LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 7

STRUCT & ARRAY

Oleh:

MUHAMMAD RUSDIYANTO

2311102053

S1IF-11-02

S1 TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

I. DASAR TEORI

Dalam bahasa pemrograman Golang, type digunakan untuk membuat alias atau mendefinisikan tipe data baru yang berbasis pada tipe data yang sudah ada. Dengan menggunakan type, programmer dapat memberi nama baru pada tipe data yang sudah ada, sehingga kode lebih mudah dibaca dan diorganisasi. Misalnya, kita bisa membuat alias untuk int sebagai UserID dengan type UserID int, sehingga UserID dapat diperlakukan seperti int. Alias ini bermanfaat dalam menyederhanakan kode dan menghindari kesalahan saat ada variabel yang harus mengikuti tipe tertentu dalam konteks yang spesifik, seperti ID pengguna atau nilai mata uang.

Struct di Golang adalah tipe data komposit yang memungkinkan kita untuk menggabungkan beberapa tipe data berbeda menjadi satu kesatuan yang disebut record. Struct dapat diibaratkan sebagai "template" yang mendefinisikan data yang berbeda dalam satu entitas, dengan setiap data atau field di dalam struct memiliki nama dan tipe data sendiri. Misalnya, kita bisa membuat struct bernama Person yang memiliki field bernama Name (string), Age (int), dan Address (string). Struct berguna untuk merepresentasikan objek atau entitas dalam dunia nyata ke dalam kode, seperti data pelanggan, produk, atau transaksi.

Array di Golang adalah kumpulan elemen yang memiliki tipe data yang sama dan ukuran tetap, yang ditentukan saat deklarasi. Array digunakan untuk menyimpan sekumpulan nilai yang diindeks berdasarkan urutan atau posisi, mulai dari 0 hingga n-1. Namun, karena ukuran array tetap, penggunaan array terbatas pada situasi di mana ukuran koleksi sudah diketahui sebelumnya. Untuk menangani koleksi yang ukuran elemennya dinamis, Golang menyediakan tipe data slice. Slice adalah abstraksi dari array yang tidak memiliki batasan ukuran tetap, sehingga lebih fleksibel dan dapat diubah ukurannya. Slice memudahkan manipulasi koleksi data dengan menambahkan atau menghapus elemen tanpa harus mendeklarasikan array baru.

Map di Golang adalah tipe data yang memungkinkan penyimpanan pasangan nilai key-value, mirip dengan dictionary di bahasa lain. Map sangat berguna ketika kita perlu menyimpan data yang berhubungan dengan nilai unik, seperti data inventory yang diindeks dengan nama produk, atau data siswa yang diindeks berdasarkan ID siswa. Dalam deklarasi map, kita mendefinisikan tipe data key dan tipe data value yang disimpan. Map di

Golang bersifat dinamis, artinya kita dapat menambah atau menghapus entri dalam map sesuai kebutuhan. Selain itu, akses terhadap elemen map cepat karena Golang menggunakan hashing untuk mengelola dan mencari data dalam map, menjadikannya pilihan yang efisien untuk pencarian data yang terstruktur.

II. GUIDED

Guided 1 | Source Code

```
package main

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
}

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},
    }

    // Mengithung 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)
    }
}
```

