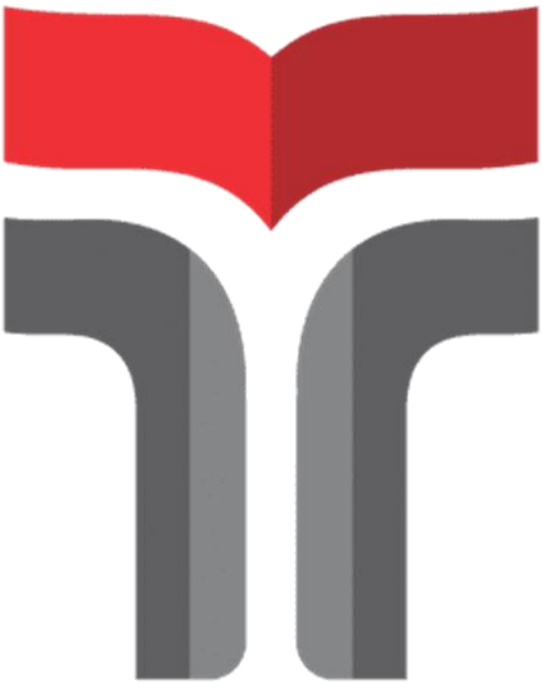
LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2

MODUL 7

STRUCK & ARRAY



Oleh:

TSAQIF KANZ AHMAD
2311102075
IF-11-02

S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM
PURWOKERTO
2024

I. DASAR TEORI

A. Tipe Bentukan

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

A.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		}

Fakultas Informatika

Sebagai contoh perhatikan program Go berikut beserta hasil eksekusinya!

```
package main
import "fmt"
type bilangan int
type pecahan float64
func main() {
       var a, b bilangan
        var hasil pecahan
        a = 9
        a = 5
        hasil - pecahan(a) / pecahan(b)
         fmt.Println(hasil)
}
```

A.2 Struct

A.2.1 Pengertian

Struct adalah kumpulan definisi variabel (atau property) dan atau fungsi (atau method), yang dibungkus sebagai tipe data baru dengan nama tertentu. Property dalam struct, tipe datanya bisa bervariasi. Mirip seperti map, hanya saja key-nya sudah didefinisikan di awal, dan tipe data tiap itemnya bisa berbeda. Dari sebuah struct, kita bisa buat variabel baru, yang memiliki atribut sesuai skema struct tersebut. Variabel tersebut dipanggil dengan istilah object atau variabel object. Dengan memanfaatkan struct, penyimpanan data yang sifatnya kolektif menjadi lebih mudah, lebih rapi, dan mudah untuk dikelola.

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

A.2.2 Deklarasi Struct

Kombinasi keyword **type** dan **struct** digunakan untuk deklarasi struct. Di bawah ini merupakan contoh cara penerapannya.

```
type student struct {
    name string
    grade int
}
```

Struct **student** dideklarasikan memiliki 2 property, yaitu **name** dan **grade**. Property adalah istilah untuk variabel yang menempel ke struct.

A.2.3 Penerapan Struct Untuk Membuat Object

Struct student yang sudah disiapkan di atas kita gunakan untuk membuat variabel objek. Variabel tersebut tipe datanya adalah **student**. Kemudian dari variabel object, kita bisa mengakses isi property variabel. Contoh:

```
func main() {
    var s1 student
    s1.name = "john wick"
    s1.grade = 2

    fmt.Println("name  :", s1.name)
    fmt.Println("grade :", s1.grade)
}
```

Cara membuat variabel objek sama seperti pembuatan variabel biasa. Tinggal tulis saja nama variabel diikuti nama struct, contoh: **var s1 student**. Semua property variabel objek pada awalnya memiliki zero value sesuai tipe datanya. Misalnya, 0 untuk tipe **int**, dan string kosong "" untuk string. Property variabel objek bisa diakses nilainya menggunakan notasi titik, contohnya **s1.name**. Nilai property-nya juga bisa diubah, contohnya **s1.grade = 2**.

```
novalagung:belajar-golang $ go run bab23.go
name  : john wick
grade : 2
novalagung:belajar-golang $
```

A.2.4 Inisialisasi Object Struct

Cara inisialisasi variabel objek adalah dengan menuliskan nama struct yang telah dibuat diikuti dengan kurung kurawal. Nilai

masing-masing property bisa diisi pada saat inisialisasi. Pada contoh berikut, terdapat 3 buah variabel objek yang dideklarasikan dengan cara berbeda.

```
var s1 = student{}
s1.name = "wick"
s1.grade = 2

var s2 = student{"ethan", 2}

var s3 = student{name: "jason"}

fmt.Println("student 1 :", s1.name)
fmt.Println("student 2 :", s2.name)
fmt.Println("student 3 :", s3.name)
```

Pada kode di atas, variabel **s1** menampung objek cetakan **student**. Variabel tersebut kemudian di-set nilai property-nya. Variabel objek **s2** dideklarasikan dengan metode yang sama dengan **s1**, pembedanya di **s2** nilai propertinya di isi langsung ketika deklarasi. Nilai pertama akan menjadi nilai property pertama (yaitu **name**), dan selanjutnya berurutan.

