

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2  
MODUL 7  
STRUCT DAN ARRAY**



Oleh:

MUHAMMAD FAUZAN

103112400064

12 IF 01

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

## **I. DASAR TEORI**

### **1. Array**

#### **Definisi**

Array adalah kumpulan elemen dengan tipe data yang sama dan ukuran tetap. Ukuran array merupakan bagian dari tipe datanya, sehingga [5]int berbeda dengan [10]int. Setelah dideklarasikan, ukuran array tidak dapat diubah.

#### **Karakteristik**

- **Ukuran tetap:** Ditentukan saat deklarasi dan tidak dapat diubah.
- **Indeks dimulai dari 0:** Elemen pertama berada pada indeks 0.
- **Nilai:** Array adalah nilai; menyalin array akan menyalin seluruh isinya.

#### **Kelebihan**

- **Performa tinggi:** Karena elemen disimpan secara kontigu di memori.
- **Efisien untuk akses indeks:** Cocok untuk manipulasi data berbasis indeks.

#### **Kekurangan**

- **Kurang fleksibel:** Ukuran tetap membuatnya kurang cocok untuk data dinamis.
- **Kurang praktis:** Untuk operasi yang memerlukan penambahan atau penghapusan elemen, array kurang efisien.

## 2. Slice

### Definisi

Slice adalah abstraksi dari array yang memungkinkan ukuran dinamis. Slice tidak menyimpan data sendiri, melainkan merujuk ke bagian dari array. [Go](#)

### Karakteristik

- **Ukuran dinamis:** Dapat bertambah atau berkurang selama runtime.
- **Referensi ke array:** Slice adalah referensi ke array yang mendasarinya.
- **Fungsi make:** Slice dapat dibuat menggunakan fungsi make. [Go](#)

### Kelebihan

- **Fleksibel:** Ukuran dapat berubah sesuai kebutuhan.
- **Fungsi bawaan:** Go menyediakan fungsi bawaan seperti append untuk memanipulasi slice. [Go](#)

### Kekurangan

- **Overhead:** Karena slice adalah referensi, perubahan pada slice dapat memengaruhi array yang mendasarinya.

### 3. Struct

#### Definisi

Struct adalah tipe data komposit yang mengelompokkan beberapa field dengan tipe data yang berbeda menjadi satu entitas. Struct digunakan untuk merepresentasikan objek atau entitas dengan berbagai atribut.

#### Karakteristik

- **Field dengan tipe berbeda:** Setiap field dapat memiliki tipe data yang berbeda.
- **Field unik:** Nama field dalam satu struct harus unik.
- **Embedded fields:** Struct dapat memiliki field yang merupakan struct lain, memungkinkan komposisi dan pewarisan perilaku.

#### Kelebihan

- **Representasi objek nyata:** Cocok untuk merepresentasikan entitas dunia nyata seperti mahasiswa, produk, atau buku.
- **Dapat memiliki metode:** Struct dapat memiliki metode yang didefinisikan untuk tipe tersebut.

#### Kekurangan

- **Tidak mendukung pewarisan:** Go tidak mendukung pewarisan kelas seperti dalam bahasa pemrograman berorientasi objek lainnya.
- **Komposisi lebih disarankan:** Go mendorong penggunaan komposisi daripada pewarisan untuk membangun hierarki tipe.

## 4. Map

### Definisi

Map adalah koleksi pasangan key-value, di mana setiap key unik dan digunakan untuk mengakses nilai yang terkait.

### Karakteristik

- **Key unik:** Setiap key dalam map harus unik dan dapat dibandingkan.
- **Nilai dapat diakses dengan key:** Nilai dalam map diakses menggunakan key yang sesuai.
- **Fungsi make:** Map harus diinisialisasi menggunakan fungsi `make` sebelum digunakan. [Go](#)

### Kelebihan

- **Akses cepat:** Operasi pencarian, penambahan, dan penghapusan elemen dalam map sangat efisien.
- **Fleksibel:** Map dapat digunakan untuk berbagai keperluan, seperti menyimpan konfigurasi atau data yang tidak terstruktur.

