

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VII
STRUCK & ARRAY**



Disusun Oleh:
Bayu Kuncoro Adi / 2311102031
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. Tipe Bentuk

Tipe bentuk memungkinkan pemrograman untuk mendefinisikan suatu tipe data baru pada suatu bahasa pemrograman. Tipe bentuk ini dapat dibedakan atas dua jenis, yaitu Alias dan Struct.

1. Alias (Type)

Bahasa pemrograman pada umumnya mengizinkan pemrograman untuk mengubah nama suatu tipe data dengan nama baru yang lebih ringkas dan familiar. Sebagai contoh “**Integer**” dapat dirubah dengan nama alias “**bilangan**”. Caranya dengan menggunakan kata kunci “**type**”.

	Notasi Algoritma	Notasi dalam bahasa Go
1	kamus	package main
2	type <nama alias> <tipe data>	
3		type <nama alias> <tipe data>
4	algoritma	
5	...	func main(){
6	
7		}

Sebagai contoh perhatikan program Go berikut beserta hasil eksekusinya!

```
1 package main
2 import "fmt"
3 type bilangan int
4 type pecahan float64
5 func main(){
6     var a,b bilangan
7     var hasil pecahan
8     a = 9
9     b = 5
10    hasil = pecahan(a) / pecahan(b)
11    fmt.Println(hasil)
12 }
```

```
E:\DEV\GO>go build Demo.go
E:\DEV\GO> Demo.exe
1.8
```

2. Struct atau Record

Structure memungkinkan pemrograman untuk mengelompokkan beberapa data atau nilai yang memiliki relasi atau keterkaitan tertentu menjadi suatu kesatuan. Masing-masing nilai tersimpan dalam field dari structure tersebut.

	Notasi Algoritma	Notasi dalam bahasa Go
1	kamus	package main
2	type <nama struct> <	type <nama struct> struct {
3	<field 1> <type data>	<field 1> <type data>
4	<field 2> <type data>	<field 2> <type data>
5	<field 3> <type data>	<field 3> <type data>
6	>	}
7		

Berbeda dengan bahasa pemrograman lain, kesamaan tipe dari dua variabel berjenis structure bukan karena namanya tetapi karena strukturnya. Dua variabel dengan nama-nama field dan tipe field yang sama (dan dalam urutan yang sama) dianggap mempunyai tipe yang sama. Tentunya akan lebih memudahkan jika structure tersebut didefinisikan sebagai sebuah tipe baru, sehingga deklarasi structure tidak perlu lagi seluruh field-nya ditulis ulang berkali-kali.

```
1 package main
2 import "fmt"
3 type waktu struct {
4     jam, menit, detik int
5 }
6
7 func main(){
8     var wParkir, wPulang, durasi waktu
9     var dParkir, dPulang, lParkir int
10    fmt.Scan(&wParkir.jam, &wParkir.menit, &wParkir.detik)
11    fmt.Scan(&wPulang.jam, &wPulang.menit, &wPulang.detik)
12    dParkir = wParkir.detik + wParkir.menit*60 + wParkir.jam*3600
13    dPulang = wPulang.detik + wPulang.menit*60 + wPulang.jam*3600
14    lParkir = dPulang - dParkir
15    durasi.jam = lParkir / 3600
16    durasi.menit = lParkir % 3600 / 60
17    durasi.detik = lParkir % 3600 % 60
18    fmt.Printf("Lama parkir: %d jam %d menit %d detik",
19        durasi.jam, durasi.menit, durasi.detik)
20 }
```

```
E:\DEV\GO>go build Demo.go
```

```
E:\DEV\GO> Demo.exe
```

```
7 30 0
```

```
10 45 15
```

```
Lama parkir: 3 iam 15 menit 15 detik
```

B. Array

Array mempunyai ukuran (jumlah elemen) yang tetap (statis) selama eksekusi program, sehingga jumlah elemen array menjadi bagian dari deklarasi variabel dengan tipe array.

	Notasi dalam bahasa Go
1	var (
2	// array arr mempunyai 73 elemen, masing-masing bertipe CircType2
3	arr [73]CircType
4	
5	// array buf dengan 5 elemen, dengan nilai awal 7, 3, 5, 2, dan 11.
6	buf = [5]byte{7, 3, 5, 2, 11}
7	
8	// mhs adalah array dengan 2000 elemen bertipe NewType
9	mhs [2000]NewType
10	
11	// rec adalah array dari array, yaitu matriks, atau array berdimensi-2
12	rec [20][40]float64
13)

Jumlah elemen array dapat diminta dengan fungsi `len` yang tersedia. Sebagai contoh `len(arr)` akan menghasilkan 74 untuk contoh diatas.