Pada deklarasi s3, dilakukan juga pengisian property ketika pencetakan objek. Hanya saja, yang diisi hanya name saja. Cara ini cukup efektif jika digunakan untuk membuat objek baru yang nilai property-nya tidak semua harus disiapkan di awal. Keistimewaan lain menggunakan cara ini adalah penentuan nilai property bisa dilakukan dengan tidak berurutan. Contohnya:

```
var s4 = student{name: "wayne", grade: 2}
var s5 = student{grade: 2, name: "bruce"}
```

A.2.5 Variabel Objek Pointer

Objek yang dibuat dari tipe struct bisa diambil nilai pointer-nya, dan bisa disimpan pada variabel objek yang bertipe struct pointer. Contoh penerapannya:

```
var s1 = student{name: "wick", grade: 2}

var s2 *student = &s1
fmt.Println("student 1, name :", s1.name)
fmt.Println("student 4, name :", s2.name)

s2.name = "ethan"
fmt.Println("student 1, name :", s1.name)
fmt.Println("student 4, name :", s2.name)
```

s2 adalah variabel pointer hasil cetakan struct **student. s2** menampung nilai referensi **s1**, menjadikan setiap perubahan pada property variabel tersebut, akan juga berpengaruh pada variabel objek **s1**.Meskipun **s2** bukan variabel asli, property nya tetap bisa diakses seperti biasa. Inilah keistimewaan property dalam objek

pointer, tanpa perlu di-dereferensi nilai asli property tetap bisa diakses. Pengisian nilai pada property tersebut juga bisa langsung menggunakan nilai asli, contohnya seperti **s2.name = "ethan"**.

```
novalagung:belajar-golang $ go run bab23.go
student 1, name : wick
student 4, name : wick
student 1, name : ethan
student 4, name : ethan
novalagung:belajar-golang $
```

A.2.6 Embedded Struct

Embedded struct adalah mekanisme untuk menempelkan sebuah struct sebagai properti struct lain. Agar lebih mudah dipahami, mari kita bahas kode berikut.

```
package main

import "fmt"

type person struct {
    name string
    age  int
}

type student struct {
    grade int
    person
}

func main() {
    var s1 = student{}
    s1.name = "wick"
    s1.age = 21
    s1.grade = 2

    fmt.Println("name  :", s1.name)
    fmt.Println("age   :", s1.age)
    fmt.Println("age   :", s1.person.age)
    fmt.Println("grade :", s1.grade)
}
```

Pada kode di atas, disiapkan struct **person** dengan properti yang tersedia adalah **name** dan **age**. Disiapkan juga struct **student** dengan property **grade**. Struct **person** di-embed ke dalam struct **student**. Caranya cukup mudah, yaitu dengan menuliskan nama struct yang ingin di-embed ke dalam body **struct** target. Embedded struct adalah **mutable**, nilai property-nya nya bisa diubah. Khusus untuk properti yang bukan merupakan properti asli (melainkan properti turunan dari struct lain), pengaksesannya dilakukan dengan

cara mengakses struct *parent-nya* terlebih dahulu, contohnya **s1.person.age**. Nilai yang dikembalikan memiliki referensi yang sama dengan **s1.age**.

A.2.7 Embedded Struct Dengan Nama Properti Yang Sama

Jika salah satu nama properti sebuah struct memiliki kesamaan dengan properti milik struct lain yang di-embed, maka pengaksesan property-nya harus dilakukan secara eksplisit atau jelas. Silakan lihat kode berikut agar lebih jelas.

```
package main

import "fmt"

type person struct {
    name string
    age  int
}

type student struct {
    person
    age   int
    grade int
}

func main() {
    var s1 = student{}
    s1.name = "wick"
    s1.age = 21        // age of student
    s1.person.age = 22 // age of person

    fmt.Println(s1.name)
    fmt.Println(s1.age)
    fmt.Println(s1.person.age)
}
```

Struct **person** di-embed ke dalam struct **student**, dan kedua struct tersebut kebetulan salah satu nama property-nya ada yang sama, yaitu **age**. Cara mengakses property **age** milik struct **person** lewat objek struct **student**, adalah dengan menuliskan nama struct yang di-embed kemudian nama property-nya, contohnya: **s1.person.age = 22**.

A.2.8 Pengisian Nilai Sub-Struct

Pengisian nilai property sub-struct bisa dilakukan dengan langsung memasukkan variabel objek yang tercetak dari struct yang sama.

```
var p1 = person{name: "wick", age: 21}
var s1 = student{person: p1, grade: 2}
```

```
fmt.Println("name  :", s1.name)
fmt.Println("age   :", s1.age)
fmt.Println("grade :", s1.grade)
```

Pada deklarasi **s1**, property **person** diisi variabel objek **p1**.

A.2.9 Anymous Struct

Anonymous struct adalah struct yang tidak dideklarasikan di awal sebagai tipe data baru, melainkan langsung ketika pembuatan objek. Teknik ini cukup efisien digunakan pada use case pembuatan variabel objek yang struct-nya hanya dipakai sekali.

```
package main

import "fmt"

type person struct {
    name string
    age  int
}

func main() {
    var s1 = struct {
        person
        grade int
    }{}
    s1.person = person{"wick", 21}
    s1.grade = 2

    fmt.Println("name  :", s1.person.name)
    fmt.Println("age   :", s1.person.age)
    fmt.Println("grade :", s1.grade)
}
```

