

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMOGRAMAN II**  
**MODUL X**  
**PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Oleh:

NAMA : DWI HESTI ARIANI

NIM : 2311102094

KELAS : 11- IF -02

**S1 TEKNIK INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024**

## I. DASAR TEORI

Nilai ekstrim dalam himpunan data merujuk pada nilai tertinggi (max) dan nilai terendah (min) dalam Kumpulan data tersebut. Proses pencarian nilai ekstrim melibatkan iterasi melalui elemen-elemen dalam himpunan data untuk membandingkan dan menemukan nilai yang memenuhi kriteria maksimum atau minimum.

Himpunan data dapat berupa array atau slice yang menyimpan sekumpulan elemen, baik itu angka, karakter, atau objek lainnya. Golang menyediakan struktur data ini untuk memudahkan pengelolaan dan manipulasi data.

### Algoritma Pencarian Nilai Ekstrim

Algoritma dasar untuk mencari nilai ekstrim dapat dilakukan dengan pendekatan berikut:

- **Inisialisasi:** Tentukan nilai awal untuk maksimum dan minimum. Biasanya, ini dilakukan dengan mengambil elemen pertama dari himpunan data.
- **Iterasi:** Lakukan iterasi melalui setiap elemen dalam himpunan data.
- **Perbandingan:** Bandingkan setiap elemen dengan nilai maksimum dan minimum saat ini. Jika elemen lebih besar dari maksimum saat ini, perbarui nilai maksimum. Jika elemen lebih kecil dari minimum saat ini, perbarui nilai minimum.
- **Output:** Setelah iterasi selesai, hasilkan nilai maksimum dan minimum.

Berikut ini adalah notasi dalam pseudocode dan Bahasa Go untuk pencarian nilai terbesar atau maximum

|   | Notasi Algoritma               | Notasi dalam bahasa Go      |
|---|--------------------------------|-----------------------------|
| 1 | $\text{max} \leftarrow 1$      | $\text{max} = 0$            |
| 2 | $i \leftarrow 2$               | $i = 1$                     |
| 3 | while $i \leq n$ do            | for $i < n$ {               |
| 4 | if $a[i] > a[\text{max}]$ then | if $a[i] > a[\text{max}]$ { |
| 5 | $\text{max} \leftarrow i$      | $\text{max} = i$            |
| 6 | endif                          | }                           |
| 7 | $i \leftarrow i + 1$           | $i = i + 1$                 |
| 8 | endwhile                       | }                           |

## II. GUIDED

### Guided 1

#### Source Code

```
package main

import (
    "fmt"
)

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

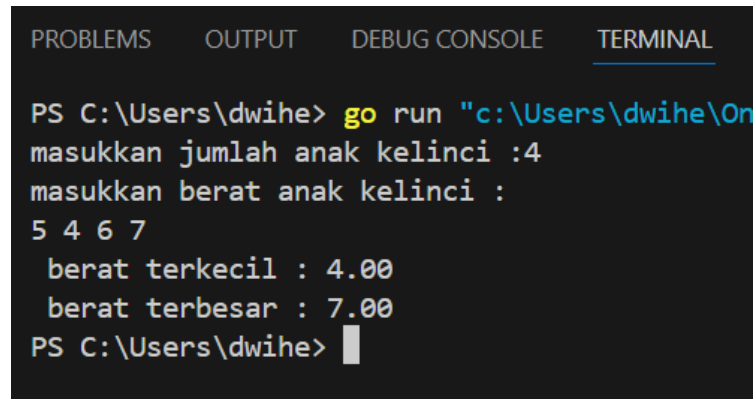
    fmt.Print("masukkan jumlah anak kelinci :")
    fmt.Scan(&N)

    fmt.Println("masukkan berat anak kelinci :")
    for i := 0; i < N; i++ {
        fmt.Scan(&berat[i])
    }

    // inisialisasi nilai min dan max dg elemen pertama
    min := berat[0]
    max := berat[0]

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

## Output Program

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The command 'go run "c:\Users\dwihe\On' is entered. The program prompts for the number of children ('masukkan jumlah anak kelinci :4') and their weights ('masukkan berat anak kelinci :'). The user enters '5 4 6 7'. The program outputs 'berat terkecil : 4.00' and 'berat terbesar : 7.00'.