Indeks array dimulai dari 0, sehingga indeks arr pada contoh adalah `0..len(arr)-1`

Contoh:

```
1 // Mengganti isi elemen ke-0 dengan nilai dari elemen ke-7
2 arr[0] = arr[7]
3
4 // Mengambil data field x dari elemen ke-i
5 currX = arr[i].center.x
6
7 // Mengambil elemen terakhir
8 n := len(arr)
9 buf := arr[n-1]
```

1. Slice (Array Dinamik)

Array dalam Go juga dapat mempunyai ukuran yang dinamik. (Tidak digunakan di kelas Algoritma Pemrograman). Deklarasi mirip dengan deklarasi array, tetapi jumlah elemen dikosongkan.

```
1 // declaring chop as an empty slice of float64
2 var chop []float64
3
4 // declaring sl01 as a slice
5 var sl01 = []int{ 11, 2, 3, 5, 7, 13 }
```

Sebuah slice dapat diprealokasi menggunakan fungsi built-in **make**

```
1 // Prealokasi 10 elemen untuk sl02 dan sejumlah tempat tambahan
2 var sl02 []int = make([]int, 10, 20)
3
4 // Prealokasi 7 elemen untuk sl03 tanpa tempat tambahan
5 var sl03 []circType = make([]circType, 7)
```

Fungsi built-in `len` dapat digunakan untuk mengetahui ukuran slice. Fungsi lain, `cap`, dapat digunakan untuk mengetahui total tempat yang disediakan untuk slice tersebut.

```
1 // Cetak jumlah elemen dan tempat yang tersedia untuk sl02
2 fmt.Println( len(sl02), cap(sl02) )
```

Fungsi built-in `append` dapat digunakan untuk menambahkan elemen ke suatu slice, dan bila perlu memperbesar tempat untuk slice tersebut.

```
1 /* Append elemen baru, membuat slice baru, dan menyimpan kembali slice baru
   ke variabel semula. Boleh juga disimpan ke variabel lain, sehingga variabel
   semula masih menyimpan slice yang asli. */
2 sl01 = append(sl01, 17)
3 sl01 = append(sl01, 19, 23)
```

Sebuah slice baru juga dapat terbentuk dengan mengambil slice dari suatu array atau slice yang lain.

```
1 // Ambil 3 elemen pertama dari suatu slice atau array
2 sl04 = arr[:4]
3
4 // Ambil beberapa elemen terakhir, dimulai dari indeks 5
5 sl05 = sl01[5:]
6
7 // Salin semua dari slice/array aslinya
8 sl06 = sl05[:]
9
10 // Salin element dari indeks 3 sampai, tapi tidak termasuk, 5.
11 // Jadi dalam contoh hanya 2 elemen sl06[3] dan sl06[4] yang disalin
12 sl07 = sl06[3:5]
```

2. Map

Tipe array lain, sebuah array dinamik. Indeksnya (di sini disebut kunci) tidak harus berbentuk integer. Indeks dapat berasal dari tipe apa saja. Struktur ini disebut map.

```
1 // Deklarasi variabel dct sebagai map bilangan bulat dengan kunci string
2 var dct map[string]int
3
4 // Deklarasi map lain dct1 dari elemen string dengan kunci juga string
5 // Mempunyai nilai awal dct1["john"] = "hi", dct1["anne"] = "darling"
6 var dct1 = map[string]string{ "john":"hi", "anne":"darling" }
7
8 // Deklarasi dan prealokasi tempat untuk map dct2
9 var dct2 map[float64]int = make(map[float64]int, 10)
10
11 // Mengambil nilai yang tersimpan dengan kunci "john"
12 fmt.Println( dct1["john"] )
13
14 // Mengganti nilai yang tersimpan pada kunci "anne", dan
15 // Membuat entri baru dengan kunci "boy"
16 dct1["anne"] = "lovely"
17 dct1["boy"] = "runaround"
18
19 // Menghapus entri dengan kunci "john"
20 delete(dct1, "john")
```