Pada kode di atas, variabel **s1** langsung diisi objek anonymous struct yang memiliki property **grade**, dan property **person** yang merupakan embedded struct. Salah satu aturan yang perlu diingat dalam pembuatan anonymous struct adalah, deklarasi harus diikuti dengan inisialisasi. Bisa dilihat pada **s1** setelah deklarasi struktur struct, terdapat kurung kurawal untuk inisialisasi objek. Meskipun nilai tidak diisikan di awal, kurung kurawal tetap harus ditulis.

```
// anonymous struct tanpa pengisian property
var s1 = struct {
    person
    grade int
}{}

// anonymous struct dengan pengisian property
var s2 = struct {
```

```
    person
    grade int
}{
    person: person{"wick", 21},
    grade:  2,
}
```

A.2.10 Kombinasi Slice & Struct

Slice dan **struct** bisa dikombinasikan seperti pada slice dan **map**, caranya penggunaannya-pun mirip, cukup tambahkan tanda **[ ]** sebelum tipe data pada saat deklarasi.

```
type person struct {
    name string
    age  int
}

var allStudents = []person{
    {name: "Wick", age: 23},
    {name: "Ethan", age: 23},
    {name: "Bourne", age: 22},
}

for _, student := range allStudents {
    fmt.Println(student.name, "age is",
student.age)
}
```

A.2.11 Inisialisasi Slice Anonymous Struct

Anonymous struct bisa dijadikan sebagai tipe sebuah slice. Dan nilai awalnya juga bisa diinisialisasi langsung pada saat deklarasi. Berikut adalah contohnya:

```
var allStudents = []struct {
    person
    grade int
}{
    {person: person{"wick", 21}, grade: 2},
    {person: person{"ethan", 22}, grade: 3},
    {person: person{"bond", 21}, grade: 3},
}

for _, student := range allStudents {
    fmt.Println(student)
}
```

A.2.12 Deklarasi Anonymous Struct Menggunakan Keyword var

Cara lain untuk deklarasi anonymous struct adalah dengan menggunakan keyword **var**.

```
var student struct {
    person
    grade int
}

student.person = person{"wick", 21}
student.grade = 2
```

Statement **type student struct** adalah contoh cara deklarasi struct. Maknanya akan berbeda ketika keyword **type** diganti **var**, seperti pada contoh di atas **var student struct**, yang artinya dicetak sebuah objek dari anonymous struct kemudian disimpan pada variabel bernama **student**. Deklarasi anonymous struct menggunakan metode ini juga bisa dilakukan dengan disertai inisialisasi data.

```
// hanya deklarasi
var student struct {
    grade int
}

// deklarasi sekaligus inisialisasi
var student = struct {
    grade int
} {
    12,
}
```

A.2.13 Nested Struct

Nested struct adalah anonymous struct yang di-embed ke sebuah struct. Deklarasinya langsung di dalam struct peng-embed. Contoh:

```
type student struct {
    person struct {
        name string
        age  int
    }
    grade   int
    hobbies []string
}
```

Teknik ini biasa digunakan ketika decoding data JSON yang struktur datanya cukup kompleks dengan proses decode hanya sekali.

A.2.14 Deklarasi Dan Inisialisasi Struct Secara Horizontal

Deklarasi struct bisa dituliskan secara horizontal, caranya bisa dilihat pada kode berikut:

```
type person struct { name string; age int;
hobbies []string }
```

Tanda semi-colon (;) digunakan sebagai pembatas deklarasi poperty yang dituliskan secara horizontal. Inisialisasi nilai juga bisa dituliskan dengan metode ini. Contohnya:

```
var p1 = struct { name string; age int } { age:
22, name: "wick" }
var p2 = struct { name string; age int } {
"ethan", 23 }
```

A.2.15 Tag property dalam struct

Tag merupakan informasi opsional yang bisa ditambahkan pada property struct.

```
type person struct {
    name string `tag1`
    age  int    `tag2`
}
```

Tag biasa dimanfaatkan untuk keperluan encode/decode data. Informasi tag juga bisa diakses lewat reflect. Nantinya akan ada pembahasan yang lebih detail mengenai pemanfaatan tag dalam struct, terutama ketika sudah masuk chapter JSON.

B. Array

Array adalah kumpulan data bertipe sama, yang disimpan dalam sebuah variabel. Array memiliki kapasitas yang nilainya ditentukan pada saat pembuatan, menjadikan elemen/data yang disimpan di array tersebut jumlahnya tidak boleh melebihi yang sudah dialokasikan. Default nilai tiap elemen array pada awalnya tergantung dari tipe datanya. Jika int maka tiap element zero value-nya adalah 0, jika bool maka false, dan seterusnya. Setiap elemen array memiliki indeks berupa angka yang merepresentasikan posisi urutan elemen tersebut. Indeks array dimulai dari 0.

Contoh penerapan array:

```
var names [4]string
names[0] = "trafalgar"
names[1] = "d"
names[2] = "water"
names[3] = "law"

fmt.Println(names[0], names[1], names[2], names[3])
```

Variabel names dideklarasikan sebagai array string dengan alokasi kapasitas elemen adalah 4 slot. Cara mengisi slot elemen array bisa dilihat di kode di atas, yaitu dengan langsung mengakses elemen menggunakan indeks, lalu mengisinya.

```
novalagung:belajar-golang $ go run bab14.go
trafalgar d water law
novalagung:belajar-golang $
```

B.1 Pengisian Elemen Array yang Melebihi Alokasi Awal

Pengisian elemen array pada indeks yang tidak sesuai dengan jumlah alokasi menghasilkan error. Contoh: jika array memiliki 4 slot, maka pengisian nilai slot 5 seterusnya adalah tidak valid.

```
var names [4]string
names[0] = "trafalgar"
names[1] = "d"
names[2] = "water"
names[3] = "law"
names[4] = "ez" // baris kode ini menghasilkan error
```