```
PS C:\Users\dwihe> go run "c:\Users\dwihe\On
masukkan jumlah anak kelinci :4
masukkan berat anak kelinci :
5 4 6 7
berat terkecil : 4.00
berat terbesar : 7.00
PS C:\Users\dwihe>
```

## Deskripsi Program

Program diatas ialah program sederhana dalam bahasa Go yang berfungsi untuk menampilkan nilai berat terkecil dan berat terbesar dari inputan yang diberikan oleh user, variable 'N' merupakan integer yang akan menyimpan jumlah anak kelinci yang di input oleh user, variable 'berat' merupakan arrat yang dapat menyimpan hingga 1000 nilai yang digunakan untuk menyimpan berat masing-masing anak kelinci.

Program akan menginisialisasi variable 'min' dan 'max' yang berisi nilai berat anak kelinci pertama, lalu program akan melakukan iterasi dari elemen kedua sampai elemen terakhir dari array 'berat' , jika ditemukan berat yang lebih kecil dari 'min' maka program akan memperbarui nilai 'min' tersebut, begitu pun dengan nilai 'max'.

## Guided 2

### Source Code

```
package main

import (
    "fmt"
)

func main() {
    var x, y int
    fmt.Print("masukkan jumlah ikan dan kapasitas wadah : ")
    fmt.Scan(&x, &y)
```

```

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

jumlahWadah := (x + y - 1) / y
totalBeratWadah := make([]float64, jumlahWadah)

for i := 0; i < x; i++ {
    indeksWadah := i / y
    totalBeratWadah[indeksWadah] += berat[i]
}
//output total berat tiap wadah
fmt.Println("total berat tiap wadah :")
for _, total := range totalBeratWadah {
    fmt.Printf("%.2f", total)
}
fmt.Println()

// output rata2
fmt.Println("rata -rata berat tiap wadah: ")
for _, total := range totalBeratWadah {
    rataRata := total / float64(y)
    fmt.Printf("%.2f", rataRata)
}
fmt.Println()
}

```

### Output Program

```

PS C:\Users\dwih> go run "c:\Users\dwih\OneDrive\Doc
masukkan jumlah ikan dan kapasitas wadah : 4 2
masukkan berat tiap ikan:
2 3 2.5 1
total berat tiap wadah :
5.003.50
rata -rata berat tiap wadah:
2.501.75
PS C:\Users\dwih> █

```

## Deskripsi Program

Program ini merupakan program sederhana Bahasa Golang yang berfungsi untuk menghitung total berat ikan yang dimasukkan ke dalam masing-masing wadah berdasarkan kapasitas wadah yang diberikan, program ini juga menghitung rata-rata berat ikan per wadah. Program akan meminta user untuk memasukkan jumlah ikan dan kapasitas wadah dalam satu input, lalu program akan menyimpan masukkan dari user ke dalam variable x dan y.

Program membuat array dengan Panjang x untuk menyimpan berat masing-masing ikan dalam variable 'berat', user diminta untuk memasukkan berat dari setiap ikan dan setiap berat yang disimpan dalam variable 'berat'. Kemudian program akan menghitung jumlah wadah yang diperlukan dengan rumus  $(x + y - 1) / y$ , untuk menghitung total berat per wadah program melakukan iterasi pada perulangan for dimana indeks wadah dihitung dengan membagi indeks ikan 'i' dengan kapasitas wadah 'y', kemudian program akan membagi total berat wadah dengan kapasitas wadah 'y' untuk mendapatkan rata-rata berat ikan per wadah.

## III. UNGUIDED

### Source Code

#### Unguided 1

