

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
NILAI EKSTRIM MIN/MAX
MODUL XI**



Disusun Oleh :

Rakha Arbiyandanu / 2311102263

IF-11-6

Dosen Pengampu :

ABEDNEGO DWI SEPTIADI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dalam dunia pemrograman, seringkali kita dihadapkan pada kebutuhan untuk mencari nilai maksimum dan minimum dari sekumpulan data. Di Golang, ada beberapa cara untuk mencapai tujuan ini, mulai dari memanfaatkan fungsi bawaan yang simpel hingga merancang algoritma sendiri. Masing-masing pendekatan memiliki kelebihan dan kekurangan, tergantung pada kebutuhan dan kompleksitas data yang diolah.

Salah satu cara termudah adalah dengan menggunakan fungsi `math.Max` dan `math.Min` dari package `math`. Fungsi ini sangat praktis untuk membandingkan dua bilangan dan langsung mendapatkan nilai maksimum atau minimum di antara keduanya. Namun, jika kita berurusan dengan array atau slice yang berisi banyak data, kita perlu menggunakan pendekatan lain, seperti iterasi dengan loop atau memanfaatkan package `sort`.

Iterasi dengan loop memungkinkan kita untuk menelusuri setiap elemen dalam array atau slice dan membandingkannya satu per satu. Dengan menyimpan nilai maksimum dan minimum sementara, kita dapat memperbarui nilai-nilai tersebut setiap kali menemukan elemen yang lebih besar atau lebih kecil. Metode ini cukup efisien untuk data berukuran kecil hingga sedang.

Alternatif lainnya adalah dengan menggunakan package `sort`. Dengan mengurutkan data terlebih dahulu, kita bisa langsung mendapatkan nilai maksimum dan minimum dari elemen pertama dan terakhir array. Namun, perlu diingat bahwa proses pengurutan data memiliki kompleksitas waktu yang lebih tinggi dibandingkan iterasi sederhana, sehingga mungkin kurang efisien untuk data berukuran sangat besar.

Pemilihan metode yang tepat bergantung pada kebutuhan dan karakteristik data. Untuk data sederhana, fungsi bawaan `math.Max` dan `math.Min` sudah cukup. Untuk data yang lebih kompleks, iterasi dengan loop atau penggunaan package `sort` bisa menjadi pilihan yang lebih baik. Yang terpenting adalah memahami prinsip dasar dari setiap metode dan memilih yang paling efisien untuk kasus yang dihadapi.

I. GUIDED

Soal Studi Case

Pencarian Nilai Ekstrim pada array bertipe dasar

Sourcecode

```
package main

import "fmt"

// Mendeklarasikan tipe data array arrInt dengan panjang
2023
type arrInt [2023]int

// Fungsi untuk mencari indeks elemen terkecil dalam array
func terkecil(tabInt arrInt, n int) int {
    var idx int = 0 // idx menyimpan indeks elemen
    terkecil
    var j int = 1
    for j < n {
        if tabInt[idx] > tabInt[j] {
            idx = j // Simpan indeks j jika elemen di
            indeks j lebih kecil
        }
        j = j + 1
    }
    return idx
}

// Fungsi main untuk menguji fungsi terkecil
func main() {
    var n int
    var tab arrInt

    // Meminta input jumlah elemen array
    fmt.Print("Masukkan jumlah elemen (maks 2023): ")
    fmt.Scan(&n)

    // Validasi input jumlah elemen
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah elemen harus antara 1 dan
2023.")
        return
    }

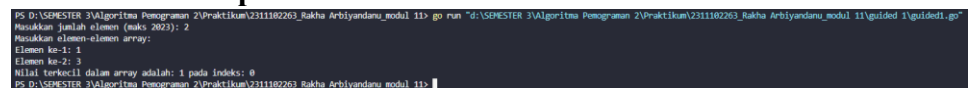
    // Memasukkan elemen-elemen array
    fmt.Println("Masukkan elemen-elemen array:")
    for i := 0; i < n; i++ {
        fmt.Print("Elemen ke-", i+1, ": ")
        fmt.Scan(&tab[i])
    }

    // Memanggil fungsi terkecil untuk menemukan indeks
    elemen terkecil
```

```
idxMin := terkecil(tab, n)

// Menampilkan nilai dan indeks terkecil
fmt.Println("Nilai terkecil dalam array adalah:",
tab[idxMin], "pada indeks:", idxMin)
}
```