Solusi dari masalah di atas adalah dengan menggunakan keyword **append**

B.2 Inisialisasi Nilai Awal Array

Pengisian elemen array bisa dilakukan pada saat deklarasi variabel. Caranya dengan menuliskan data elemen dalam kurung kurawal setelah tipe data, dengan pembatas antar elemen adalah tanda koma (,).

```
var fruits = [4]string{"apple", "grape", "banana",
"melon"}

fmt.Println("Jumlah element \t\t", len(fruits))
fmt.Println("Isi semua element \t", fruits)
```

Penggunaan fungsi **fmt.Println( )** pada data array tanpa mengakses indeks tertentu, menghasilkan output dalam bentuk string dari semua array yang ada. Teknik ini umum digunakan untuk keperluan debugging data array.

```
[novalagung:belajar-golang $ go run bab14.go
Jumlah element          4
Isi semua element       [apple grape banana melon]
novalagung:belajar-golang $
```

Fungsi **len( )** berfungsi untuk menghitung jumlah elemen sebuah array.

B.3 Inisialisasi Nilai Array Dengan Gaya Vertikal

Elemen array bisa dituliskan dalam bentuk horizontal (seperti yang sudah dicontohkan di atas) ataupun dalam bentuk vertikal.

```
var fruits [4]string

// cara horizontal
fruits  = [4]string{"apple", "grape", "banana",
"melon"}

// cara vertikal
fruits  = [4]string{
    "apple",
    "grape",
    "banana",
    "melon",
```

```
}
```

Khusus untuk deklarasi array dengan cara vertikal, tanda koma wajib dituliskan setelah setiap elemen (termasuk elemen terakhir), agar tidak memunculkan syntax error.

B.4 Inisialisasi Nilai Awal Array Tanpa Jumlah Elemen

Deklarasi array yang nilainya diset di awal, boleh tidak dituliskan jumlah lebar array-nya, cukup ganti dengan tanda 3 titik (...). Metode penulisan ini membuat kapasitas array otomatis dihitung dari jumlah elemen array yang ditulis.

```
var numbers = [...]int{2, 3, 2, 4, 3}

fmt.Println("data array \t:", numbers)
fmt.Println("jumlah elemen \t:", len(numbers))
```

Variabel **numbers** secara otomatis kapasitas elemennya adalah **5**.

```
[novalagung:belajar-golang $ go run bab14.go
data array      : [2 3 2 4 3]
jumlah elemen   : 5
novalagung:belajar-golang $
```

B.5 Array Multidimensi

Array multidimensi adalah array yang tiap elemennya juga berupa array. Cara deklarasi array multidimensi secara umum sama dengan array biasa, bedanya adalah pada array biasa, setiap elemen berisi satu nilai, sedangkan pada array multidimensi setiap elemen berisi array. Khusus penulisan array yang merupakan subdimensi/elemen, boleh tidak dituliskan jumlah datanya. Contohnya bisa dilihat pada deklarasi variabel **numbers2** di kode berikut.

```
var numbers1 = [2][3]int{[3]int{3, 2, 3}, [3]int{3,
4, 5}}
var numbers2 = [2][3]int{{3, 2, 3}, {3, 4, 5}}

fmt.Println("numbers1", numbers1)
fmt.Println("numbers2", numbers2)
```

Kedua array di atas memiliki jumlah dan isi elemen yang sama.

```
[novalagung:belajar-golang $ go run bab14.go
numbers1 [[3 2 3] [3 4 5]]
numbers2 [[3 2 3] [3 4 5]]
novalagung:belajar-golang $
```

B.6 Perulangan Elemen Array Menggunakan Keyword for

Keyword **for** dan array memiliki hubungan yang sangat erat. Dengan memanfaatkan perulangan/looping menggunakan keyword ini, elemen-elemen dalam array bisa didapat. Ada beberapa cara yang bisa digunakan untuk me-looping data array, yg pertama adalah dengan memanfaatkan variabel iterasi perulangan untuk mengakses elemen berdasarkan indeks-nya. **Contoh :**

```
var fruits = [4]string{"apple", "grape", "banana",
"melon"}

for i := 0; i < len(fruits); i++ {
    fmt.Printf("elemen %d : %s\n", i, fruits[i])
}
```

Perulangan di atas dijalankan sebanyak jumlah elemen array **fruits** (bisa diketahui dari kondisi **i < len(fruits**). Di tiap perulangan, elemen array diakses lewat variabel iterasi i.

```
[novalagung:belajar-golang $ go run bab14.go
elemen 0 : apple
elemen 1 : grape
elemen 2 : banana
elemen 3 : melon
novalagung:belajar-golang $
```

B.7 Perulangan Elemen Array Menggunakan Keyword for - range

Ada cara lain yang lebih sederhana untuk operasi perulangan array, yaitu menggunakan kombinasi keyword for - range. Contoh pengaplikasiannya bisa dilihat di kode berikut.

```
var fruits = [4]string{"apple", "grape", "banana",
"melon"}

for i, fruit := range fruits {
    fmt.Printf("elemen %d : %s\n", i, fruit)
}
```

Array **fruits** diambil elemen-nya secara berurutan. Nilai tiap elemen ditampung variabel oleh **fruit** (tanpa huruf s), sedangkan indeks nya ditampung variabel **i**. Output program di atas, sama persis dengan output program sebelumnya, hanya saja cara yang diterapkan berbeda.

