

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 7**

**MATERI**



Oleh:

ABISAR FATHIR

103112400068

12-IF-01

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO**

**2025**

## I. DASAR TEORI

Pencarian nilai ekstrem adalah proses untuk menemukan nilai terkecil (minimum) dan nilai terbesar (maksimum) dari sekumpulan data. Teknik ini umum digunakan dalam pengolahan data karena memberikan informasi penting mengenai rentang dan karakteristik data.

Dalam praktiknya, algoritma pencarian nilai ekstrem bekerja dengan cara membandingkan setiap elemen dalam array secara berurutan. Pertama, data pertama diasumsikan sebagai nilai minimum atau maksimum. Kemudian, setiap data berikutnya dibandingkan dengan nilai sementara tersebut, dan jika ditemukan data yang lebih kecil atau lebih besar, maka nilai minimum atau maksimum diperbarui. Setelah semua data diperiksa, nilai ekstrem yang didapat dianggap valid.

Selain mencari nilai maksimum dan minimum, sering kali diperlukan juga untuk menghitung nilai rata-rata. Nilai rata-rata diperoleh dengan menjumlahkan semua elemen dalam array, kemudian membaginya dengan jumlah elemen yang ada.

Dalam bahasa pemrograman Go, pencarian nilai ekstrem dapat diimplementasikan menggunakan array dan perulangan. Modul algoritma dan pemrograman yang relevan menggunakan tipe data array statis, seperti `arrBalita [100]float64`, untuk menyimpan kumpulan berat badan balita. Subprogram atau fungsi digunakan untuk memisahkan tugas pengolahan data, seperti fungsi `hitungMinMax` untuk mencari nilai ekstrem, dan fungsi `rerata` untuk menghitung nilai rata-rata.

Konsep ini sangat bermanfaat di berbagai bidang, termasuk dalam pendataan kesehatan di Posyandu, untuk memantau status gizi anak berdasarkan berat badan. Selain itu, prinsip pencarian nilai ekstrem juga banyak digunakan dalam dunia bisnis, pendidikan, dan analisis data secara umum.

## II. GUIDED

### 1. Program ke 1

Source Code:

```
package main

import "fmt"


func main() {

var n int

fmt.Print("Masukkan jumlah tanaman: ")

fmt.Scan(&n)


var tinggitanaman [500]float64

for i := 0; i < n; i++ {

fmt.Printf("Masukkan tinggi tanaman ke-%d(cm):", i+1)

fmt.Scan(&tinggitanaman[i])

}


min, max := tinggitanaman[0], tinggitanaman[0]

for i := 0; i < n; i++ {

if tinggitanaman[i] < min {
```

```

min = tinggitanaman[i]
}
if tinggitanaman[i] > max {
max = tinggitanaman[i]
}
}

fmt.Printf("\nTinggi tanaman tertinggi: %.2f cm\n", max)

fmt.Printf("\nTinggi tanaman terendah: %.2f cm\n", min)
}

```

Output Program:

```

Masukkan jumlah tanaman: 2
Masukkan tinggi tanaman ke-1(cm):2
Masukkan tinggi tanaman ke-2(cm):3

Tinggi tanaman tertinggi: 3.00 cm
Tinggi tanaman terendah: 2.00 cm

```

Deskripsi Program: Program digunakan untuk menghitung tanaman yang paling tinggi dan terendah berdasarkan inputan yang dimasukkan ke indeks array tinggi tanaman