Screenshoot Output



```
PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_modul 11> go run "d:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_modul 11\guided 1\guided1.go"
Masukkan jumlah elemen (maks: 2023): 2
Masukkan elemen-elemen array:
Elemen ke-1: 1
Elemen ke-2: 3
Nilai terkecil dalam array adalah: 1 pada indeks: 0
PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_modul 11>
```

Deskripsi Program

Program ini dibuat untuk mencari angka terkecil dari sekumpulan angka, maksimal 2023 angka. Pertama, kamu diminta menentukan berapa banyak angka yang ingin dimasukkan. Setelah itu, masukkan angka-angka tersebut satu per satu. Program akan memproses angka-angka ini dan mencari tahu angka terkecilnya. Nggak cuma itu, program juga akan menunjukkan di posisi ke berapa angka terkecil itu berada. Sayangnya, program ini punya batasan, yaitu hanya bisa memproses maksimal 2023 angka saja.

II. GUIDED

Soal Studi Case

Menghitung hasil penjumlahan 1 hingga n Base-case $n == 1$.

Sourcecode

```
package main

import "fmt"

// Definisi struct mahasiswa dengan atribut nama, nim,
kelas, jurusan, dan ipk
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk                        float64
}

// Definisi tipe data array mahasiswa dengan kapasitas
maksimal 2023
type arrMhs [2023]mahasiswa

// Fungsi untuk mencari IPK tertinggi dalam array
mahasiswa
func ipk(T arrMhs, n int) float64 {
    var tertinggi float64 = T[0].ipk
    var j int = 1
    for j < n {
        if tertinggi < T[j].ipk {
            tertinggi = T[j].ipk
        }
        j = j + 1
    }
    return tertinggi
}

// Fungsi main untuk mengisi data mahasiswa dan mencari
IPK tertinggi
func main() {
    var n int
    var dataMhs arrMhs

    // Meminta input jumlah mahasiswa
    fmt.Print("Masukkan jumlah mahasiswa (maks 2023): ")
    fmt.Scan(&n)

    // Validasi jumlah mahasiswa yang dimasukkan
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah mahasiswa harus antara 1 dan
2023.")
        return
    }

    // Mengisi data mahasiswa
    for i := 0; i < n; i++ {
        fmt.Printf("\nMasukkan data mahasiswa ke-%d\n",
i+1)
```

```

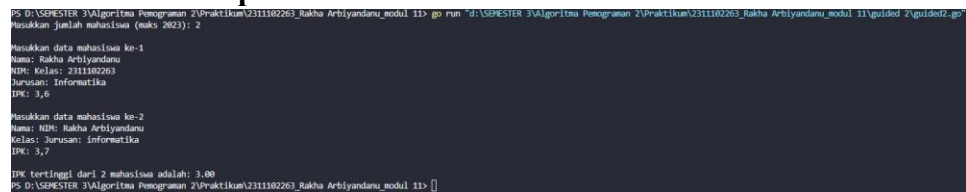
        fmt.Print("Nama: ")
        fmt.Scan(&dataMhs[i].nama)
        fmt.Print("NIM: ")
        fmt.Scan(&dataMhs[i].nim)
        fmt.Print("Kelas: ")
        fmt.Scan(&dataMhs[i].kelas)
        fmt.Print("Jurusan: ")
        fmt.Scan(&dataMhs[i].jurusan)
        fmt.Print("IPK: ")
        fmt.Scan(&dataMhs[i].ipk)
    }

    // Mencari dan menampilkan IPK tertinggi
    tertinggi := ipk(dataMhs, n)
    fmt.Printf("\nIPK tertinggi dari %d mahasiswa adalah:
%.2f\n", n, tertinggi)

}

```

Screenshot Output



```

PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_nodul 11> go run -d:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_nodul 11\guided 2\guided2.go
Masukkan jumlah mahasiswa (maks 2023): 2
Masukkan data mahasiswa ke-1
Nama: Rakha Arbiyandaru
NIM: 2311180263
Jurusan: Informatika
IPK: 3,6
Masukkan data mahasiswa ke-2
Nama: HDM: Rakha Arbiyandaru
Kelas: Jurusan: Informatika
IPK: 3,7
IPK tertinggi dari 2 mahasiswa adalah: 3.00
PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandaru_nodul 11> |

```

Deskripsi Program

Kode ini struct mahasiswa untuk menyimpan data mahasiswa nama, NIM, kelas, jurusan, IPK dan array arrMhs untuk menyimpan data maksimal 2023 mahasiswa. Fungsi ipk mencari IPK tertinggi dalam array tersebut dengan membandingkan setiap IPK mahasiswa. Fungsi main meminta input jumlah mahasiswa, lalu mengisi data masing-masing mahasiswa. Terakhir, program memanggil fungsi ipk untuk mencari dan menampilkan IPK tertinggi dari data yang telah diinputkan.