B.8 Penggunaan Variabel Underscore _ Dalam for - range

Terkadang, dalam penerapan looping menggunakan for - range, ada kebutuhan di mana yang dibutuhkan dari perulangan adlah adalah elemen-nya saja, sedangkan indeks-nya tidak, **Contoh :**

```
var fruits = [4]string{"apple", "grape", "banana",
"melon"}

for i, fruit := range fruits {
    fmt.Printf("nama buah : %s\n", fruit)
}
```

Hasil dari kode program di atas adalah error, karena Go tidak memperbolehkan adanya variabel yang menganggur atau tidak dipakai.

```
[novalagung:belajar-golang $ go run bab14.go
# command-line-arguments
./bab14.go:8: i declared and not used
novalagung:belajar-golang $
```

Di sinilah salah satu kegunaan dari variabel pengangguran, atau underscore (_). Tampung saja nilai yang tidak ingin digunakan ke underscore.

```
var fruits = [4]string{"apple", "grape", "banana",
"melon"}

for _, fruit := range fruits {
    fmt.Printf("nama buah : %s\n", fruit)
}
```

Pada kode di atas, yang sebelumnya adalah variabel **i** diganti dengan **_** , karena kebetulan variabel **i** tidak digunakan.

```
novalagung:belajar-golang $ go run bab14.go
nama buah : apple
nama buah : grape
nama buah : banana
nama buah : melon
novalagung:belajar-golang $
```

Bagaiamana jika sebaliknya? Misal, yang dibutuhkan hanya indeks-nya saja, nilainya tidak penting. Maka cukup tulis satu variabel saja setelah keyword for, yaitu variabel penampung nilai indeks.

```
for i, _ := range fruits { }
// atau
for i := range fruits { }
```

B.9 Alokasi Elemen Array Menggunakan Keyword make

Deklarasi sekaligus alokasi kapasitas array juga bisa dilakukan lewat keyword make.

```
var fruits = make([]string, 2)
fruits[0] = "apple"
fruits[1] = "manggo"

fmt.Println(fruits)  // [apple manggo]
```

Parameter pertama keyword make diisi dengan tipe data elemen array yang diinginkan, parameter kedua adalah jumlah elemennya. Pada kode di atas, variabel fruits tercetak sebagai array string dengan kapasitas alokasi 2 slot.

B.10 Maps

Maps adalah struktur data bawaan yang mewakili kumpulan pasangan kunci-nilai. Peta menyediakan cara yang efisien untuk menyimpan, mengambil, dan memperbarui nilai berdasarkan kunci yang unik. Untuk mendeklarasikan maps dapat menggunakan **make** fungsi atau menginisialisasinya langsung dengan nilai:

```
// Menggunakan make
 ages := make ( map [ string ] int )   // Membuat map
kosong dengan kunci string dan nilai integer
```

```
// Menginisialisasi langsung
 scores := map [ string ] int {
    "John" : 90 ,
    "Alice" : 85 ,
    "Bob" : 92 ,
}
```

Untuk mengakses elemen dalam peta, gunakan tombol di dalam tanda kurung siku. Misalnya:

```
johnScore := scores[ "John" ]   // Mengambil nilai
yang terkait dengan kunci "John"
```

Anda juga dapat menggunakan nilai pengembalian kedua opsional saat mengakses nilai peta untuk memeriksa apakah kunci tersebut ada di peta:

```
skor, ada := skor[ "Alice" ]
jika  ada {
    // Kunci ada di peta
     fmt.Println( "Skor Alice:" , skor)
} jika tidak {
    // Kunci tidak ada di peta
    fmt.Println(" Skor Alice tidak ditemukan ")
}
```

Maps bersifat dinamis dan dapat bertambah atau berkurang seiring elemen ditambahkan atau dihapus. Untuk menambahkan atau memperbarui elemen dalam peta, Anda menetapkan nilai ke kunci tertentu:

```
scores[ "Alice" ] = 90   // Memperbarui nilai yang
terkait dengan kunci "Alice"
 scores[ "Charlie" ] = 88   // Menambahkan pasangan
kunci-nilai baru ke peta
```

Untuk menghapus elemen dari peta, menggunakan **delete** fungsi:

```
delete (skor, "Bob" )   // Menghapus pasangan kunci-
nilai dengan kunci "Bob" dari peta
```

Maps digunakan secara luas dalam Go untuk berbagai keperluan, seperti menghitung kejadian, menyimpan data dalam cache, dan membangun tabel pencarian yang efisien. Memahami cara bekerja dengan peta akan memungkinkan Anda untuk menangani data nilai-kunci secara efisien dalam program Go