### Kekurangan

- **Tidak terurut:** Map tidak menjamin urutan elemen saat iterasi.
- **Key harus dapat dibandingkan:** Tipe data yang digunakan sebagai key harus dapat dibandingkan (comparable).

## II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

Contoh 1

```
package main

import (
    "fmt"
    "time"
)

// Struct untuk barang dalam struk belanja
type Item struct {
    Name   string
    Price  float64
    Quantity int
}

// Struct untuk struk belanja
type Receipt struct {
    StoreInfo string
    Date      time.Time
    Items     []Item
    TotalAmount float64
}

// Method untuk menghitung total harga semua item
func (r *Receipt) CalculateTotal() {
    var total float64
    for _, item := range r.Items {
        total += item.Price * float64(item.Quantity)
    }
    r.TotalAmount = total
}

// Method untuk mencetak struk belanja
func (r Receipt) PrintReceipt() {
    fmt.Println("=====")
    fmt.Println(r.StoreInfo)
    fmt.Println("Tanggal:", r.Date.Format("02-01-2006 15:04"))
    fmt.Println("=====")
    fmt.Printf("%-15s %-10s %-8s %-10s\n", "Item", "Harga", "Jumlah",
```

```

"Total")
    fmt.Println("-----")

    for _, item := range r.Items {
        itemTotal := item.Price * float64(item.Quantity)
        fmt.Printf("%-15s Rp%-9.2f %-8d Rp%-9.2f\n", item.Name,
item.Price, item.Quantity, itemTotal)
    }

    fmt.Println("=====")
    fmt.Printf("%-35s Rp%-9.2f\n", "Total Belanja:", r.TotalAmount)
    fmt.Println("=====")
    fmt.Println("Terima kasih telah berbelanja!")
}

func main() {
    receipt := Receipt{
        StoreInfo: "Toko Sembako Makmur\nJl. Raya No. 123, Jakarta",
        Date:      time.Now(),
        Items: []Item{
            {Name: "Beras", Price: 12000, Quantity: 5},
            {Name: "Gula", Price: 15000, Quantity: 2},
            {Name: "Minyak", Price: 20000, Quantity: 1},
            {Name: "Telur", Price: 2000, Quantity: 10},
        },
    }

    receipt.CalculateTotal()
    receipt.PrintReceipt()
}

```

## Screenshots Output

```
● =====  
Toko Sembako Makmur  
Jl. Raya No. 123, Jakarta  
Tanggal: 24-04-2025 17:16  
=====
```

Item	Harga	Jumlah	Total
Beras	Rp12000.00	5	Rp60000.00
Gula	Rp15000.00	2	Rp30000.00
Minyak	Rp20000.00	1	Rp20000.00
Telur	Rp2000.00	10	Rp20000.00

```
=====
```

Total Belanja:			Rp130000.00
----------------	--	--	-------------

```
=====
```

Terima kasih telah berbelanja!

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)>

// Foto hasil dari menjalankan code

Deskripsi: Program ini adalah simulasi struk belanja menggunakan bahasa pemrograman Go (Golang) yang memanfaatkan konsep struct dan method. Tujuan utama program ini adalah menunjukkan bagaimana Go dapat digunakan untuk membuat sistem pencatatan transaksi sederhana yang efisien dan terstruktur.