A. GUIDED

1. Guided 1

```
package main

import "fmt"

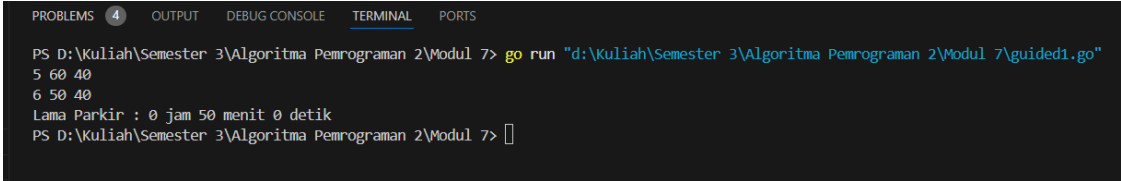
type waktu struct {
    jam, menit, detik int
}

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 Program



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\guided1.go"
5 60 40
6 50 40
Lama Parkir : 0 jam 50 menit 0 detik
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> 
```

Deskripsi dan Cara Kerja Program

Program ini bertujuan untuk menghitung durasi waktu parkir berdasarkan waktu kedatangan dan waktu kepulangan yang diberikan oleh pengguna. Program dimulai dengan mendefinisikan tipe data struktural `'waktu'` yang menyimpan informasi jam, menit, dan detik. Di dalam fungsi `'main'`, program mendeklarasikan variabel untuk waktu parkir (`'wParkir'`), waktu pulang (`'wPulang'`), dan durasi parkir (`'durasi'`). Setelah itu, program membaca input dari pengguna untuk waktu parkir dan waktu pulang, kemudian mengonversi masing-masing waktu tersebut ke dalam satuan detik (`'dParkir'` dan `'dPulang'`). Durasi parkir dalam detik dihitung dengan mengurangi waktu kedatangan dari waktu kepulangan (`'lParkir'`). Selanjutnya, program menghitung durasi parkir dalam satuan jam, menit, dan detik dengan membagi dan mengambil sisa pembagian dari `'lParkir'`. Akhirnya, program menampilkan durasi parkir dalam format yang sesuai. Contoh penggunaan: jika pengguna memasukkan waktu parkir sebagai `'10:30:45'` dan waktu pulang sebagai `'12:45:50'`, program akan menghitung dan menampilkan "Lama Parkir : 2 jam 15 menit 5 detik".

2. Guided 2

```
//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
}

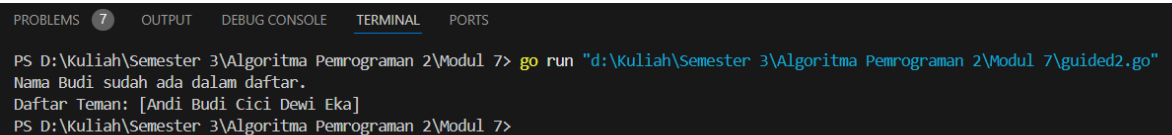
func main() {
    // Slice awal untuk daftar teman dengan beberapa data
    daftarTeman := []string{"Andi", "Budi", "Cici"}

    // 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 Program



The screenshot shows a terminal window with the following content:

```
PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\guided2.go"
Nama Budi sudah ada dalam daftar.
Daftar Teman: [Andi Budi Cici Dewi Eka]
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7>
```

Deskripsi dan Cara Kerja Program

Program ini berfungsi untuk menambah nama-nama baru ke dalam daftar teman dengan pengecekan agar tidak ada nama yang duplikat. Di awal program, slice `daftarTeman` diinisialisasi dengan beberapa nama awal. Fungsi `sudahAda` digunakan untuk memeriksa apakah suatu nama sudah ada di dalam `daftarTeman`. Fungsi ini melakukan iterasi pada `daftarTeman` dan membandingkan setiap elemen dengan nama yang ingin dicek, mengembalikan nilai `true` jika nama tersebut ditemukan, dan `false` jika tidak. Di dalam fungsi `main`, ada slice `namaBaru` yang berisi nama-nama yang akan ditambahkan ke `daftarTeman`. Program melakukan iterasi pada `namaBaru` dan untuk setiap nama, memeriksa apakah nama tersebut sudah ada di `daftarTeman` menggunakan fungsi `sudahAda`. Jika nama belum ada, nama tersebut akan ditambahkan ke `daftarTeman` menggunakan `append`. Jika nama sudah ada, program akan mencetak pesan bahwa nama tersebut sudah ada dalam daftar. Setelah semua nama baru diproses, program menampilkan daftar teman yang telah diperbarui. Contoh penggunaan: jika `daftarTeman` berisi ["Andi", "Budi", "Cici"] dan `namaBaru` berisi ["Dewi", "Budi", "Eka"], program akan menambahkan "Dewi" dan "Eka" ke dalam daftar dan mencetak pesan bahwa "Budi" sudah ada. Hasil akhirnya adalah daftar teman berisi ["Andi", "Budi", "Cici", "Dewi", "Eka"].

3. Guided 3

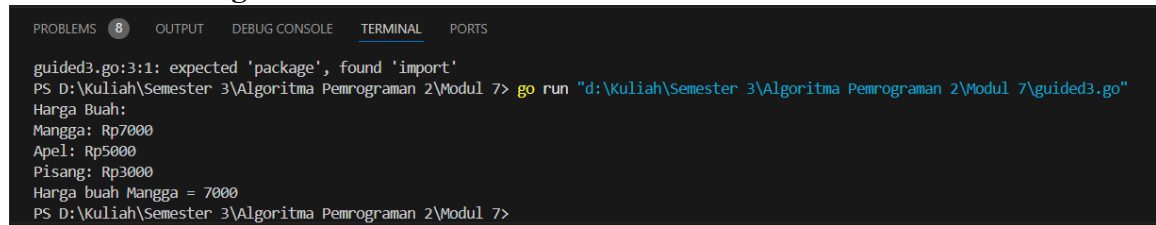
```
//Guided 2 - Map
package main
import (
    "fmt"
)

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 Program



```
PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS
guided3.go:3:1: expected 'package', found 'import'
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\guided3.go"
Harga Buah:
Mangga: Rp7000
Apel: Rp5000
Pisang: Rp3000
Harga buah Mangga = 7000
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7>
```

Deskripsi dan Cara Kerja Program

Program ini mendemonstrasikan penggunaan map di dalam bahasa pemrograman Go untuk menyimpan dan menampilkan harga buah-buahan. Map hargaBuah dibuat dengan kunci berupa nama buah dan nilai berupa harga buah dalam bentuk integer. Tiga buah, yaitu "Apel", "Pisang", dan "Mangga", dimasukkan ke dalam map dengan harga masing-masing Rp5000, Rp3000, dan Rp7000. Program kemudian menampilkan harga dari setiap buah yang ada di dalam map dengan melakukan iterasi menggunakan loop for range, di mana setiap iterasi mencetak nama buah dan harganya dalam format "buah: Rp[harga]". Di akhir, program juga secara khusus mencetak harga buah "Mangga" menggunakan akses langsung dengan kunci map.

A. UNGUIDED

1. 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 radlusnya.

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

Fungsi untuk menghitung jarak titik (a,b) dan (c,d) Dimana rumus jarak adalah:

$$\text{jarak} = \sqrt{(a - c)^2 + (b - d)^2}$$

Dan juga fungsi untuk menentukan posisi sebuah titik sembarang berada di dalam suatu lingkaran atau tidak.

```
function jarak(p, q : titik) -> real
{Mengembalikan jarak antara titik p(x,y) dan titik q(x,y)}

function didalam(c:lingkaran, p:titik) -> boolean
{Mengembalikan true apabila titik p(x,y) berada di dalam lingkaran c yang
memiliki titik pusat (cx,cy) dan radius r}
```