II. GUIDED

1) SOURCE CODE

```go
package main

//Mengurutkan data array
import (
    "fmt"
    "sort"
)

// Struktur untuk menampung data mahasiswa
type Mahasiswa struct {
    Nama       string
    Matematika int
    Fisika     int
    Kimia      int
    RataRata   float64
}

// Fungsi untuk menghitung rata-rata nilai tiap mahasiswa
func hitungRataRata(m *Mahasiswa) {
    total := m.Matematika + m.Fisika + m.Kimia
    m.RataRata = float64(total) / 3.0
}

// Fungsi utama untuk mengelola dan mengurutkan data mahasiswa
berdasarkan nilai rata-rata
func main() {
    // Array untuk menampung data mahasiswa
    mahasiswa := []Mahasiswa{
        {"Ali", 85, 90, 80, 0},
        {"Budi", 70, 75, 80, 0},
        {"Cici", 90, 85, 95, 0},
        {"Doni", 60, 65, 70, 0},
        {"Eka", 100, 95, 90, 0},
    }

    // Menghitung rata-rata nilai tiap mahasiswa
    for i := range mahasiswa {
        hitungRataRata(&mahasiswa[i])
    }

    // Mengurutkan mahasiswa berdasarkan nilai rata-rata
(descending)
    sort.Slice(mahasiswa, func(i, j int) bool {
```

```
        return mahasiswa[i].RataRata > mahasiswa[j].RataRata
    })

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

SCREENSHOT OUTPUT :

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
Guided\Guided1.go"
Peringkat mahasiswa berdasarkan rata-rata nilai:
1. Eka - Rata-rata: 95.00 (Matematika: 100, Fisika: 95, Kimia: 90)
2. Cici - Rata-rata: 90.00 (Matematika: 90, Fisika: 85, Kimia: 95)
3. Ali - Rata-rata: 85.00 (Matematika: 85, Fisika: 90, Kimia: 80)
4. Budi - Rata-rata: 75.00 (Matematika: 70, Fisika: 75, Kimia: 80)
5. Doni - Rata-rata: 65.00 (Matematika: 60, Fisika: 65, Kimia: 70)
5. Doni - Rata-rata: 65.00 (Matematika: 60, Fisika: 65, Kimia: 70)
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM :

Program tersebut adalah program mengelola data mahasiswa, menghitung rata-rata nilai dan mengurutkan data berdasarkan nilai rata-rata. Program mengelola data mahasiswa dan menghitung rata-rata nilai dari tiga mata pelajaran, lalu mengurutkan data mahasiswa berdasarkan nilai rata-rata secara menurun. Pada struct Mahasiswa, program menyimpan data tiap mahasiswa, lalu menggunakan fungsi hitungRataRata untuk menghitung rata-rata nilai mereka. Pada fungsi utama, data mahasiswa diisi lalu dihitung rata-ratanya, kemudian diurutkan menggunakan sort.Slice. Setelah itu, program menampilkan peringkat mahasiswa berdasarkan rata-rata nilai beserta detail nilai tiap mata pelajaran. Program menampilkan hasil akademis mahasiswa dari nilai tertinggi hingga nilai terendah.

2) **SOURCE CODE**

```go
package main

import "fmt"

func main() {
    // Membuat map dengan NIM sebagai kunci dan Nama sebagai nilai
    mahasiswa := map[string]string{
        "20231001": "Andi",
        "20231002": "Budi",
        "20231003": "Cici",
    }

    // Menambahkan data baru ke map
    mahasiswa["20231004"] = "Dedi"

    // Menampilkan seluruh isi map dalam format kolom dan baris
    fmt.Println("Daftar Mahasiswa:")
    fmt.Println("NIM\t\tNama")
    fmt.Println("------------------------")
    for nim, nama := range mahasiswa {
        fmt.Printf("%s\t%s\n", nim, nama)
    }

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

    // Menghapus data berdasarkan NIM
    delete(mahasiswa, "20231003")

    // Menampilkan isi map setelah data dihapus dalam format kolom dan baris
    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)
    }
}
```

SCREENSHOT OUTPUT :

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
Guided\Guided2.go"
Daftar Mahasiswa:
NIM             Nama
------------------------
20231001        Andi
20231002        Budi
20231003        Cici
20231004        Dedi

Nama Mahasiswa dengan NIM 20231002 adalah Budi

Daftar Mahasiswa setelah dihapus:
NIM             Nama
------------------------
20231001        Andi
20231002        Budi
20231004        Dedi
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM :

Program tersebut adalah Program menunjukkan cara menyimpan dan mengelola data siswa menggunakan tipe data kartu, menggunakan NIM sebagai kunci dan nama sebagai nilainya. Pertama map bernama "Mahasiswa" dibuat dan diinisialisasi dengan tiga entri yang terdiri dari NIM dan nama mahasiswa. Data baru dengan NIM "20231004" dan nama "Dedi" kemudian ditambahkan ke dalam map. Program ini menggunakan loop for-range untuk melakukan iterasi melalui setiap pasangan nilai kunci di peta dan menampilkan semua data yang ada dalam format tabel dengan kolom NIM dan Nama. Program kemudian mengakses data siswa menggunakan NIM tertentu ("20231002") dan mengeluarkan nama mahasiswa yang sesuai. Selanjutnya, program menggunakan fungsi delete() untuk menghapus data NIM "20231003". Setelah dihapus, program akan menampilkan kembali seluruh isi map dan menampilkan perubahan yang terjadi. Dimana NIM "20231003" tidak lagi muncul dalam daftar.

III. UNGUIDED

1. Suatu lingkaran didefinisikan dengan koordinat titilą 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 dl dalam lingkaran 1 dan 2", 'Titik dl dalam llngkaran 1", ”Titik di dalam llngkaran 2",** atau **'Titik dl luar llngkaran 1 dan 2". Contoh**

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:

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

dan juga fungsi untuk menentukan posisi sebuah titilk 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}
```