## Contoh 2

```
package main

import (
    "fmt"
)

func main() {
    // Deklarasi dan inisialisasi array nilai mahasiswa
    nilaiMahasiswa := [5]int{85, 90, 78, 88, 95}

    fmt.Println("Data Nilai Mahasiswa:")
    fmt.Println("=====")

    // Menampilkan nilai per mahasiswa
    for i, nilai := range nilaiMahasiswa {
        fmt.Printf("Mahasiswa %d: %d\n", i+1, nilai)
    }

    // Menghitung rata-rata nilai
    var total int
    for _, nilai := range nilaiMahasiswa {
        total += nilai
    }
    rataRata := float64(total) / float64(len(nilaiMahasiswa))

    fmt.Println("=====")
    fmt.Printf("Rata-rata nilai: %.2f\n", rataRata)

    // Mencari nilai tertinggi dan terendah
    tertinggi := nilaiMahasiswa[0]
    terendah := nilaiMahasiswa[0]

    for _, nilai := range nilaiMahasiswa {
        if nilai > tertinggi {
            tertinggi = nilai
        }
        if nilai < terendah {
            terendah = nilai
        }
    }

    fmt.Printf("Nilai tertinggi: %d\n", tertinggi)
```

```

    fmt.Printf("Nilai terendah: %d\n", terendah)

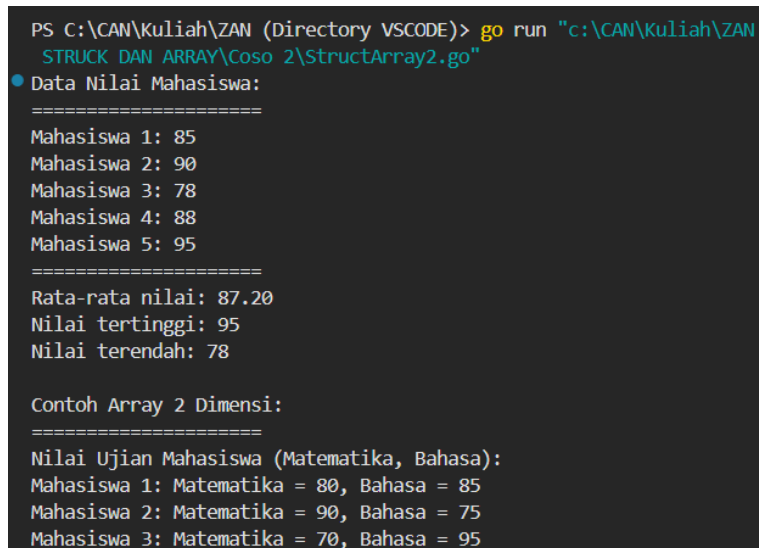
    // Contoh array 2 dimensi
    fmt.Println("\nContoh Array 2 Dimensi:")
    fmt.Println("=====")

    // Nilai ujian mahasiswa dalam 2 mata kuliah (Matematika, Bahasa)
    nilaiUjian := [3][2]int{
        {80, 85},
        {90, 75},
        {70, 95},
    }

    // Menampilkan nilai ujian per mahasiswa
    fmt.Println("Nilai Ujian Mahasiswa (Matematika, Bahasa):")
    for i, nilai := range nilaiUjian {
        fmt.Printf("Mahasiswa %d: Matematika = %d, Bahasa = %d\n",
i+1, nilai[0], nilai[1])
    }
}

```

### Screenshots Output



```

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN
STRUCK DAN ARRAY\Coso 2\StructArray2.go"
• Data Nilai Mahasiswa:
=====
Mahasiswa 1: 85
Mahasiswa 2: 90
Mahasiswa 3: 78
Mahasiswa 4: 88
Mahasiswa 5: 95
=====
Rata-rata nilai: 87.20
Nilai tertinggi: 95
Nilai terendah: 78

Contoh Array 2 Dimensi:
=====
Nilai Ujian Mahasiswa (Matematika, Bahasa):
Mahasiswa 1: Matematika = 80, Bahasa = 85
Mahasiswa 2: Matematika = 90, Bahasa = 75
Mahasiswa 3: Matematika = 70, Bahasa = 95

```

// Foto hasil dari menjalankan code

Deskripsi: Program ini bertujuan untuk memperlihatkan cara penggunaan array satu dimensi dan dua dimensi dalam bahasa Go untuk mengelola data nilai mahasiswa.