Catatan: Lihat paket math dalam lampiran untuk menggunakan fungsi math.Sqrt() untuk menghitung akar kuadrat.

```
package main

import (
    "fmt"
    "math"
)

type Titik struct {
    x, y int
}

type Lingkaran struct {
    titikPusat Titik
    radius      int
}

// Fungsi untuk menghitung jarak antara dua titik
func jarakAntarTitik(a, b Titik) float64 {
    return math.Sqrt(math.Pow(float64(a.x-b.x), 2) +
math.Pow(float64(a.y-b.y), 2))
}

// Fungsi untuk mengecek apakah titik berada di dalam lingkaran
func cekTitikDiDalam(lingkaran Lingkaran, titik Titik) bool {
    return jarakAntarTitik(lingkaran.titikPusat, titik) <=
float64(lingkaran.radius)
}

func main() {
    var lingkaran1, lingkaran2 Lingkaran
    var titikSembarang Titik

    // Input titik pusat dan radius lingkaran 1
    fmt.Print("Masukkan titik pusat dan radius lingkaran 1: ")
    fmt.Scan(&lingkaran1.titikPusat.x, &lingkaran1.titikPusat.y,
&lingkaran1.radius)

    // Input titik pusat dan radius lingkaran 2
    fmt.Print("Masukkan titik pusat dan radius lingkaran 2: ")
    fmt.Scan(&lingkaran2.titikPusat.x, &lingkaran2.titikPusat.y,
&lingkaran2.radius)

    // Input titik sembarang
    fmt.Print("Masukkan titik sembarang: ")
    fmt.Scan(&titikSembarang.x, &titikSembarang.y)

    // Mengecek posisi titik terhadap lingkaran 1 dan 2
    diDalamLingkaran1 := cekTitikDiDalam(lingkaran1, titikSembarang)
    diDalamLingkaran2 := cekTitikDiDalam(lingkaran2, titikSembarang)

    // Menentukan output berdasarkan posisi titik
    if diDalamLingkaran1 && diDalamLingkaran2 {
        fmt.Println("\nTitik di dalam lingkaran 1 dan 2")
    } else if diDalamLingkaran1 {
```

```

        fmt.Println("\nTitik di dalam lingkaran 1")
    } else if diDalamLingkaran2 {
        fmt.Println("\nTitik di dalam lingkaran 2")
    } else {
        fmt.Println("\nTitik di luar lingkaran 1 dan 2")
    }
}

```

Screenshoot Program

```

PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\unguided1.go"
Masukkan titik pusat dan radius lingkaran 1: 7
1
2
Masukkan titik pusat dan radius lingkaran 2: 3
4
5
Masukkan titik sembarang: 7
5

Titik di dalam lingkaran 2
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> 

```

Deskripsi dan Cara Kerja Program

Program di atas merupakan implementasi dalam bahasa Go untuk mengecek apakah sebuah titik sembarang berada di dalam salah satu atau kedua lingkaran yang diberikan. Program ini memanfaatkan struktur data `Titik` untuk merepresentasikan koordinat titik dan `Lingkaran` untuk merepresentasikan lingkaran dengan titik pusat dan radius. Fungsi `jarakAntarTitik` digunakan untuk menghitung jarak antara dua titik menggunakan rumus jarak Euclidean. Fungsi `cekTitikDiDalam` memeriksa apakah jarak antara titik pusat lingkaran dan titik sembarang lebih kecil atau sama dengan radius lingkaran, menandakan bahwa titik berada di dalam lingkaran. Pada bagian `main`, pengguna diminta memasukkan titik pusat dan radius untuk dua lingkaran serta koordinat titik sembarang. Program kemudian mengevaluasi posisi titik sembarang terhadap kedua lingkaran dan mencetak hasilnya, menunjukkan apakah titik tersebut berada di dalam salah satu atau kedua lingkaran, atau di luar keduanya.

