

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VII**

**STRUCK & ARRAY**



**Disusun Oleh :**

**Nama lengkap / NIM**

**Kelas**

**Dosen Pengampu :**

**-----**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## 1. DASAR TEORI

Dalam pemrograman, **struct** (struktur) adalah tipe data yang memungkinkan kita untuk mengelompokkan beberapa variabel yang berbeda tipe dalam satu kesatuan data. Penggunaan struct sangat berguna ketika kita ingin merepresentasikan suatu entitas atau objek yang memiliki beberapa atribut terkait. Sebagai contoh, jika kita ingin menyimpan data seorang mahasiswa, kita mungkin perlu menyimpan informasi seperti nama, nim (nomor induk mahasiswa), dan nilai. Dengan menggunakan struct, kita bisa mengelompokkan atribut-atribut ini dalam satu wadah, sehingga lebih mudah untuk dikelola dan diakses. Setiap bagian dalam struct disebut "field" atau "member".

**Array**, di sisi lain, adalah struktur data yang digunakan untuk menyimpan kumpulan elemen yang memiliki tipe data yang sama. Array menyimpan elemen secara berurutan, dan setiap elemen bisa diakses menggunakan indeks, yang dimulai dari 0 untuk elemen pertama. Array banyak digunakan saat kita perlu menyimpan data dalam jumlah besar yang memiliki tipe data seragam, seperti deretan angka atau daftar nama. Array dapat berbentuk satu dimensi atau lebih, seperti array dua dimensi yang dapat digunakan untuk merepresentasikan matriks atau tabel.

Kedua konsep ini, struct dan array, dapat digunakan bersama untuk menyimpan dan mengelola data yang lebih kompleks. Misalnya, jika kita ingin menyimpan data dari beberapa mahasiswa, kita bisa membuat array dari struct mahasiswa. Dengan cara ini, setiap elemen dalam array merepresentasikan data lengkap dari satu mahasiswa, sementara seluruh array merepresentasikan kumpulan data mahasiswa. Dengan kombinasi struct dan array, kita dapat menyusun data yang lebih kompleks namun tetap terorganisir, terutama dalam program yang membutuhkan pengelolaan data dalam jumlah besar.

Secara keseluruhan, **struct** dan **array** adalah dua konsep penting yang saling melengkapi dalam pemrograman. Struct memungkinkan kita menyimpan data yang saling terkait tetapi berbeda tipe dalam satu unit, sementara array memudahkan penyimpanan data bertipe sama dalam jumlah banyak secara terurut. Kombinasi keduanya memungkinkan kita mengelola data yang lebih kompleks secara efisien dan terstruktur.