### III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

#### Soal 1

```
// MUHAMMAD FAUZAN
//103112400064
package main

import (
    "fmt"
    "math"
)

func dalamLingkaran(x, y, r, xt, yt int) bool {
    d := math.Sqrt(float64((xt-x)*(xt-x) + (yt-y)*(yt-y)))
    return d <= float64(r)
}

func main() {
    var x1, y1, r1 int
    var x2, y2, r2 int
    var xt, yt int

    fmt.Scan(&x1, &y1, &r1)
    fmt.Scan(&x2, &y2, &r2)
    fmt.Scan(&xt, &yt)

    dalam1 := dalamLingkaran(x1, y1, r1, xt, yt)
    dalam2 := dalamLingkaran(x2, y2, r2, xt, yt)

    if dalam1 && dalam2 {
        fmt.Println("Titik di dalam lingkaran 1 dan 2")
    } else if dalam1 {
        fmt.Println("Titik di dalam lingkaran 1")
    } else if dalam2 {
        fmt.Println("Titik di dalam lingkaran 2")
    } else {
        fmt.Println("Titik di luar lingkaran 1 dan 2")
    }
}
```

## Screenshots Output

```
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\1\103112400064_Unguided1.go"
1 1 5
8 8 4
2 2
Titik di dalam lingkaran 1
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\1\103112400064_Unguided1.go"
1 2 3
4 5 6
7 8
Titik di dalam lingkaran 2
```

// Foto hasil dari menjalankan code

Deskripsi: Kode di atas adalah program dalam bahasa Go yang menggunakan struktur data (struct) untuk memeriksa apakah suatu titik berada di dalam satu atau dua lingkaran.

## Soal 2

```
// MUHAMMAD FAUZAN
//103112400064
package main

import (
    "fmt"
    "math"
)

func main() {
    var n, x, hapusIdx, cari int
    fmt.Print("Jumlah elemen: ")
    fmt.Scan(&n)

    data := make([]int, n)
    fmt.Println("Masukkan elemen:")
    for i := 0; i < n; i++ {
        fmt.Printf("indeks ke-%d: ", i)
        fmt.Scan(&data[i])
    }

    // a. Tampilkan seluruh isi array
    fmt.Print("a. Isi array: ")
    for _, v := range data {
        fmt.Print(v, " ")
    }
    fmt.Println()

    // b. Indeks ganjil
    fmt.Print("b. Indeks ganjil: ")
    for i := 1; i < len(data); i += 2 {
        fmt.Print(data[i], " ")
    }
    fmt.Println()

    // c. Indeks genap
    fmt.Print("c. Indeks genap: ")
    for i := 0; i < len(data); i += 2 {
        fmt.Print(data[i], " ")
    }
    fmt.Println()
}
```

**// d. Indeks kelipatan x**

```
fmt.Print("Masukkan Indeks kelipatan x: ")
fmt.Scan(&x)
fmt.Printf("d. Indeks kelipatan %d: ", x)
for i := 0; i < len(data); i++ {
    if i%x == 0 {
        fmt.Print(data[i], " ")
    }
}
fmt.Println()
```

**// e. Hapus elemen pada indeks tertentu**

```
fmt.Print("Masukan Indeks yang ingin dihapus: ")
fmt.Scan(&hapusIdx)
data = append(data[:hapusIdx], data[hapusIdx+1:]...) // hapus elemen

fmt.Print("e. Array Setelah dihapus: ")
for _, v := range data {
    fmt.Print(v, " ")
}
fmt.Println()
```

**// f. Rata-rata**

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

**// g. Simpangan baku**

```
var jumlahKuadrat float64
for _, v := range data {
    selisih := float64(v) - rata
    jumlahKuadrat += selisih * selisih
}
sd := math.Sqrt(jumlahKuadrat / float64(len(data)))
fmt.Printf("g. Simpangan baku: %.2f\n", sd)
```

**// h. Frekuensi bilangan tertentu**

```
fmt.Print("Masukan Nilai yang ingin dicari frekuensinya: ")
fmt.Scan(&cari)
```

```

    freq := 0
    for _, v := range data {
        if v == cari {
            freq++
        }
    }
    fmt.Printf("h. Frekuensi %d: %d\n", cari, freq)
}