Guided 1 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run 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. Aldi - 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)
```

Guided 1 | Penjelasan

Program di atas adalah program untuk menghitung rata-rata nilai tiga mata pelajaran (Matematika, Fisika, dan Kimia) untuk setiap mahasiswa dalam daftar, kemudian mengurutkan dan menampilkan peringkat mereka berdasarkan nilai rata-rata secara menurun (descending). Misalnya, dengan data mahasiswa yang tersedia, program ini akan mencetak daftar mahasiswa mulai dari yang memiliki rata-rata tertinggi hingga terendah.

Proses dimulai dengan mendeklarasikan tipe data Mahasiswa, yang memiliki atribut Nama, Matematika, Fisika, Kimia, dan RataRata. Fungsi hitungRataRata bertugas menghitung rata-rata nilai untuk setiap mahasiswa dengan menambahkan nilai Matematika, Fisika, dan Kimia, lalu membaginya dengan tiga. Hasil perhitungan ini disimpan dalam atribut RataRata milik objek Mahasiswa.

Dalam fungsi main, program menginisialisasi data mahasiswa dengan nilai awal untuk setiap mata pelajaran dan nol pada nilai rata-rata. Kemudian, untuk setiap mahasiswa dalam array mahasiswa, program memanggil fungsi hitungRataRata untuk mengisi nilai rata-rata. Setelah itu, program mengurutkan array mahasiswa menggunakan fungsi sort.Slice berdasarkan nilai rata-rata secara menurun (dari terbesar ke terkecil). Terakhir, program menampilkan daftar mahasiswa yang sudah diurutkan, dengan format yang mencantumkan nama mahasiswa, nilai rata-rata, dan nilai tiap mata pelajaran.

Guided 2 | Source Code

```
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)
    }
}
```

Guided 2 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run 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
------------------------
20231002        Budi
20231004        Dedi
20231001        Andi
```

Guided 2 | Penjelasan

Program di atas adalah program untuk mengelola data mahasiswa menggunakan struktur data map di Go, dengan NIM sebagai kunci dan nama mahasiswa sebagai nilai. Program ini dapat menambahkan, menampilkan, mengakses, dan menghapus data mahasiswa dari map tersebut.

Pada bagian awal, program mendeklarasikan map mahasiswa dengan beberapa data awal, di mana NIM (misalnya, "20231001") menjadi kunci, dan nama mahasiswa (misalnya, "Andi") menjadi nilai. Program menambahkan data baru untuk mahasiswa dengan NIM "20231004" dan nama "Dedi". Setelah itu, program mencetak seluruh isi map dalam bentuk tabel dengan kolom "NIM" dan "Nama."

Selanjutnya, program menampilkan nama mahasiswa berdasarkan NIM tertentu, yaitu "20231002". Kemudian, data mahasiswa dengan NIM "20231003" dihapus dari map menggunakan fungsi delete. Terakhir, program mencetak kembali daftar mahasiswa setelah penghapusan, dalam format tabel yang sama.

III. UNGUIDED

Unguided 1 | Source Code

```
package main

import (
    "fmt"
    "math"
)

type titik struct {
    x int
    y int
}

type lingkaran struct {
    pusat  titik
    radius int
}

func jarak(p, q titik) float64 {
    return math.Pow(float64(p.x-q.x), 2) + math.Pow(float64(p.y-q.y), 2)
}

func didalam(c lingkaran, p titik) bool {
    var jarak_titik = math.Sqrt(jarak(c.pusat, p))
    if jarak_titik < float64(c.radius) {
        return true
    }
    return false
}

func main() {
    var lingkaran1, lingkaran2 lingkaran
    var titikSembarang titik
    var hasil1, hasil2 bool
    fmt.Print("Titik pusat (x,y) dan jejari lingkaran 1 : ")
    fmt.Scanln(&lingkaran1.pusat.x, &lingkaran1.pusat.y,
&lingkaran1.radius)
    fmt.Print("Titik pusat (x,y) dan jejari lingkaran 2 : ")
    fmt.Scanln(&lingkaran2.pusat.x, &lingkaran2.pusat.y,
&lingkaran2.radius)
    fmt.Print("Titik sembarang (x,y) : ")
    fmt.Scanln(&titikSembarang.x, titikSembarang.y)
    hasil1 = didalam(lingkaran1, titikSembarang)
    hasil2 = didalam(lingkaran2, titikSembarang)
```

```
    if hasil1 && hasil2 {
        fmt.Println("Titik di dalam lingkaran 1 dan 2")
    } else if hasil1 {
        fmt.Println("Titik di dalam lingkaran 1")
    } else if hasil2 {
        fmt.Println("Titik di dalam lingkaran 2")
    } else {
        fmt.Println("Titik di luar lingkaran 1 dan 2")
    }
}
```