## I. GUIDED

### 1. Soal Studi Case

#### Sourcecode

```
package main

import (
    "fmt"
    "math"
)

type titik struct {
    x int
    y int
}

type lingkaran struct {
    pusat titik
    radius int
}

func hitungjarak(a, b titik) float64 {
    return math.Sqrt(float64((a.x-b.x)*(a.x-b.x) + (a.y-b.y)*(a.y-b.y)))
}

func titikdalamlingkaran(t titik, l lingkaran) bool {
    jarak := hitungjarak(t, l.pusat)
    return jarak <= float64(l.radius)
}

func main() {
    var cx1, cy1, r1 int
    fmt.Print("Masukkan koordinator pusat dan radius lingkaran 1 (cx1, cy1, r1): ")
    fmt.Scanln(&cx1, &cy1, &r1)
    lingkaran1 := lingkaran{pusat: titik{x: cx1, y: cy1}, radius: r1}

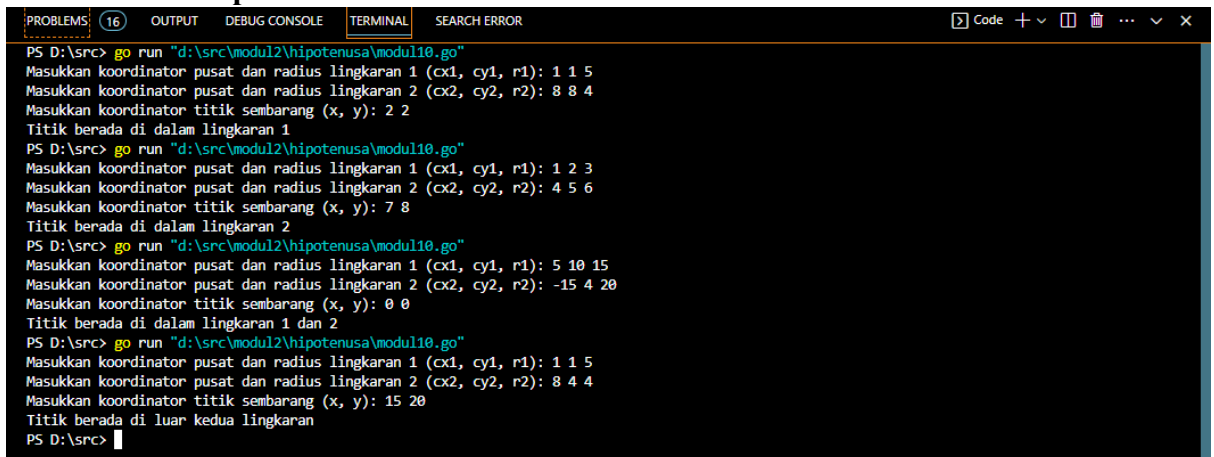
    var cx2, cy2, r2 int
    fmt.Print("Masukkan koordinator pusat dan radius lingkaran 2 (cx2, cy2, r2): ")
    fmt.Scanln(&cx2, &cy2, &r2)
    lingkaran2 := lingkaran{pusat: titik{x: cx2, y: cy2}, radius: r2}

    var x, y int
    fmt.Print("Masukkan koordinator titik sembarang (x, y): ")
    fmt.Scanln(&x, &y)
    titikSembarang := titik{x: x, y: y}
```

```
didalam1 := titikdalamlingkaran(titikSembarang, lingkaran1)
didalam2 := titikdalamlingkaran(titikSembarang, lingkaran2)
```

```
if didalam1 && didalam2 {
    fmt.Println("Titik berada di dalam lingkaran 1 dan 2")
} else if didalam1 {
    fmt.Println("Titik berada di dalam lingkaran 1")
} else if didalam2 {
    fmt.Println("Titik berada di dalam lingkaran 2")
} else {
    fmt.Println("Titik berada di luar kedua lingkaran")
}
}
```

## Screenshot Output



```
PS D:\src> go run "d:\src\modul2\hipotenusa\modul10.go"
Masukkan koordinat pusat dan radius lingkaran 1 (cx1, cy1, r1): 1 1 5
Masukkan koordinat pusat dan radius lingkaran 2 (cx2, cy2, r2): 8 8 4
Masukkan koordinat titik sembarang (x, y): 2 2
Titik berada di dalam lingkaran 1
PS D:\src> go run "d:\src\modul2\hipotenusa\modul10.go"
Masukkan koordinat pusat dan radius lingkaran 1 (cx1, cy1, r1): 1 2 3
Masukkan koordinat pusat dan radius lingkaran 2 (cx2, cy2, r2): 4 5 6
Masukkan koordinat titik sembarang (x, y): 7 8
Titik berada di dalam lingkaran 2
PS D:\src> go run "d:\src\modul2\hipotenusa\modul10.go"
Masukkan koordinat pusat dan radius lingkaran 1 (cx1, cy1, r1): 5 10 15
Masukkan koordinat pusat dan radius lingkaran 2 (cx2, cy2, r2): -15 4 20
Masukkan koordinat titik sembarang (x, y): 0 0
Titik berada di dalam lingkaran 1 dan 2
PS D:\src> go run "d:\src\modul2\hipotenusa\modul10.go"
Masukkan koordinat pusat dan radius lingkaran 1 (cx1, cy1, r1): 1 1 5
Masukkan koordinat pusat dan radius lingkaran 2 (cx2, cy2, r2): 8 4 4
Masukkan koordinat titik sembarang (x, y): 15 20
Titik berada di luar kedua lingkaran
PS D:\src>
```

## Deskripsi Program

Program ini bekerja dengan menerima input dari pengguna untuk mendefinisikan dua lingkaran dan satu titik sembarang. Berdasarkan posisi titik sembarang tersebut, program akan memeriksa apakah titik tersebut berada di dalam lingkaran pertama, lingkaran kedua, keduanya, atau di luar kedua lingkaran.

Pertama, program menggunakan dua struct (titik dan lingkaran) untuk merepresentasikan konsep titik dan lingkaran. Struct titik memiliki dua atribut, x dan y, yang mewakili koordinat titik dalam ruang dua dimensi. Struct lingkaran memiliki atribut pusat (yang bertipe titik) dan radius (bertipe int), yang menunjukkan pusat dan jari-jari lingkaran. Struct ini memudahkan penyimpanan data terkait yang relevan dalam satu kesatuan.

