LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

MODUL VII

STRUCK & ARRAY

Oleh:

Fadhel Yussie Ramadhan

2311102322

S1 TEKNIK INRMATIKA

INSTITUT TEKNOLOGI TLKOM PURWOKERTO

2024

I. DASAR TEORI

- **Tipe Bentukan**.

Tipe bentukan memungkinkan pemrograman untuk mendefinisikan suatu tipe data baru pada suatu bahasa pemrograman. Tipe bentukan 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. Sebagal contoh "Integer" dapat dirubah dengan nama allas "bilangan ". Caranya dengan menggunakan kata kunci "type".

2. Struct atau Record

   Stucture memungkinkan pemrograman untuk mengelompokkan beberapa data atau nilai yang memiliki relasi atau keterkaitan tertentu menjadi suatu kesatuan. Masing-masing nilai tersimpan dalam field dari stucture tersebut. Berbeda dengan bahasa pemrograman lain

- **Array**

  Array mempunyal ukuran (jumlah elemen) yang tetap (statis) selama eksekusi program, sehingga Jumlah elemen array menjadi bagian dari deklarasi variabel dengan tipe array.Jumlah elemen array dapat diminta dengan fungsi len yang tersedia.

  Slice (Array dinamik)

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

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

  • 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.

II. GUIDED

1. Source Code

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

type Mahasiswa struct {
    Nama       string
    Matematika int
    Fisika     int
    Kimia      int
    RataRata   float64
}


 (m *Mahasiswa) {     total :=
m.Matematika + m.Fisika +
m.Kimia
    m.RataRata = float64(total / 3)
}

func main() {
    // Array untuk menampung data
mahasiswa
    mahasiswa := []Mahasiswa{
{"Aldi", 85, 90, 80, 0},
        {"Budi", 70, 75, 80, 0},
        {"Cici", 90, 85, 95, 0},
        {"Doni", 60, 65, 70, 0},
        {"Eka", 100, 95, 90, 0},
    }

    for i := range mahasiswa {
hitungRataRata(&mahasiswa[i])
    }
```

```
(descending)
    sort.Slice(mahasiswa, func(i, j int)
bool {
        return mahasiswa[i].RataRata >
mahasiswa[j].RataRata
    })

fmt.Println("Peringkat mahasiswa
berdasarkan rata - rata nilai: ")
for i, m := range mahasiswa {
fmt.Printf("%d. %s - Ratarata: %.2f
(Matematika: %d, Fisika:
%d, Kimia: %d)\n", i+1, m.Nama,
m.RataRata, m.Matematika, m.Fisika,
m.Kimia)
    }
}
```

Output

Keterangan

menghitung rata-rata nilai matematika, fisika, dan kimia untuk setiap siswa, kemudian mengurutkan siswa berdasarkan nilai rata-rata tersebut secara menurun. Hasil akhirnya adalah peringkat siswa dari nilai rata-rata tertinggi hingga terendah.

2. Source Code

```
Package main
Import "fmt"

func main() {

    mahasiswa := map[string]string{
        "20231001": "Andi",
        "20231002": "Budi",
        "20231003": "Cici",
    }


fmt.Println("Daftar Mahasiswa:")
fmt.Println("NIM\t\tNama")
fmt.Println("-----------------------
--")     for nim, nama := range
mahasiswa {
fmt.Printf("%s\t%s\n", nim, nama)
    }

nim := "20231002"
    fmt.Println("\nNama Mahasiswa
dengan NIM", nim,
"adalah", mahasiswa[nim])

    delete(mahasiswa, "20231003")

fmt.Println("\nDaftar Mahasiswa
setelah dihapus:")
fmt.Println("NIM\t\tNama")
fmt.Println("----------------------
---")     for nim, nama := range
mahasiswa {
fmt.Printf("%s\t%s\n", nim, nama)
    }
}
```

Output

Keterangan
Program ini menggunakan folder untuk menyimpan data siswa dengan NIM sebagai kunci dan nama sebagai nilainya. User dapat menambah, mengambil, dan menghapus data dalam folder. Ini juga menampilkan data sebelum dan sesudah penghapusan dalam format tabel .

III. UNGUIDED

1. Source Code

