**

Program di atas adalah program Go yang dirancang untuk memproses teks dan menganalisis apakah teks tersebut merupakan palindrom (kata atau kalimat yang dibaca sama baik dari depan maupun belakang). Program menggunakan array bertipe ``rune`` dengan kapasitas maksimal 127 karakter dan memiliki beberapa fungsi utama: ``isiArray`` untuk menerima input teks dan menyimpannya dalam array (sambil menghilangkan spasi dan tanda baca), ``cetakArray`` untuk menampilkan isi array, ``balikanArray`` untuk membalikkan urutan karakter dalam array, dan ``palindrom`` untuk mengecek apakah teks tersebut merupakan palindrom dengan cara membandingkan array asli dengan array yang sudah dibalik. Dalam fungsi ``main``, program menerima input teks dari pengguna, menampilkan teks tersebut dalam bentuk terbalik, dan memberikan output berupa ``true`` jika teks tersebut adalah palindrom atau ``false`` jika bukan.

## **IV. KESIMPULAN**

Pembelajaran tentang struct dan array merupakan aspek penting dalam pemrograman yang membantu dalam pengelolaan data secara efisien. Struct, atau struktur, memungkinkan pengembang untuk mengelompokkan berbagai tipe data yang berbeda menjadi satu kesatuan yang terorganisir, sehingga memudahkan representasi objek yang kompleks. Sementara itu, array menyediakan cara untuk menyimpan sekumpulan data dengan tipe yang sama dalam urutan tertentu, memungkinkan akses dan manipulasi data secara cepat. Keduanya saling melengkapi; struct memberikan fleksibilitas dalam mendefinisikan tipe data yang lebih kompleks, sedangkan array menawarkan cara untuk mengelola koleksi data secara sistematis. Dengan memahami dan menguasai kedua konsep ini, programmer dapat meningkatkan kemampuan dalam merancang program yang lebih terstruktur dan efisien, serta memecahkan masalah dengan lebih baik.

## **V. REFERENSI**

- [1] Modul 7 Praktikum Algoritma 2
- [2] A. A. A. Donovan and B. W. Kernighan, *The Go Programming Language*. Boston, MA: Addison-Wesley, 2015.