## 2. Program ke 2

Source Code:

```
package main

import "fmt"

func main() {
    var x, y int

    fmt.Print("Masukkan jumlah buku dan jumlah buku per rak : ")

    fmt.Scan(&x, &y)

    var hargabuku [500]float64

    fmt.Println("\nMasukkan harga setiap buku (dalam ribuan rp):")

    for i := 0; i < x; i++ {

        fmt.Scan(&hargabuku[i])

    }

    var hargaratarata []float64

    for i := 0; i < x; i += y {

        total := 0.0

        for j := i; j < i+y && j < x; j++ {

            total += hargabuku[j]

        }

        hargaratarata = append(hargaratarata, total/float64(y))

    }
```

```
min, max := hargaratarata[0], hargaratarata[0]

for _, harga := range hargabuku[:x] {
    if harga < min {
        min = harga
    }
    if harga > max {
        max = harga
    }
}

fmt.Printf("\nRata-rata harga per rak")

for _, avg := range hargaratarata {
    fmt.Printf("%.2f ", avg)

}

fmt.Printf("\nHarga termahal: %.2f Rp\n", max)
fmt.Printf("\nHarga termurah: %.2f Rp\n", min)
}
```

Output Program:

```

Masukkan jumlah buku dan jumlah buku per rak : 4 2

Masukkan harga setiap buku (dalam ribuan rp):
10000
100
100
100

Rata-rata harga per rak5050.00 100.00
Harga termahal: 10000.00 Rp

Harga termurah: 100.00 Rp

```

Deskripsi Program:Program ini dirancang untuk mengelola dan menganalisis data harga buku yang disimpan dalam rak-rak buku. Aplikasi ini memulai dengan meminta pengguna untuk memasukkan dua data utama: jumlah total buku yang akan diolah (dengan batas maksimal 500 buku) dan kapasitas setiap rak buku (berapa banyak buku yang dapat dimasukkan dalam satu rak).

### 3. Program ke 3

Source Code:

```

package main

import "fmt"

func main() {
    var n int

    fmt.Println("Masukkan jumlah siswa: ")
    fmt.Scan(&n)

    var nilaiSiswa [200]float64

    var totalNilai float64 = 0

```

```
fmt.Println("\nMasukkan nilai ujian masing-masing siswa:")
```

```
for i := 0; i < n; i++ {
```

```
    fmt.Scan(&nilaiSiswa[i])
```

```
    totalNilai += nilaiSiswa[i]
```

```
}
```

```
rataRata := totalNilai / float64(n)
```

```
min, max := nilaiSiswa[0], nilaiSiswa[0]
```

```
var diAtasRataRata int = 0
```

```
for _, nilai := range nilaiSiswa[:n] {
```

```
    if nilai < min {
```

```
        min = nilai
```

```
    }
```

```
    if nilai > max {
```

```
        max = nilai
```

```
    }
```

```
    if nilai > rataRata {
```

```
        diAtasRataRata++
```

```
    }
```

```
}
```

```
fmt.Printf("\nNilai terendah: %.0f\n", min)
```

```
fmt.Printf("Nilai tertinggi: %.0f\n", max)
```



```
fmt.Printf("Rata-rata kelas: %.2f\n", rataRata)

fmt.Printf("Jumlah siswa di atas rata-rata: %d\n", diAtasRataRata)

}
```

Output Program:

```
Masukkan jumlah siswa: 2

Masukkan nilai ujian masing-masing siswa:
10
100

Nilai terendah: 10
Nilai tertinggi: 100
Rata-rata kelas: 55.00
Jumlah siswa di atas rata-rata: 1
```

Deskripsi Program: Program ini merupakan sebuah aplikasi sederhana untuk menganalisis nilai ujian siswa. Program dimulai dengan meminta pengguna memasukkan jumlah siswa yang akan dianalisis, dengan batas maksimal 200 siswa. Setelah itu, program akan meminta input nilai ujian untuk masing-masing siswa satu per satu.