```
package main

import (
"fmt"
    "math"
)

type Titik struct {
    x, y int
}
type Lingkaran struct {
    pusat
Titik
radius int
}

func jarak(p, q Titik) float64 {
    return math.Sqrt(float64((p.x-
q.x)*(p.x-q.x) + (p.yq.y)*(p.y-
q.y))) }
func didalamLingkaran(l Lingkaran,
t Titik) bool {     return
jarak(l.pusat, t) <=
float64(l.radius)
}
func posisiTitik(l1, l2 Lingkaran,
t Titik) string {
    dalamL1 :=
didalamLingkaran(l1, t)
dalamL2 :=
didalamLingkaran(l2, t)
if dalamL1 && dalamL2 {
        return "Titik di dalam
lingkaran 1 dan 2"
    } else if dalamL1 {
        return "Titik di dalam
lingkaran 1"
    } else if dalamL2 {
        return "Titik di dalam
lingkaran 2"
    } else {
        return "Titik di luar
lingkaran 1 dan 2"
```

```
} }
func main() {
var l1, l2
Lingkaran
    var t Titik
    fmt.Println("Masukkan koordinat
pusat dan radius lingkaran 1:")
    fmt.Scan(&l1.pusat.x,
&l1.pusat.y, &l1.radius)
```

```
    fmt.Println("Masukkan koordinat
pusat dan radius lingkaran 2:")
fmt.Scan(&l2.pusat.x, &l2.pusat.y,
&l2.radius)     fmt.Println("Masukkan
```

```
koordinat titik sembarang:")
fmt.Scan(&t.x, &t.y)      hasil :=
posisiTitik(l1, l2, t)
fmt.Println(hasil)
}
```

Output

Keterangan

Program ini menentukan posisi suatu titik relatif terhadap dua lingkaran. Kode ini menghitung jarak suatu titik ke pusat lingkaran dan memeriksa apakah titik berada dalam radius lingkaran.Berdasarkan hasil pengujian, kode menghasilkan salah satu dari empat hasil: "Titik di dalam lingkaran 1 dan 2", "Titik di dalam lingkaran 1", "Titik di dalam lingkaran 2" ” dan “Titik di luar lingkaran 1 dan 2.” . Hal ini memudahkan untuk melihat letak titik-titik tersebut relatif terhadap kedua lingkaran .

2. Source Code

```
package main

import (
"fmt"
   "math"
)

func displayArray(arr []int) {
fmt.Println("Isi array:", arr)
}
func displayElementsByIndex(arr []int, start int, step int,
desc string) {
   fmt.Printf("Elemen dengan indeks %s: ", desc)
```

```
    for i := start; i < len(arr); i += step {         fmt.Print(arr[i], "
")
    }
    fmt.Println()
}
func deleteElementAtIndex(arr *[]int, index int) {     if
index >= 0 && index < len(*arr) {
        *arr = append((*arr)[:index], (*arr)[index+1:]...)
displayArray(*arr)
    } else {
        fmt.Println("Indeks tidak valid.")
    } }
func calculateAverage(arr []int) float64 {     if
len(arr) == 0 {
        return 0
    }
    sum := 0
    for _, v := range arr {         sum
+= v
    }
    return float64(sum) / float64(len(arr))
}
func calculateStandardDeviation(arr []int, avg float64) float64 {
    if len(arr) == 0 {
        return 0
    }
    var varianceSum float64     for _, v := range arr {
varianceSum += math.Pow(float64(v)-avg, 2)
    }
    return math.Sqrt(varianceSum / float64(len(arr)))
}
func countFrequency(arr []int, num int) int {
count := 0     for _, v := range arr {         if v == num
{
            count++
        }
    }
    return count
}
func main() {     var N
int
    fmt.Print("Masukkan jumlah elemen array: ")
```