2. Sebuah array digunakan untuk menampung sekumpulan bilangan bulat. Buatlah program yang digunakan untuk mengist array tersebut sebanyak N elemen nilai. Asumsikan array memiliki kapasitas penyimpanan data sejumlah elemen tertentu. Program dapat menampilkan beberapa informasi berikut:
 - a. Menampilkan keseluruhan isi dari array.
 - b. Menampilkan elemen-elemen array dengan indeks ganjil saja.
 - c. 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. informatice lab
 - 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.

```
package main

import (
    "fmt"
    "math"
)

func tampilkanArray(arr []int) {
    fmt.Println("Isi Array:")
    for _, elemen := range arr {
        fmt.Print(elemen, " ")
    }
    fmt.Println()
}

func tampilkanIndeksGanjil(arr []int) {
    fmt.Println("Elemen dengan indeks ganjil:")
    for i := 1; i < len(arr); i += 2 {
        fmt.Print(arr[i], " ")
    }
    fmt.Println()
}

func tampilkanIndeksGenap(arr []int) {
    fmt.Println("Elemen dengan indeks genap:")
    for i := 0; i < len(arr); i += 2 {
        fmt.Print(arr[i], " ")
    }
    fmt.Println()
}

func tampilkanIndeksKelipatanX(arr []int, x int) {
    fmt.Printf("Elemen dengan indeks kelipatan %d:\n", x)
    for i := 0; i < len(arr); i++ {
        if i%x == 0 {
            fmt.Print(arr[i], " ")
        }
    }
    fmt.Println()
}

func hapusElemen(arr []int, indeks int) []int {
    arr = append(arr[:indeks], arr[indeks+1:]...)
    return arr
}

func hitungRataRata(arr []int) float64 {
    var sum int
    for _, elemen := range arr {
```

```

        sum += elemen
    }
    return float64(sum) / float64(len(arr))
}

func hitungStandarDeviasi(arr []int) float64 {
    mean := hitungRataRata(arr)
    var sumSquares float64
    for _, elemen := range arr {
        sumSquares += math.Pow(float64(elemen)-mean, 2)
    }
    variance := sumSquares / float64(len(arr))
    return math.Sqrt(variance)
}

func hitungFrekuensi(arr []int, bilangan int) int {
    var count int
    for _, elemen := range arr {
        if elemen == bilangan {
            count++
        }
    }
    return count
}

func main() {
    var N, x, indeksHapus, bilanganCari int

    fmt.Print("Masukkan jumlah elemen array (N): ")
    fmt.Scan(&N)

    arr := make([]int, N)

    fmt.Println("Masukkan nilai untuk setiap elemen array:")
    for i := 0; i < N; i++ {
        fmt.Printf("Elemen ke-%d: ", i+1)
        fmt.Scan(&arr[i])
    }

    tampilkanArray(arr)

    tampilkanIndeksGanjil(arr)

    tampilkanIndeksGenap(arr)

    fmt.Print("Masukkan nilai x untuk indeks kelipatan x: ")
    fmt.Scan(&x)
    tampilkanIndeksKelipatanX(arr, x)

    fmt.Print("Masukkan indeks elemen yang akan dihapus: ")
    fmt.Scan(&indeksHapus)
    arr = hapusElemen(arr, indeksHapus)
    fmt.Println("Array setelah elemen dihapus:")
    tampilkanArray(arr)

    rataRata := hitungRataRata(arr)
    fmt.Printf("Rata-rata array: %.2f\n", rataRata)
}

```



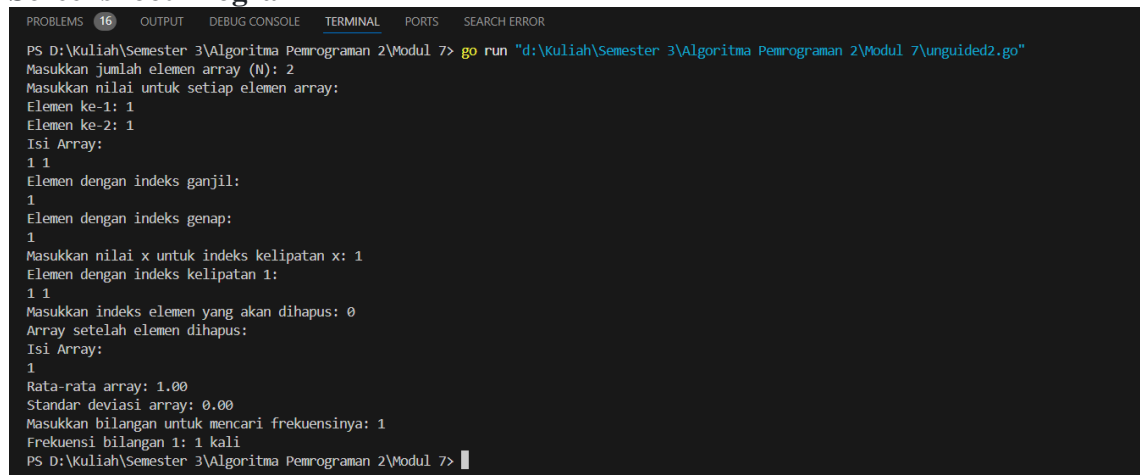
```

    standarDeviasi := hitungStandarDeviasi(arr)
    fmt.Printf("Standar deviasi array: %.2f\n", standarDeviasi)

    fmt.Print("Masukkan bilangan untuk mencari frekuensinya: ")
    fmt.Scan(&bilanganCari)
    frekuensi := hitungFrekuensi(arr, bilanganCari)
    fmt.Printf("Frekuensi bilangan %d: %d kali\n", bilanganCari,
frekuensi)
}

```

Screenshoot Program



```

PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\unguided2.go"
Masukkan jumlah elemen array (N): 2
Masukkan nilai untuk setiap elemen array:
Elemen ke-1: 1
Elemen ke-2: 1
Isi Array:
1 1
Elemen dengan indeks ganjil:
1
Elemen dengan indeks genap:
1
Masukkan nilai x untuk indeks kelipatan x: 1
Elemen dengan indeks kelipatan 1:
1 1
Masukkan indeks elemen yang akan dihapus: 0
Array setelah elemen dihapus:
1
Rata-rata array: 1.00
Standar deviasi array: 0.00
Masukkan bilangan untuk mencari frekuensinya: 1
Frekuensi bilangan 1: 1 kali
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7>