SOURCE CODE

```
package main

import (
    "fmt"
    "math"
)

type Point struct {
    x, y int
}

type Circle struct {
    center Point
    radius int
}

func distance(p1, p2 Point) float64 {
    return math.Sqrt(float64((p1.x-p2.x)*(p1.x-p2.x) + (p1.y-
p2.y)*(p1.y-p2.y)))
}

func isInCircle(p Point, c Circle) bool {
    return distance(p, c.center) <= float64(c.radius)
}

func main() {
    var c1, c2 Circle
    var p Point

    fmt.Scan(&c1.center.x, &c1.center.y, &c1.radius)
    fmt.Scan(&c2.center.x, &c2.center.y, &c2.radius)
    fmt.Scan(&p.x, &p.y)

    if isInCircle(p, c1) && isInCircle(p, c2) {
        fmt.Println("Titik di dalam lingkaran 1 dan 2")
    } else if isInCircle(p, c1) {
        fmt.Println("Titik di dalam lingkaran 1")
    } else if isInCircle(p, c2) {
        fmt.Println("Titik di dalam lingkaran 2")
    } else {
        fmt.Println("Titik di luar lingkaran 1 dan 2")
    }
}
```

SCREENSHOT OUTPUT

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
3 3 7
6 6 2
4 4
Titik di dalam lingkaran 1
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
1 2 3
4 5 6
7 8
Titik di dalam lingkaran 2
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
10 15 20
-5 2 10
2 2
Titik di dalam lingkaran 1 dan 2
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
1 1 5
8 8 4
15 20
Titik di luar lingkaran 1 dan 2
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM

Program tersebut adalah program menentukan apakah titik berada didalam salah satu atau kedua lingkaran yang didefinisikan. Pada program tersebut terdapat dua definisi struktur data yaitu Point untuk menyimpan koordinat titik (x, y) dan circle untuk merepresentasikan lingkaran yang terdiri dari center sebagai titik pusat dan radius sebagai jari-jari lingkaran. Terdapat fungsi distance untuk menghitung jarak antara dua titik menggunakan rumus Euclidean serta fungsi isInCircle yang memeriksa apakah jarak dari suatu titik ke pusat lingkaran lebih kecil atau sama dengan jari-jari lingkaran, menandakan bahwa titik berada di dalam lingkaran tersebut. Pada fungsi utama program meminta user memasukkan data dua lingkaran dan satu titik Program kemudian menggunakan fungsi isInCircle untuk menentukan apakah titik tersebut berada di dalam salah satu, kedua, atau di luar kedua lingkaran, dan menampilkan hasil yang sesuai.

2. 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:

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. Menampilląan 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.

SOURCE CODE

```go
package main

import (
    "fmt"
    "math"
)

func main() {
    var N, x, indexHapus, target int
    fmt.Print("Masukkan jumlah elemen array: ")
    fmt.Scan(&N)

    arr := make([]int, N)
    for i := range arr {
        fmt.Printf("Masukkan elemen ke-%d: ", i)
        fmt.Scan(&arr[i])
    }

    fmt.Println("Isi array:", arr)

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

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

    fmt.Print("Masukkan bilangan x untuk indeks kelipatan: ")
    fmt.Scan(&x)
    fmt.Printf("Elemen dengan indeks kelipatan %d: ", x)
    for i := 0; i < N; i += x {
        fmt.Print(arr[i], " ")
    }
    fmt.Println()

    fmt.Print("Masukkan indeks yang ingin dihapus: ")
    fmt.Scan(&indexHapus)
    arr = append(arr[:indexHapus], arr[indexHapus+1:]...)
    fmt.Println("Isi array setelah penghapusan:", arr)

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

    var varianceSum float64
    for _, v := range arr {
        varianceSum += math.Pow(float64(v)-average, 2)
    }
    stdDev := math.Sqrt(varianceSum / float64(len(arr)))
    fmt.Printf("Standar deviasi: %.2f\n", stdDev)

    fmt.Print("Masukkan bilangan untuk frekuensi: ")
    fmt.Scan(&target)
    count := 0
    for _, v := range arr {
        if v == target {
            count++
        }
    }
    fmt.Printf("Frekuensi dari bilangan %d: %d kali\n", target,
count)
}
```

SCREENSHOT OUTPUT

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run
Unguided\Unguded2.go"
Masukkan jumlah elemen array: 10
Masukkan elemen ke-0: 2
Masukkan elemen ke-1: 3
Masukkan elemen ke-2: 1
Masukkan elemen ke-3: 1
Masukkan elemen ke-4: 1
Masukkan elemen ke-5: 0
Masukkan elemen ke-6: 2
Masukkan elemen ke-7: 0
Masukkan elemen ke-8: 7
Masukkan elemen ke-9: 5
Isi array: [2 3 1 1 1 0 2 0 7 5]
Elemen dengan indeks ganjil: 3 1 0 0 5
Elemen dengan indeks genap: 2 1 1 2 7
Masukkan bilangan x untuk indeks kelipatan: 5
Elemen dengan indeks kelipatan 5: 2 0
Masukkan indeks yang ingin dihapus: 2
Isi array setelah penghapusan: [2 3 1 1 0 2 0 7 5]
Rata-rata: 2.33
Standar deviasi: 2.21
Masukkan bilangan untuk frekuensi: 2
Frekuensi dari bilangan 2: 2 kali
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM

Program tersebut adalah program memasukkan sejumlah elemen integer ke dalam array, menampilkan elemen-elemen berdasarkan indeks ganjil, genap, dan kelipatan tertentu. Pertama, program meminta jumlah elemen, lalu mengisi array tersebut. Setelah itu, program menampilkan elemen pada indeks ganjil dan genap, kemudian meminta pengguna menentukan kelipatan indeks tertentu dan menampilkan elemen pada indeks tersebut. Program juga memungkinkan pengguna menghapus elemen di indeks tertentu menggunakan fungsi append. Selanjutnya, program menghitung rata-rata semua elemen, menampilkannya dengan format dua desimal, lalu menghitung variansi dan standar deviasi. Terakhir, program menghitung dan menampilkan frekuensi kemunculan bilangan tertentu yang dimasukkan 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.

```
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
```

SOURCE CODE

```
package main

import (
    "fmt"
)

func main() {
    var klubA, klubB string

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

    var pertandingan []string

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

        if skorA < 0 || skorB < 0 {
            break
        }

        if skorA > skorB {
            pertandingan = append(pertandingan,
fmt.Sprintf("Hasil %d : %s", i, klubA))
        } else if skorB > skorA {
            pertandingan = append(pertandingan,
fmt.Sprintf("Hasil %d : %s", i, klubB))
        } else {
            pertandingan = append(pertandingan,
fmt.Sprintf("Hasil %d : Draw", i))
        }
    }

    fmt.Println()
    for _, hasil := range pertandingan {
        fmt.Println(hasil)
    }

    fmt.Println("Pertandingan selesai")
}
```

SCREENSHOT OUTPUT

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
Klub A : Liverpool
Klub B : Barcelona

Pertandingan 1 : 2 0
Pertandingan 2 : 0 3
Pertandingan 3 : 3 3
Pertandingan 4 : 1 3
Pertandingan 5 : 5 2
Pertandingan 6 : 0 0
Pertandingan 7 : -1 2

Hasil 1 : Liverpool
Hasil 2 : Barcelona
Hasil 3 : Draw
Hasil 4 : Barcelona
Hasil 5 : Liverpool
Hasil 6 : Draw
Pertandingan selesai
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM

Program tersebut adalah program untuk mencatat dan menampilkan hasil-hasil pertandingan antara dua klub sepak bola. Pertama, program meminta pengguna untuk memasukkan nama-nama klub yang akan bertanding. Kemudian, program akan terus meminta input skor pertandingan hingga salah satu atau kedua skor adalah negatif. Setelah itu, program akan menentukan pemenang pertandingan berdasarkan skor yang dimasukkan, dan menyimpan hasilnya dalam format tertentu. Terakhir, program akan menampilkan semua hasil pertandingan yang telah disimpan sebelumnya dan menampilkan akhir output pertandingan selesai. Dengan demikian, pengguna dapat dengan mudah merekap dan melihat hasil-hasil pertandingan antara dua klub.

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. Lengkapilah 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          : S E N A N G .
Reverse teks  : G N A N E S

Teks          : K A T A K .
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      : K A T A K
Palindrom ? true

Teks      : S E N A N G
Palindrom ? false
```

SOURCE CODE

```go
package main

import (
    "fmt"
    "strings"
)

const NMAX int = 127

type Tabel struct {
    tab [NMAX]rune
    m   int
}

func IsiArray(t *Tabel, n *int) {
    fmt.Println("Masukkan teks :")
    var input string
    fmt.Scanln(&input)

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

func printArray(t Tabel, n int) {
    for i := 0; i < n; i++ {
        fmt.Print(string(t.tab[i]))
    }
    fmt.Println()
}

func reverseArray(t *Tabel, n int) {
```

```
    for i := 0; i < n/2; i++ {
        t.tab[i], t.tab[n-i-1] = t.tab[n-i-1], t.tab[i]
    }
}

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

func main() {
    var t Tabel
    var n int
    IsiArray(&t, &n)
    fmt.Print("Teks: ")
    printArray(t, n)
    fmt.Print("Reverse teks: ")
    reverseArray(&t, n)
    printArray(t, n)

    if isPalindrome(t, n) {
        fmt.Println("Palindrome: true")
    } else {
        fmt.Println("Palindrome: false")
    }
}
```

SCREENSHOT OUTPUT

```
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run "
c:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code\Unguided\Ung
uided4.go"
Masukkan teks :
Sapi
Teks: Sapi
Reverse teks: ipaS
Palindrome: false
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code> go run "
c:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code\Unguided\Ung
uided4.go"
Masukkan teks :
Katak
Teks: Katak
Reverse teks: kataK
Palindrome: true
PS C:\Users\ACER\Documents\Semester 3\Algoritma Pemrograman 2\Praktikum\Modul 7\Source code>
```

PENJELASAN PROGRAM

Program tersebut adalah program untuk memeriksa apakah sebuah teks yang dimasukkan oleh pengguna merupakan palindrom atau tidak. Pertama Program meminta pengguna untuk memasukkan teks, lalu menyimpannya dalam struktur data Tabel yang terdiri dari sebuah array tab untuk menyimpan karakter teks, serta variabel m untuk menyimpan jumlah karakter yang dimasukkan. Selanjutnya, program akan membalikkan urutan karakter dalam teks menggunakan fungsi reverseArray dan menampilkannya. Setelah itu, program akan memeriksa apakah teks tersebut membentuk palindrom dengan cara membandingkan karakter pada indeks awal dan akhir menggunakan fungsi IsPalindrome. Jika semua pasangan karakter cocok, maka teks dianggap sebagai palindrom dan program akan menampilkan hasilnya.