```
    fmt.Scan(&N)     arr
:= make([]int, N)     for
i := 0; i < N; i++ {
        fmt.Printf("Masukkan elemen ke-%d: ", i)
fmt.Scan(&arr[i])
    }
    displayArray(arr)
displayElementsByIndex(arr, 1, 2, "ganjil")
displayElementsByIndex(arr, 0, 2, "genap")
var x int
    fmt.Print("Masukkan bilangan x untuk kelipatan
indeks: ")     fmt.Scan(&x)
displayElementsByIndex(arr, x, x,
fmt.Sprintf("kelipatan %d", x))
    var index int     fmt.Print("Masukkan indeks untuk
menghapus elemen:
")
    fmt.Scan(&index)
    deleteElementAtIndex(&arr, index)     avg :=
calculateAverage(arr)     fmt.Println("Rata-rata
array:", avg)     stdDev :=
calculateStandardDeviation(arr, avg)
fmt.Println("Simpangan baku (standar deviasi):",
stdDev)     var num int
    fmt.Print("Masukkan bilangan untuk
menghitung frekuensi: ")     fmt.Scan(&num)
    fmt.Printf("Frekuensi bilangan %d: %d\n", num,
countFrequency(arr, num))
}
```

Output

Keterangan

Program ini menggunakan fungsi berikut untuk mengelola array bilangan bulat. Menampilkan semua elemen array, menampilkan elemen pada indeks yang ganjil, genap, atau kelipatan x, menghilangkan elemen pada indeks tertentu, dan menghitung mean dan deviasi standar array. Kemudian hitung berapa kali angka tersebut muncul di array .

3. Source Code

```
package main

import (
   "fmt"
)

func main() {
   var klubA, klubB string
fmt.Print("Masukkan nama Klub A: ")
fmt.Scanln(&klubA)
   fmt.Print("Masukkan nama Klub B:
")    fmt.Scanln(&klubB)    var
pemenang []string    var pertandingan
int    for i := 1; ; i++ {        var skorA,
skorB int
      fmt.Printf("Masukkan skor pertandingan %d (%s vs
%s): ", i, klubA,
         klubB)
      fmt.Scan(&skorA, &skorB)
```

```
        if skorA < 0 || skorB < 0 {
            fmt.Println("Skor tidak valid, menghentikan
input...")
            pertandingan = i - 1
            break
        }
        if skorA > skorB {
pemenang = append(pemenang,
fmt.Sprintf("Hasil %d: %s menang", i,
klubA))
        } else if skorB > skorA {
pemenang = append(pemenang,
fmt.Sprintf("Hasil %d: %s menang", i,
klubB))
        } else {
            pemenang = append(pemenang,
fmt.Sprintf("Hasil %d: Draw", i))
        }
    }
    fmt.Println("Hasil pertandingan:")
for i := 0; i < pertandingan; i++ {
        fmt.Println(pemenang[i])
    }
    fmt.Println("Pertandingan selesai")
}
```

Output

Keterangan

Memungkinkan pengguna memasukkan skor dan menampilkan hasil pertandingan antara dua klub sepak bola Nilai negatif akan menghentikan input. Program akan menentukan pemenang berdasarkan skor. Jika skornya lebih tinggi, Klub A menang; jika skornya lebih tinggi, Klub B menang; jika skornya sama, hasilnya adalah "seri". Hasil setiap permainan disimpan dalam array, dan setelah input selesai, program menampilkan ringkasan hasil permainan dan pesan "permainan selesai". Program ini efektif dalam merekam dan menampilkan hasil permainan karena kontrol inputnya yang sangat baik.

4. Source Code

```
package main

import "fmt"

const NMAX int = 127

type tabel [NMAX]rune

func isiArray(t *tabel, n *int) {
   var input string
fmt.Scanln(&input)
   *n = 0
   for _, ch := range input {
if ch == '.' || *n >= NMAX {
break
      }
      t[*n] = ch
      *n++
   } }
func cetakArray(t tabel, n int) {
   for i := 0; i < n; i++ {
fmt.Printf("%c", t[i])
   }
   fmt.Println()
}
func balikkanArray(t *tabel, n int) {
   for i := 0; i < n/2; i++ {
t[i], t[n-i-1] = t[n-i-1], t[i]
   } } func main() {     var tab
tabel     var n int
fmt.Print("Teks: ")
isiArray(&tab, &n)
balikkanArray(&tab, n)
fmt.Print("Reverse Teks: ")
cetakArray(tab, n)
}
```

Output

Keterangan

Program ini memproses string dan menggunakan array. Meskipun mempunyai keterbatasan dalam fleksibilitas input, program ini berhasil mencapai tujuan yaitu membalikkan urutan karakter dan menampilkan hasilnya. Dengan beberapa perbaikan, seperti opsi masukan yang diperluas dan penanganan tipe karakter yang berbeda, program ini dapat menjadi alat yang lebih berguna dan mumpuni.