## UNGUIDED

1.

### Sourcecode

```
package main

import (
    "fmt"
    "math"
)

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

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

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

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

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

    var x int
    fmt.Print("Masukkan bilangan untuk kelipatan indeks: ")
    fmt.Scanln(&x)
    fmt.Printf("Elemen pada indeks kelipatan %d: ", x)
    for i := x; i < N; i += x {
        fmt.Print(array[i], " ")
    }
    fmt.Println()
}
```

```

var idx int
fmt.Print("Masukkan indeks elemen yang ingin dihapus: ")
fmt.Scanln(&idx)
if idx >= 0 && idx < N {
    array = append(array[:idx], array[idx+1:]...)
    fmt.Println("Isi array setelah penghapusan:", array)
} else {
    fmt.Println("Indeks tidak valid.")
}

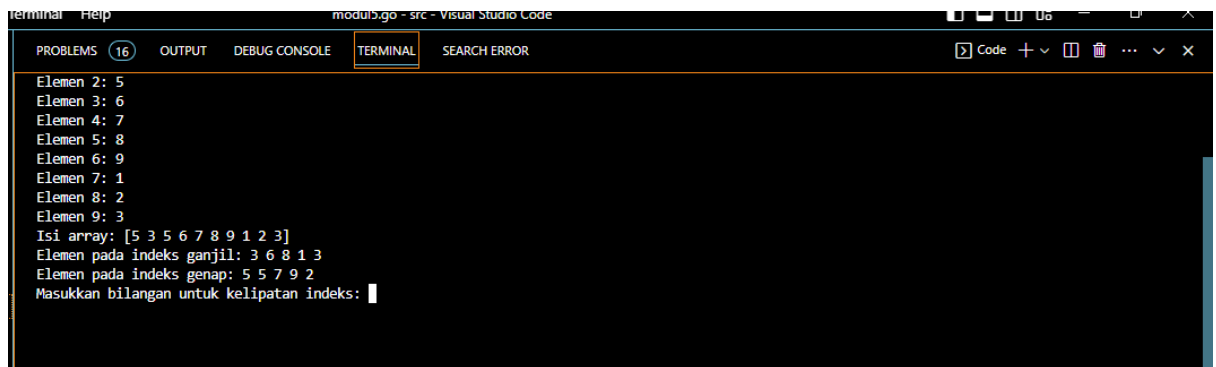
sum := 0
for _, value := range array {
    sum += value
}
rataRata := float64(sum) / float64(len(array))
fmt.Printf("Rata-rata elemen array: %.2f\n", rataRata)

var varianceSum float64
for _, value := range array {
    varianceSum += math.Pow(float64(value)-rataRata, 2)
}
stdDeviasi := math.Sqrt(varianceSum / float64(len(array)))
fmt.Printf("Standar deviasi elemen array: %.2f\n", stdDeviasi)

var cari int
fmt.Print("Masukkan bilangan yang ingin dicari frekuensinya: ")
fmt.Scanln(&cari)
frekuensi := 0
for _, value := range array {
    if value == cari {
        frekuensi++
    }
}
fmt.Printf("Frekuensi bilangan %d: %d\n", cari, frekuensi)
}

```

## Screenshot Output



```
terminal Help moduls.go - src - Visual Studio Code
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL SEARCH ERROR
Elemen 2: 5
Elemen 3: 6
Elemen 4: 7
Elemen 5: 8
Elemen 6: 9
Elemen 7: 1
Elemen 8: 2
Elemen 9: 3
Isi array: [5 3 5 6 7 8 9 1 2 3]
Elemen pada indeks ganjil: 3 6 8 1 3
Elemen pada indeks genap: 5 5 7 9 2
Masukkan bilangan untuk kelipatan indeks: 
```

## Deskripsi Program

Program ini merupakan implementasi array dalam bahasa Go yang memungkinkan pengguna untuk melakukan berbagai operasi dasar pada sekumpulan bilangan bulat. Setelah menentukan jumlah dan elemen-elemen array, pengguna dapat menampilkan seluruh isi array, elemen pada indeks ganjil atau genap, dan elemen pada indeks kelipatan suatu bilangan tertentu. Program juga mendukung penghapusan elemen pada indeks tertentu, menghitung rata-rata dan simpangan baku elemen-elemen dalam array, serta mencari frekuensi kemunculan suatu bilangan dalam array. Setiap operasi dirancang agar pengguna dapat dengan mudah mengelola dan menganalisis data yang ada dalam array tersebut.