I. UNGUIDED

Soal Studi Case

Sebuah program digunakan untuk mendata berat anak kelinci yang akan dijual ke pasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat anak kelinci yang akan dijual.

Sourcecode

```
package main
import "fmt"

func main() {
    var N int
    var berat [1000]float64
    var terkecil ,terbesar float64

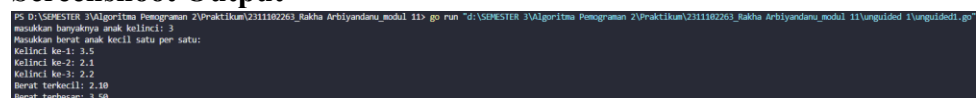
    fmt.Print("masukkan banyaknya anak kelinci: ")
    fmt.Scanln(&N)

    fmt.Println("Masukkan berat anak kecil satu per satu: ")
    for i := 0; i < N; i++ {
        fmt.Printf("Kelinci ke-%d: ", i+1)
        fmt.Scanln(&berat[i])
    }

    terkecil = berat[0]
    terbesar = berat[0]

    for i := 1; i < N; i++ {
        if berat[i] < terkecil {
            terkecil = berat[i]
        }
        if berat[i] > terbesar{
            terbesar = berat[i]
        }
    }
    fmt.Printf("Berat terkecil: %.2f\n", terkecil)
    fmt.Printf("Berat terbesar: %.2f\n", terbesar)
}
```

Screenshoot Output



```
PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311182263_Rakha Arbiyandaru_modul 11> go run "D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311182263_Rakha Arbiyandaru_modul 11\unguided 1\unguided1.go"
masukkan banyaknya anak kelinci: 3
Masukkan berat anak kecil satu per satu:
kelinci ke-1: 3.5
kelinci ke-2: 2.1
kelinci ke-3: 2.2
Berat terkecil: 2.10
Berat terbesar: 3.50
```

Deskripsi Program

Pertama, program akan meminta pengguna untuk memasukkan jumlah anak kelinci yang akan ditimbang. Selanjutnya, pengguna diminta untuk memasukkan berat badan masing-masing anak kelinci secara berurutan. Setelah semua data berat badan terkumpul, program akan memproses data tersebut untuk mencari nilai terkecil dan terbesar. Nilai terkecil dan terbesar ini kemudian akan ditampilkan sebagai output program.

II. UNGUIDED

Soal Studi Case

Sebuah program digunakan untuk menentukan tarif ikan yang akan dijual ke pasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat ikan yang akan dijual.

Sourcecode

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var x, y int
    fmt.Print("Masukkan jumlah ikan yang akan dijual (x) dan\njumlah ikan per wadah (y): ")
    fmt.Scan(&x, &y)

    // Input berat ikan
    fmt.Println("Masukkan berat ikan:")
    berat := make([]float64, x)
    for i := 0; i < x; i++ {
        fmt.Scan(&berat[i])
    }

    wadahHitung := int(math.Ceil(float64(x) / float64(y)))
    wadahTotals := make([]float64, wadahHitung)

    for i := 0; i < x; i++ {
        wadahjumlah := i / y
        wadahTotals[wadahjumlah] += berat[i]
    }
}
```



```

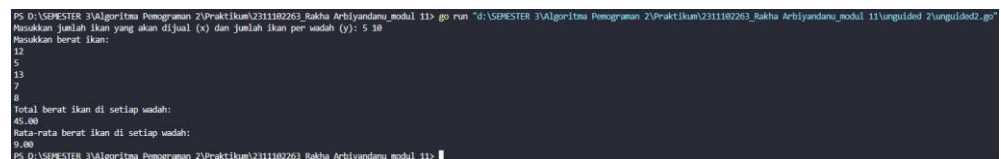
wadahRatarata := make([]float64, wadahHitung)
for i := 0; i < wadahHitung; i++ {
    if (i+1)*y <= x {
        wadahRatarata[i] = wadahTotals[i] / float64(y)
    } else {
        wadahRatarata[i] = wadahTotals[i] / float64(x%y)
    }
}

fmt.Println("Total berat ikan di setiap wadah:")
for _, total := range wadahTotals {
    fmt.Printf("%.2f ", total)
}
fmt.Println()

fmt.Println("Rata-rata berat ikan di setiap wadah:")
for _, avg := range wadahRatarata {
    fmt.Printf("%.2f ", avg)
}
fmt.Println()
}

```

Screenshoot Output