```
package main

import "fmt"

type arrBalita [100]float64

func hitungMinMax(arrBerat arrBalita, n int, bMin, bMax
*float64) {
    /* I.S. Terdefinisi array dinamis arrBerat dengan panjang n
       F.S. Menampilkan berat minimum dan maksimum balita */
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]
    for i := 1; i < n; i++ { // Hanya proses hingga jumlah
input n
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
    }
```

```

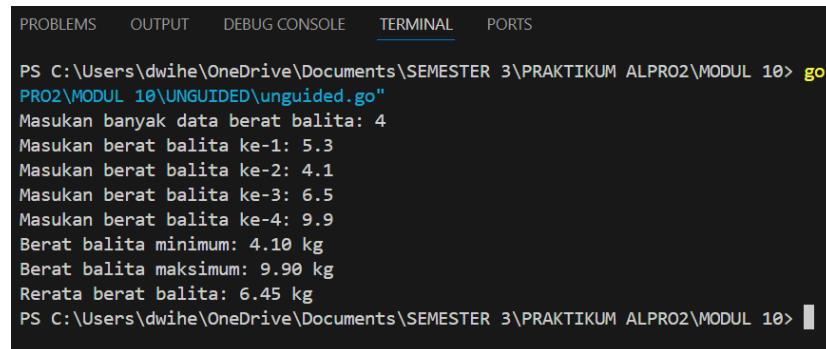
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
    }
    fmt.Printf("Berat balita minimum: %.2f kg\n", *bMin)
    fmt.Printf("Berat balita maksimum: %.2f kg\n", *bMax)
}

func rerata(arrBerat arrBalita, n int) float64 {
    /* Menghitung dan mengembalikan rerata berat balita sesuai
    jumlah input n */
    var total float64
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}

func main() {
    var n int
    var arrBalita arrBalita
    fmt.Print("Masukan banyak data berat balita: ")
    fmt.Scanln(&n)
    for i := 0; i < n; i++ {
        fmt.Printf("Masukan berat balita ke-%d: ", i+1)
        fmt.Scanln(&arrBalita[i])
    }
    var bMin, bMax float64
    hitungMinMax(arrBalita, n, &bMin, &bMax)
    rerataBalita := rerata(arrBalita, n)
    fmt.Printf("Rerata berat balita: %.2f kg\n", rerataBalita)
}

```

## Output Program

A screenshot of a Go IDE's terminal window. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), and PORTS. The command prompt shows the file path 'PS C:\Users\dwih\OneDrive\Documents\SEMESTER 3\PRAKTIKUM ALPRO2\MODUL 10' followed by 'go PRO2\MODUL 10\UNGUIDED\unguided.go'. The program's output is as follows:

```
Masukan banyak data berat balita: 4
Masukan berat balita ke-1: 5.3
Masukan berat balita ke-2: 4.1
Masukan berat balita ke-3: 6.5
Masukan berat balita ke-4: 9.9
Berat balita minimum: 4.10 kg
Berat balita maksimum: 9.90 kg
Rerata berat balita: 6.45 kg
```

The prompt returns to 'PS C:\Users\dwih\OneDrive\Documents\SEMESTER 3\PRAKTIKUM ALPRO2\MODUL 10'.

## Deskripsi Program

Program diatas ialah program sederhana dalam bahasa Go yang berfungsi untuk mengelola data berat balita dengan menghitung berat minimum, maksimum, dan rata – rata. Program menggunakan type array pada variable ‘arrBalita’ dengan Panjang maksimal 100 untuk menyimpan nilai berat balita. Func hitungMinMax digunakan untuk menghitung berat minimum dan maksimum dari array ‘arrBerat’.

Kemudian program akan menginisialisasi ‘bMin’ dan ‘bMax’ pada berat balita pertama dan akan melakukan iterasi pada perulangan for untuk mencari nilai minimum dan maksimum dari array berdasarkan jumlah nilai yang dimasukkan oleh user Setelah menemukan nilai minimum dan maksimum fungsi akan mencetak hasilnya, lalu program akan menghitung rata-rata dari berat balita dengan menjumlahkan total berat balita keseluruhan dan menyimpan nya dalam variable ‘total’ dan membagi dengan jumlah elemen dalam array .