```

Deskripsi dan Cara Kerja Program

Program di atas adalah aplikasi berbasis Go yang memanipulasi dan menganalisis elemen-elemen dari sebuah array. Pengguna diminta untuk memasukkan jumlah elemen array (N) dan nilai setiap elemennya. Program menyediakan beberapa fungsi untuk berbagai operasi: `tampilkanArray` untuk menampilkan semua elemen, `tampilkanIndeksGanjil` dan `tampilkanIndeksGenap` untuk menampilkan elemen pada indeks ganjil dan genap, serta `tampilkanIndeksKelipatanX` untuk menampilkan elemen pada indeks kelipatan x. Fungsi `hapusElemen` menghapus elemen pada indeks tertentu, `hitungRataRata` menghitung rata-rata elemen, `hitungStandarDeviasi` menghitung standar deviasi, dan `hitungFrekuensi` menghitung frekuensi kemunculan bilangan tertentu dalam array. Setelah memasukkan data, program akan menampilkan hasil dari operasi-operasi ini berdasarkan input pengguna.

3. 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
```

```
package main

import (
    "fmt"
)

func main() {
    var klubA, klubB string
    var skorA, skorB int
    var hasil []string

    fmt.Print("Klub A : ")
```

```

    fmt.Scanln(&klubA)
    fmt.Print("Klub B : ")
    fmt.Scanln(&klubB)

    for i := 1; i <= 9; i++ {
        fmt.Printf("Pertandingan %d : ", i)
        fmt.Scan(&skorA, &skorB)

        if skorA < 0 || skorB < 0 {
            break
        }
        var result string
        if skorA > skorB {
            result = klubA
        } else if skorB > skorA {
            result = klubB
        } else {
            result = "Draw"
        }

        hasil = append(hasil, fmt.Sprintf("Pertandingan %d: %s", i,
result))
    }

    fmt.Println("\nHasil Pertandingan:")
    for _, result := range hasil {
        fmt.Println(result)
    }
    fmt.Println("Pertandingan selesai")
}

```

Screenshoot Program

```

PROBLEMS 18 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\unguided3.go"
Klub A : Barcelona
Klub B : Real Madrid
Pertandingan 1 : Pertandingan 2 : Pertandingan 3 : Pertandingan 4 : Pertandingan 5 : Pertandingan 6 : 3
1
Pertandingan 7 : 2
3
Pertandingan 8 : 3
4
Pertandingan 9 : 4
4

Hasil Pertandingan:
Pertandingan 1: Draw
Pertandingan 2: Draw
Pertandingan 3: Draw
Pertandingan 4: Draw
Pertandingan 5: Draw
Pertandingan 6: Barcelona
Pertandingan 7: Real
Pertandingan 8: Real
Pertandingan 9: Draw
Pertandingan selesai
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7>

```

Deskripsi dan Cara Kerja Program

Program di atas adalah aplikasi berbasis Go yang mencatat hasil pertandingan antara dua klub sepak bola, yaitu klub A dan klub B. Pengguna diminta memasukkan nama kedua klub. Selanjutnya, program akan meminta input skor untuk setiap pertandingan dari 1 hingga 9, di mana skor negatif akan menghentikan input lebih awal. Untuk setiap pertandingan, program menentukan pemenang berdasarkan skor yang lebih tinggi atau menandai pertandingan sebagai "Draw" jika skornya sama. Hasil dari setiap pertandingan disimpan dalam slice `hasil` dan dicetak setelah semua pertandingan selesai atau input dihentikan. Program memastikan bahwa semua hasil pertandingan ditampilkan sebelum mencetak pesan "Pertandingan selesai".

4. Sebuah array digunakan untuk menampung sekumpulan karakter, Anda diminta untuk membuat sebuah subprogram untuk melakukan membalikkan urutan isi array dan memeriksa apakah membentuk palindrom.

Lengkapl 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	: <u>S E N A N G</u> .
Reverse teks	: G N A N E S
Teks	: <u>K A T A K</u> .
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 ekom.universitu *Palindrom adalah teks yang dlbaca dar awal atau akhIr adalah sama, contoh: KATAK, APA, KASUR_RUSAK.

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

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

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