Unguided 1 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided1.go
Titik pusat (x,y) dan jejari lingkaran 1 : 1 1 5
Titik pusat (x,y) dan jejari lingkaran 2 : 8 8 4
Titik sembarang (x,y) : 2 2
Titik di dalam lingkaran 1
```

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided1.go
Titik pusat (x,y) dan jejari lingkaran 1 : 1 2 3
Titik pusat (x,y) dan jejari lingkaran 2 : 4 5 6
Titik sembarang (x,y) : 7 8
Titik di dalam lingkaran 2
```

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided1.go
Titik pusat (x,y) dan jejari lingkaran 1 : 5 10 15
Titik pusat (x,y) dan jejari lingkaran 2 : -15 4 20
Titik sembarang (x,y) : 0 0
Titik di dalam lingkaran 1 dan 2
```

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided1.go
Titik pusat (x,y) dan jejari lingkaran 1 : 1 1 5
Titik pusat (x,y) dan jejari lingkaran 2 : 8 8 4
Titik sembarang (x,y) : 15 20
Titik di luar lingkaran 1 dan 2
```

Unguided 1 | Penjelasan

Program di atas adalah modifikasi dari program unguided pada modul 3. Perbedaannya dalam program ini terletak dalam penyimpanan data lingkaran dan pusat lingkaran. Dalam program ini, informasi terkait lingkaran disimpan dalam struct `lingkaran`, sementara informasi terkait titik disimpan dalam struct `titik`.

Proses dimulai dengan mendeklarasikan dua tipe data: titik untuk menyimpan koordinat (x, y) dan lingkaran untuk menyimpan data pusat lingkaran serta jari-jari. Fungsi jarak menghitung jarak kuadrat antara dua titik (p, q) menggunakan rumus jarak Euclidean, dan fungsi didalam mengecek apakah titik sembarang p berada di dalam lingkaran c dengan

menghitung jarak antara pusat lingkaran dan titik sembarang, lalu membandingkannya dengan jari-jari lingkaran.

Dalam fungsi main, program mengambil input dari pengguna untuk dua lingkaran (koordinat pusat dan jari-jari) dan titik sembarang. Dengan menggunakan fungsi didalam, program mengecek apakah titik sembarang berada di dalam lingkaran pertama, kedua, atau keduanya, dan kemudian mencetak hasilnya. Jika titik berada di dalam kedua lingkaran, program menampilkan pesan bahwa titik tersebut berada di dalam lingkaran 1 dan 2, atau menampilkan pesan sesuai dengan lingkaran tempat titik berada.