```

PS D:\SEPMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandani_modul 11> go run "D:\SEPMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandani_modul 11\unguided 2\unguided2.go"
Masukkan jumlah ikan yang akan dijual (x) dan jumlah ikan per wadah (y): 5 10
Masukkan berat ikan:
12
5
12
5
7
8
Total berat ikan di setiap wadah:
45.00
Rata-rata berat ikan di setiap wadah:
9.00
PS D:\SEPMESTER 3\Algoritma Pemrograman 2\Praktikum\2311180263_Rakha Arbiyandani_modul 11>

```

Deskripsi Program

Program ini menghitung total sama rata-rata berat ikan di tiap wadah. Jadi, awalnya user disuruh masukan jumlah ikan (x), kapasitas wadah (y), sama berat tiap ikan. Habis itu, program bakal hitung berapa banyak wadah yang dibutuhkan, terus berat ikannya dimasukin ke wadah satu per satu. Total berat tiap wadah langsung dijumlahin, dan kalau wadah terakhir nggak penuh, rata-ratanya dihitung dari ikan yang ada aja. Terakhir, hasilnya bakal ditampilkan: total berat per wadah sama rata-rata beratnya, pake angka dua digit di belakang koma biar rapi.

III. UNGUIDED

Soal Studi Case

Pos Pelayanan Terpadu (posyandu) sebagai tempat pelayanan kesehatan perlu mencatat data berat balita (dalam kg). Petugas akan memasukkan data tersebut ke dalam array. Dari data yang diperoleh akan dicari berat balita terbesar, terkecil, dan rata-ratanya. Buatlah program dengan spesifikasi subprogram sebagai berikut

Sourcecode

```
package main

import (
    "fmt"
)

type arrBalita [100]float64

func hitungMinMax(arr arrBalita, n int, min, max *float64) {
    *min = arr[0]
    *max = arr[0]

    for i := 1; i < n; i++ {
        if arr[i] < *min {
            *min = arr[i]
        }
        if arr[i] > *max {
            *max = arr[i]
        }
    }
}

func rataRata(arr arrBalita, n int) float64 {
    var total float64 = 0
    for i := 0; i < n; i++ {
        total += arr[i]
    }
    return total / float64(n)
}

func main() {
    var n int
    var berat arrBalita
    var min, max float64
```

```

    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)

    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan berat balita ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }

    hitungMinMax(berat, n, &min, &max)

    rata := rataRata(berat, n)

    // Output hasil
    fmt.Printf("Berat balita minimum: %.2f kg\n", min)
    fmt.Printf("Berat balita maksimum: %.2f kg\n", max)
    fmt.Printf("Rata-rata berat balita: %.2f kg\n", rata)
}

```

Screenshoot Output

```

PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311102263_Rakha Arbiyandaru_modul 11> go run "d:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311102263_Rakha Arbiyandaru_modul 11\unguided 3\unguided3.go"
Masukkan banyak data berat balita: 4
Masukkan berat balita ke-1: 5.3
Masukkan berat balita ke-2: 6.2
Masukkan berat balita ke-3: 4.1
Masukkan berat balita ke-4: 9.9
Berat balita minimum: 4.10 kg
Berat balita maksimum: 9.90 kg
Rata-rata berat balita: 6.38 kg
PS D:\SEMESTER 3\Algoritma Pemrograman 2\Praktikum\2311102263_Rakha Arbiyandaru_modul 11>

```

Deskripsi Program

Program ini buat ngolah data berat balita, jadi bisa cari yang paling ringan (minimum), paling berat (maksimum), sama rata-ratanya. Awalnya, user masukin jumlah balita dan beratnya satu per satu. Terus, program pake fungsi `hitungMinMax` buat nyari nilai terkecil dan terbesar dari data itu, sama fungsi `rataRata` buat hitung rata-ratanya. Akhirnya, hasilnya ditampilin ke user dengan format dua angka di belakang koma biar keliatan rapi.