```
package main

import (
    "fmt"
    "strings"
)

// Fungsi untuk membalikkan urutan elemen dalam array karakter
func reverseCharacters(input []rune) []rune {
    length := len(input)
    reversedArray := make([]rune, length)
    for i := 0; i < length; i++ {
        reversedArray[i] = input[length-i-1]
    }
    return reversedArray
}

// Fungsi untuk memeriksa apakah array karakter membentuk palindrom
func isPalindrome(input []rune) bool {
    reversedArray := reverseCharacters(input)
    for i := 0; i < len(input); i++ {
        if input[i] != reversedArray[i] {
            return false
        }
    }
    return true
}

func main() {
```

```

var userInput string
fmt.Print("Masukkan sekumpulan karakter: ")
fmt.Scanln(&userInput)

// Mengonversi input ke dalam array karakter (rune) dan mengabaikan
perbedaan huruf kapital
convertedArray := []rune(strings.ToLower(userInput))

// Membalikkan array karakter
reversedConvertedArray := reverseCharacters(convertedArray)

// Menampilkan array asli dan array yang dibalik
fmt.Printf("Array asli: %s\n", string(convertedArray))
fmt.Printf("Array yang dibalik: %s\n",
string(reversedConvertedArray))

// Memeriksa apakah array tersebut membentuk palindrom
if isPalindrome(convertedArray) {
    fmt.Println("Array membentuk palindrom.")
} else {
    fmt.Println("Array tidak membentuk palindrom.")
}
}

```

Screenshoot Program

```

PROBLEMS 18 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\unguided4.go"
Masukkan sekumpulan karakter: Gajah
Array asli: gajah
Array yang dibalik: hajag
Array tidak membentuk palindrom.
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7\unguided4.go"
Masukkan sekumpulan karakter: rusak
Array asli: rusak
Array yang dibalik: kasur
Array tidak membentuk palindrom.
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 7>

```

Deskripsi dan Cara Kerja Program

Program di atas adalah aplikasi berbasis Go yang memeriksa apakah sekumpulan karakter yang dimasukkan oleh pengguna membentuk palindrom. Pengguna diminta memasukkan sekumpulan karakter, yang kemudian dikonversi menjadi array karakter (rune) dan diubah menjadi huruf kecil untuk mengabaikan perbedaan huruf kapital. Program menggunakan fungsi `reverseCharacters` untuk membalikkan urutan elemen dalam array karakter dan fungsi `isPalindrome` untuk memeriksa apakah array asli sama dengan array yang dibalik. Jika kedua array sama, maka karakter membentuk palindrom; jika tidak, maka karakter tidak membentuk palindrom. Hasilnya ditampilkan kepada pengguna, bersama dengan array asli dan array yang dibalik.

KESIMPULAN

Berdasarkan praktikum yang telah dilakukan pada Modul VII - Struct & Array, dapat disimpulkan bahwa penggunaan struktur data dalam pemrograman Go sangat penting untuk mengorganisir dan mengelola data secara efisien. Struct memungkinkan pengelompokan beberapa data yang memiliki keterkaitan menjadi satu kesatuan, seperti yang ditunjukkan dalam program penghitungan durasi parkir dan pengecekan posisi titik terhadap lingkaran. Sementara itu, Array memberikan kemampuan untuk menyimpan kumpulan data dengan tipe yang sama, baik dalam bentuk statis maupun dinamis (slice), yang sangat berguna untuk manipulasi data seperti yang ditunjukkan dalam program pengolahan array bilangan dan pengecekan palindrom.

Praktikum ini juga mendemonstrasikan bagaimana kombinasi struct dan array dapat digunakan untuk menyelesaikan berbagai permasalahan pemrograman yang kompleks. Program-program yang dibuat menunjukkan implementasi konsep-konsep penting seperti pengolahan data array, manipulasi string, perhitungan matematis, dan logika pemrograman. Penggunaan fungsi-fungsi bantuan dan subprogram membuat kode lebih terstruktur dan mudah dipahami, sementara pemahaman tentang tipe data dan struktur data yang tepat membantu dalam merancang solusi yang efisien untuk berbagai kasus penggunaan, dari sistem pencatatan skor pertandingan hingga analisis statistik data array.

DAFTAR PUSTAKA

Modul 7 Praktikum Algoritma dan Pemrograman 2 Modul Struck dan Array