Unguided 2 | Source Code

```
package main

import (
    "fmt"
    "math"
)

func genap(arr *[]int) {
    fmt.Print("[")
    for i := 0; i < len(*arr); i++ {
        if i%2 == 0 {
            fmt.Printf("%v, ", (*arr)[i])
        }
    }
    fmt.Print("]\n")
}

func ganjil(arr *[]int) {
    fmt.Print("[")
    for i := 0; i < len(*arr); i++ {
        if i%2 != 0 {
            fmt.Printf("%v, ", (*arr)[i])
        }
    }
    fmt.Print("]\n")
}

func kelipatan(arr *[]int) {
    var x int
    fmt.Print("Masukkan nilai x: ")
    fmt.Scanln(&x)
```

```
    fmt.Print("[")
    for i := 0; i < len(*arr); i++ {
        if (*arr)[i]%x == 0 {
            fmt.Printf("%v, ", (*arr)[i])
        }
    }
    fmt.Print("]\n")
}

func hapus(arr *[]int) {
    var indeks int
    fmt.Print("Masukkan indeks: ")
    fmt.Scanln(&indeks)
    *arr = append((*arr)[:indeks], (*arr)[indeks+1:]...)
    fmt.Printf("%v\n", arr)
}

func rata2(arr *[]int) {
    var total int
    for i := 0; i < len(*arr); i++ {
        total += (*arr)[i]
    }
    fmt.Printf("Rata - rata: %v\n", total/len(*arr))
}

func simpangan_baku(arr *[]int) {
    var total, rerata, simpangan_baku float64
    for i := 0; i < len(*arr); i++ {
        total += float64((*arr)[i])
    }
    rerata = total / float64(len(*arr))
    for i := 0; i < len(*arr); i++ {
        simpangan_baku += math.Pow(float64((*arr)[i])-rerata, 2)
    }
    simpangan_baku = math.Sqrt(simpangan_baku / float64(len((*arr))))
    fmt.Printf("Simpangan baku dari array adalah %v\n",
simpangan_baku)
}

func frekuensi(arr *[]int) {
    var x, total int
    fmt.Print("Masukkan nilai x: ")
    fmt.Scanln(&x)
    for i := 0; i < len(*arr); i++ {
        if (*arr)[i] == x {
            total++
```

```
        }
    }
    fmt.Printf("Frekuensi %v dalam array adalah %v\n", x, total)
}

func main() {
    var arrSize, menu int
    fmt.Print("Masukkan ukuran array: ")
    fmt.Scanln(&arrSize)
    arr := make([]int, arrSize)
    for i := 0; i < len(arr); i++ {
        fmt.Printf("Masukkan nilai elemen ke - %v: ", i+1)
        fmt.Scanln(&arr[i])
    }
    for true {
        fmt.Printf("\n[PROGRAM MENU]\n")
        fmt.Printf("1. Tampilkan semua elemen\n")
        fmt.Printf("2. Tampilkan elemen indeks ganjil\n")
        fmt.Printf("3. Tampilkan elemen indeks genap\n")
        fmt.Printf("4. Tampilkan elemen kelipatan x\n")
        fmt.Printf("5. Hapus elemen\n")
        fmt.Printf("6. Tampilkan rata - rata semua elemen\n")
        fmt.Printf("7. Tampilkan simpangan baku\n")
        fmt.Printf("8. Tampilkan frekuensi bilangan x\n")
        fmt.Print("Pilih menu: ")
        fmt.Scanln(&menu)
        switch menu {
        case 1:
            fmt.Printf("%v\n", arr)
            break
        case 2:
            ganjil(&arr)
            break
        case 3:
            genap(&arr)
            break
        case 4:
            kelipatan(&arr)
            break
        case 5:
            hapus(&arr)
            break
        case 6:
            rata2(&arr)
            break
        case 7:
```

```
            simpangan_baku(&arr)
            break
        case 8:
            frekuensi(&arr)
            break
        default:
            fmt.Println("Masukkan nomor yang sesuai dengan menu!")
            break
        }
    }
}
```

Unguided 2 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided2.go
Masukkan ukuran array: 5
Masukkan nilai elemen ke - 1: 10
Masukkan nilai elemen ke - 2: 23
Masukkan nilai elemen ke - 3: 12
Masukkan nilai elemen ke - 4: 16
Masukkan nilai elemen ke - 5: 8
```

```
[PROGRAM MENU]
1. Tampilkan semua elemen
2. Tampilkan elemen indeks ganjil
3. Tampilkan elemen indeks genap
4. Tampilkan elemen kelipatan x
5. Hapus elemen
6. Tampilkan rata - rata semua elemen
7. Tampilkan simpangan baku
8. Tampilkan frekuensi bilangan x
```