```

### Screenshots Output

```

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\
2\103112400064_Unguided2.go"
• Jumlah elemen: 5
Masukkan elemen:
Data ke-0: 1
Data ke-1: 2
Data ke-2: 3
Data ke-3: 4
Data ke-4: 1
a. Isi array: 1 2 3 4 1
b. Indeks ganjil: 2 4
c. Indeks genap: 1 3 1
Masukkan Indeks kelipatan x: 2
d. Indeks kelipatan 2: 1 3 1
Masukkan Indeks yang ingin dihapus: 2
e. Array Setelah dihapus: 1 2 4 1
f. Rata-rata: 2.00
g. Simpangan baku: 1.22
Masukkan Nilai yang ingin dicari frekuensinya: 1
h. Frekuensi 1: 2

```

// Foto hasil dari menjalankan code

Deskripsi: Kode Go di atas merupakan **program manipulasi array (slice)** yang memiliki berbagai fitur yaitu:

- a. Menampilkan Seluruh Elemen:** Menunjukkan isi array secara langsung dalam satu baris.
- b. Menampilkan Nilai pada Indeks Ganjil:** Mengakses dan mencetak elemen-elemen yang berada pada indeks ganjil, seperti indeks 1, 3, 5, dst.
- c. Menampilkan Nilai pada Indeks Genap:** Sama seperti sebelumnya, tetapi untuk indeks genap seperti 0, 2, 4, dst.
- d. Menampilkan Nilai pada Indeks Kelipatan X:** Pengguna memasukkan angka  $x$ , lalu program menampilkan elemen-elemen array yang berada pada indeks kelipatan  $x$  (misalnya, 0, 3, 6 jika  $x = 3$ ).
- e. Menghapus Elemen pada Indeks Tertentu:** Menghapus elemen dari array berdasarkan indeks yang dimasukkan user menggunakan fungsi `slice append`.
- f. Menghitung Rata-Rata:** Menjumlahkan seluruh elemen lalu membaginya dengan jumlah elemen untuk mendapatkan rata-rata.
- g. Menghitung Simpangan Baku:** Mengukur seberapa jauh data tersebar dari rata-ratanya menggunakan rumus statistik standar deviasi.
- h. Mencari Frekuensi Bilangan Tertentu:** Menghitung berapa kali angka tertentu muncul dalam array.

### Contoh 3

```
// MUHAMMAD FAUZAN
//103112400064
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)
    pertandingan := 1

    for {
        fmt.Printf("Pertandingan %d : ", pertandingan)
        fmt.Scan(&skorA, &skorB)
        if skorA < 0 || skorB < 0 {
            break
        }
        if skorA > skorB {
            hasil = append(hasil, klubA)
        } else if skorB > skorA {
            hasil = append(hasil, klubB)
        } else {
            hasil = append(hasil, "Draw")
        }
        pertandingan++
    }

    for i, v := range hasil {
        if v == "Draw" {
            fmt.Printf("Hasil %d : Draw\n", i+1)
        } else {
            fmt.Printf("Hasil %d : %s\n", i+1, v)
        }
    }
    fmt.Println("pertandingan selesai.")
}
```



## Screenshots Output

```
Klub A: MU
Klub B: Inter
Pertandingan 1 : 2 0
Pertandingan 2 : 1 2
Pertandingan 3 : 2 2
Pertandingan 4 : 0 1
Pertandingan 5 : 3 2
Pertandingan 6 : 1 0
Pertandingan 7 : 5 2
Pertandingan 8 : 2 3
Pertandingan 9 : -1 2
Hasil 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.
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)>
```

// Foto hasil dari menjalankan code

Deskripsi: Program di atas bertujuan untuk mencatat dan menampilkan hasil dari serangkaian pertandingan antara dua klub sepak bola. Setiap kali skor dimasukkan, program membandingkan nilai skor klub A dan klub B.