2.

```
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
    pertandingan := 1

    for {

        var skorA, skorB int
        fmt.Printf("Masukkan skor pertandingan %d (format: skor %s skor %s): ",
            pertandingan, klubA, klubB)
```

```

    fmt.Scanf("%d %d\n", &skorA, &skorB)

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

    if skorA > skorB {
        pemenang = append(pemenang, klubA)
    } else if skorB > skorA {
        pemenang = append(pemenang, klubB)
    } else {
        pemenang = append(pemenang, "Seri") // Skor sama dianggap seri
    }
    pertandingan++
}

fmt.Println("\nHasil pertandingan:")
for i, p := range pemenang {
    if p == "Seri" {
        fmt.Printf("Pertandingan %d: Seri\n", i+1)
    } else {
        fmt.Printf("Pertandingan %d dimenangkan oleh: %s\n", i+1, p)
    }
}
}

```

Scrennshoot output

```

PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL SEARCH ERROR
Masukkan skor pertandingan 1 (format: skor MU skor MADRID): 1 0
Masukkan skor pertandingan 2 (format: skor MU skor MADRID): 2 1
Masukkan skor pertandingan 3 (format: skor MU skor MADRID): 3 2
Masukkan skor pertandingan 4 (format: skor MU skor MADRID): 4 3
Masukkan skor pertandingan 5 (format: skor MU skor MADRID): -1 -1

Hasil pertandingan:
Pertandingan 1 dimenangkan oleh: MU
Pertandingan 2 dimenangkan oleh: MU
Pertandingan 3 dimenangkan oleh: MU
Pertandingan 4 dimenangkan oleh: MU
PS D:\src>

```

Deskripsi program

Program ini memungkinkan pencatatan hasil pertandingan dengan mudah dan akan berhenti menerima input ketika skor negatif dimasukkan.



3.

```
package main

import (
    "fmt"
    "strings"
)

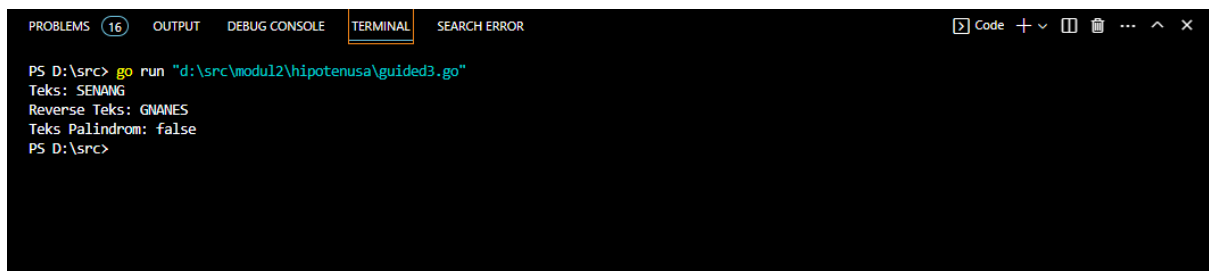
func main() {
    teks := "SENANG"
    reverseTeks := reverseString(teks)
    fmt.Println("Teks:", teks)
    fmt.Println("Reverse Teks:", reverseTeks)

    isPalindrom := isPalindrome(teks)
    fmt.Println("Teks Palindrom:", isPalindrom)
}

func reverseString(s string) string {
    runes := []rune(s)
    for i, j := 0, len(runes)-1; i < j; i, j = i+1, j-1 {
        runes[i], runes[j] = runes[j], runes[i]
    }
    return string(runes)
}

func isPalindrome(s string) bool {
    s = strings.ToLower(s)
    return s == reverseString(s)
}
```

### Screenshoot program

A screenshot of a Visual Studio Code terminal window. The terminal shows the execution of a Go program. The command entered is 'go run "d:\src\modul2\hipotenusa\guided3.go"'. The output of the program is displayed as follows: 'Teks: SENANG', 'Reverse Teks: GNANES', and 'Teks Palindrom: false'. The terminal window has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'SEARCH ERROR'. The 'TERMINAL' tab is active. The terminal title bar shows 'Code' and various window control icons.

### Deskripsi program

Program ini menampilkan string yang dibalik, juga memeriksa apakah string tersebut merupakan palindrom. Program ini menggunakan fungsi **reverseString()** untuk membalik

string input, dan fungsi **isPalindrome()** untuk memeriksa apakah string tersebut merupakan palindrom.