```
Pilih menu: 1
[10 23 12 16 8]
```

```
Pilih menu: 2
[23, 16, ]
```

```
Pilih menu: 3
[10, 12, 8, ]
```

```
Pilih menu: 4
Masukkan nilai x: 2
[10, 12, 16, 8, ]
```

```
Pilih menu: 5
Masukkan indeks: 1
&[10 12 16 8]
```

```
Pilih menu: 6
Rata - rata: 11
```

```
Pilih menu: 7
Simpangan baku dari array adalah 2.958039891549808
```

```
Pilih menu: 8
Masukkan nilai x: 10
Frekuensi 10 dalam array adalah 1
```

Unguided 2 | Penjelasan

Program di atas adalah program berbasis menu untuk melakukan operasi pada array bilangan bulat yang dimasukkan oleh pengguna. Program ini memungkinkan pengguna untuk menampilkan elemen-elemen array berdasarkan kriteria tertentu, menghapus elemen, menghitung rata-rata dan simpangan baku, serta menghitung frekuensi kemunculan angka tertentu dalam array.

Di awal, pengguna diminta untuk menentukan ukuran array dan memasukkan nilai elemen-elemen array satu per satu. Setelah itu, menu utama ditampilkan, dengan berbagai pilihan operasi:

1. Tampilkan Semua Elemen: Menampilkan seluruh elemen array.
2. Tampilkan Elemen Indeks Ganjil: Menampilkan elemen yang berada di indeks ganjil.
3. Tampilkan Elemen Indeks Genap: Menampilkan elemen yang berada di indeks genap.
4. Tampilkan Elemen Kelipatan x: Meminta pengguna memasukkan nilai x dan menampilkan elemen yang merupakan kelipatan dari x.
5. Hapus Elemen: Meminta pengguna memilih indeks elemen yang akan dihapus dari array.
6. Tampilkan Rata-rata Semua Elemen: Menghitung rata-rata nilai elemen dalam array.
7. Tampilkan Simpangan Baku: Menghitung simpangan baku elemen array, sebagai ukuran penyebaran nilai.
8. Tampilkan Frekuensi Bilangan x: Meminta pengguna memasukkan nilai x dan menghitung berapa kali nilai tersebut muncul dalam array.

Setiap fungsi melakukan operasi khusus sesuai dengan pilihannya di menu. Program terus berjalan dalam *loop* hingga pengguna memutuskan untuk berhenti (tidak ada kondisi untuk keluar dari *loop* ini dalam kode).

Unguided 3 | Source Code

```
package main

import "fmt"

type pertandingan struct {
    klub1 string
    klub2 string
    hasil []string
}

func main() {
    var pertandingan pertandingan
```

```
    var skor_k1, skor_k2 int
    fmt.Print("Klub A: ")
    fmt.Scanln(&pertandingan.klub1)
    fmt.Print("Klub B: ")
    fmt.Scanln(&pertandingan.klub2)
    for i := 1; i > 0; i++ {
        fmt.Printf("Pertandingan %v: ", i)
        fmt.Scanln(&skor_k1, &skor_k2)
        if skor_k1 < 0 || skor_k2 < 0 {
            break
        } else if skor_k1 > skor_k2 {
            pertandingan.hasil = append(pertandingan.hasil,
pertandingan.klub1)
        } else if skor_k1 < skor_k2 {
            pertandingan.hasil = append(pertandingan.hasil,
pertandingan.klub2)
        } else {
            pertandingan.hasil = append(pertandingan.hasil, "Draw")
        }
    }
    for i := 0; i < len(pertandingan.hasil); i++ {
        fmt.Printf("Hasil %v: %v\n", i+1, pertandingan.hasil[i])
    }
    fmt.Println("Pertandingan selesai")
}
```

Unguided 3 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided3.go
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
```

Unguided 3 | Penjelasan

Program di atas adalah program untuk mencatat hasil pertandingan antara dua klub sepak bola hingga pengguna memasukkan skor negatif, yang akan mengakhiri input. Setiap hasil pertandingan dicatat sebagai menang untuk salah satu klub atau "Draw" jika skornya sama.

Program dimulai dengan mendeklarasikan tipe data pertandingan, yang memiliki atribut klub1, klub2, dan hasil. Di dalam main, program meminta pengguna memasukkan nama kedua klub (klub1 dan klub2). Program kemudian memasuki loop untuk mencatat hasil setiap pertandingan. Pada setiap iterasi, skor untuk masing-masing klub diminta dari pengguna (skor_k1 untuk klub1 dan skor_k2 untuk klub2). Jika salah satu skor negatif, loop berhenti, mengakhiri input pertandingan.

Setiap skor yang valid dibandingkan:

- Jika skor_k1 lebih besar dari skor_k2, maka klub1 dicatat sebagai pemenang.
- Jika skor_k2 lebih besar dari skor_k1, maka klub2 dicatat sebagai pemenang.
- Jika skornya sama, hasil pertandingan dicatat sebagai "Draw."