#### Contoh 4

```
// MUHAMMAD FAUZAN
//103112400064
package main

import (
    "fmt"
)

const NMAX int = 127
type tabel [NMAX]rune

func isiArray(t *tabel, n *int) {
    var karakter rune
    *n = 0
    fmt.Print("Teks : ")
    for {
        fmt.Scanf("%c", &karakter)

        if karakter == '.' || *n >= NMAX {
            break
        }
        if karakter != ' ' && karakter != '\n' && karakter != '\r' {
            t[*n] = karakter
            *n++
        }
    }
}

func cetakArray(t tabel, n int) {
    for i := 0; i < n; i++ {
        fmt.Printf("%c ", t[i])
    }
    fmt.Println()
}

func balikanArray(t *tabel, n int) {
    for i := 0; i < n/2; i++ {
        temp := t[i]
        t[i] = t[n-1-i]
        t[n-1-i] = temp
    }
}
```

```

func palindrom(t tabel, n int) bool {
    var salin tabel
    for i := 0; i < n; i++ {
        salin[i] = t[i]
    }
    balikanArray(&salin, n)

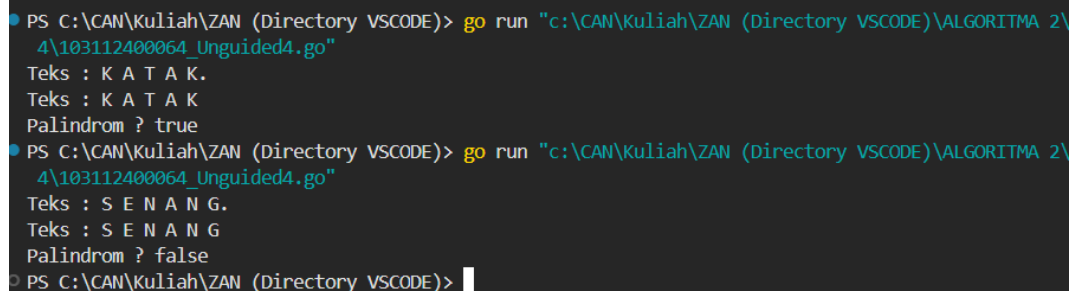
    for i := 0; i < n; i++ {
        if t[i] != salin[i] {
            return false
        }
    }
    return true
}

func main() {
    var teks tabel
    var jumlah int
    isiArray(&teks, &jumlah)
    fmt.Println("Teks : ")
    cetakArray(teks, jumlah)

    if palindrom(teks, jumlah) {
        fmt.Println("Palindrom ? true")
    } else {
        fmt.Println("Palindrom ? false")
    }
}

```

### Screenshots Output



```

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\4\10311240064_Unguided4.go"
Teks : K A T A K.
Teks : K A T A K
Palindrom ? true
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\4\10311240064_Unguided4.go"
Teks : S E N A N G.
Teks : S E N A N G
Palindrom ? false
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)>

```

// Foto hasil dari menjalankan code

Deskripsi: Program Go di atas digunakan untuk mengecek apakah sebuah teks merupakan palindrom dengan memanfaatkan array bertipe rune. Teks dimasukkan oleh pengguna karakter demi karakter hingga karakter titik (.) sebagai penanda akhir input. Spasi, newline (\n), dan carriage return (\r) akan diabaikan, sehingga program hanya menyimpan karakter penting saja ke dalam array.

Setelah input selesai dimasukkan ke array tabel, isi array dicetak satu per satu. Kemudian program akan membuat salinan dari array tersebut dan membalikkan urutannya. Fungsi palindrom membandingkan array asli dengan versi terbalik dari array tersebut. Jika seluruh karakter sama, maka input dianggap palindrom.

#### **IV. KESIMPULAN**

## V. REFERENSI

- The Go Programming Language Specification. (n.d.). *Go.dev*. Retrieved April 24, 2025, from <https://go.dev/ref/spec>
- Effective Go. (n.d.). *Go.dev*. Retrieved April 24, 2025, from [https://go.dev/doc/effective\\_go](https://go.dev/doc/effective_go)
- A Tour of Go – More Types. (n.d.). *Go.dev*. Retrieved April 24, 2025, from <https://go.dev/tour/moretypes/6>
- Go Maps in Action. (n.d.). *Go.dev Blog*. Retrieved April 24, 2025, from <https://go.dev/blog/maps>
- Go Slices: Usage and Internals. (n.d.). *Go.dev Blog*. Retrieved April 24, 2025, from <https://go.dev/blog/slices-intro>