Setelah selesai, program mencetak hasil setiap pertandingan secara berurutan. Ini menampilkan apakah pertandingan dimenangkan oleh salah satu klub atau berakhir imbang, kemudian menutup dengan pesan "Pertandingan selesai."

Unguided 4 | Source Code

```
package main

import "fmt"

const NMAX int = 127

type tabel [NMAX]rune

func isiArray(t *tabel, n *int) {
   fmt.Print("Jumlah karakter [k < 127]: ")
   fmt.Scanln(n)
   if *n > NMAX {
      fmt.Println("Jumlah karakter terlalu banyak!")
      isiArray(t, n)
   } else {
      fmt.Print("Teks: ")
      for i := 0; i < *n; i++ {
         fmt.Scanf("%c", &t[i])
      }
   }
}

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) {
   var temp tabel
   for i := 0; i < n; i++ {
      temp[i] = (*t)[n-1-i]
   }
   *t = temp
}

func main() {
   var tab tabel
```

```
    var m int
    isiArray(&tab, &m)
    balikanArray(&tab, m)
    fmt.Print("Reverse teks: ")
    cetakArray(tab, m)
}
```

Unguided 4 | Output

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided4.go
Jumlah karakter [k < 127]: 6
Teks: SENANG
Reverse teks: G N A N E S
```

Unguided 4 | Source Code - modded

```
package main

import "fmt"

const NMAX int = 127

type tabel [NMAX]rune

func isiArray(t *tabel, n *int) {
    fmt.Print("Jumlah karakter [k < 127]: ")
    fmt.Scanln(n)
    if *n > NMAX {
        fmt.Println("Jumlah karakter terlalu banyak!")
    } else {
        fmt.Print("Teks: ")
        for i := 0; i < *n; i++ {
            fmt.Scanf("%c", &t[i])
        }
    }
}

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

func palindrom(t tabel, n int) bool {
    for i := 0; i < n/2; i++ {
        if t[i] != t[n-1-i] {
```

```
            return false
        }
    }
    return true
}

func main() {
    var tab tabel
    var m int
    isiArray(&tab, &m)
    balikanArray(&tab, m)

    if palindrom(tab, m) {
        fmt.Println("Palindrom: true")
    } else {
        fmt.Println("Palindrom: false")
    }
}
```

Unguided 4 | Output - modded

```
PS C:\MyFiles\Visual Studio Projects HERE\alpro2\day5> go run unguided4-mod.go
Jumlah karakter [k < 127]: 5
Teks: KATAK
Palindrom: true
```

Unguided 4 | Penjelasan

Program di atas adalah program untuk membalik urutan karakter dalam sebuah array dan memeriksa apakah susunan tersebut merupakan palindrom. Pengguna memasukkan sejumlah karakter (maksimal 127) yang kemudian akan dibalik urutannya dan diperiksa apakah urutan tersebut sama dengan urutan awal (palindrom).

Proses dimulai dengan mengambil input jumlah karakter dan karakter teks yang diinputkan pengguna. Fungsi isiArray digunakan untuk menyimpan karakter yang dimasukkan ke dalam array tabel (t). Jika jumlah karakter melebihi batas maksimal (NMAX), program akan menampilkan pesan bahwa jumlah karakter terlalu banyak.

Fungsi balikanArray digunakan untuk membalik urutan karakter dalam array tabel. Fungsi ini menyimpan array dalam urutan terbalik di variabel sementara temp, lalu menyalin hasilnya kembali ke array asli tabel.

Setelah array dibalik, fungsi palindrom memeriksa apakah urutan karakter sama dari awal hingga akhir. Jika karakter dari awal hingga tengah sama dengan karakter dari tengah hingga akhir, fungsi akan mengembalikan

nilai true, menunjukkan bahwa array tersebut adalah palindrom. Program kemudian mencetak hasil Palindrom: true atau Palindrom: false tergantung pada hasil pemeriksaan tersebut.