### III. UNGUIDED

#### 1. Program ke 1

Source Code:

```
package main

import "fmt"

func main() {

    var beratKelinci [1000]float64
    var n int

    fmt.Print("Masukkan jumlah kelinci: ")
    fmt.Scan(&n)

    if n <= 0 || n > 1000 {
        fmt.Println("Jumlah kelinci tidak valid")
        return
    }

    fmt.Println("Masukkan berat kelinci (kg):")
    for i := 0; i < n; i++ {
```

```

    fmt.Scan(&beratKelinci[i])

}

min, max := beratKelinci[0], beratKelinci[0]

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

fmt.Printf("Berat kelinci terkecil: %.2f kg\n", min)
fmt.Printf("Berat kelinci terbesar: %.2f kg\n", max)
}

```

Output Program:

```

Masukkan jumlah kelinci: 2
Masukkan berat kelinci (kg):
100
10
Berat kelinci terkecil: 10.00 kg
Berat kelinci terbesar: 100.00 kg

```

**Deskripsi Program:**Program ini dirancang untuk membantu peternak atau pedagang kelinci dalam mencatat dan menganalisis berat badan anak kelinci yang akan dijual.

## 2. Program ke 2

Source Code:

```
package main

import "fmt"

func main() {
    var beratIkan [1000]float64
    var x, y int

    fmt.Print("Masukkan jumlah ikan dan kapasitas wadah: ")
    fmt.Scan(&x, &y)

    if x <= 0 || x > 1000 {
        fmt.Println("Jumlah ikan tidak valid")
        return
    }

    if y <= 0 {
```

```
fmt.Println("Kapasitas wadah tidak valid")
```

```
return
```

```
}
```

```
fmt.Println("Masukkan berat ikan (kg):")
```

```
for i := 0; i < x; i++ {
```

```
    fmt.Scan(&beratIkan[i])
```

```
}
```

```
var totalWadah []float64
```

```
var totalSemua float64
```

```
jumlahWadah := 0
```

```
for i := 0; i < x; i += y {
```

```
    total := 0.0
```

```
    count := 0
```

```
    for j := i; j < i+y && j < x; j++ {
```

```
        total += beratIkan[j]
```

```
        count++
```

```
    }
```

```
    totalWadah = append(totalWadah, total)
```

```
    totalSemua += total
```

```
    jumlahWadah++
```

```

}

fmt.Println("\nTotal berat per wadah:")

for _, berat := range totalWadah {
    fmt.Printf("%.2f ", berat)
}

if jumlahWadah > 0 {
    rataRata := totalSemua / float64(jumlahWadah)
    fmt.Printf("\nRata-rata berat per wadah: %.2f kg\n", rataRata)
} else {
    fmt.Println("\nTidak ada wadah yang terisi")
}
}

```

Output Program:

```

Masukkan jumlah ikan dan kapasitas wadah: 2 2
Masukkan berat ikan (kg):
100
50

Total berat per wadah:
150.00
Rata-rata berat per wadah: 150.00 kg

```

Deskripsi Program: Program ini dirancang untuk membantu peternak atau pedagang kelinci dalam mencatat dan menganalisis berat badan anak kelinci yang akan dijual.

### 3. Program ke 3

### Source Code:

```
package main

import (
    "fmt"
)

const maxData = 100

type arrBalita [maxData]float64

func hitungMinMax(arrBerat arrBalita, n int, bMin, bMax *float64) {
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]
    for i := 1; i < n; i++ {
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
    }
}
```

```
func rerata(arrBerat arrBalita, n int) float64 {  
  
    var total float64  
  
    for i := 0; i < n; i++ {  
  
        total += arrBerat[i]  
  
    }  
  
    return total / float64(n)  
  
}  
  
func main() {  
  
    var data arrBalita  
  
    var n int  
  
  
    fmt.Print("Masukan banyak data berat balita : ")  
  
    fmt.Scan(&n)  
  
  
    for i := 0; i < n; i++ {  
  
        fmt.Printf("Masukan berat balita ke-%d: ", i+1)  
  
        fmt.Scan(&data[i])  
  
    }  
  
  
    var minBerat, maxBerat float64  
  
    hitungMinMax(data, n, &minBerat, &maxBerat)
```



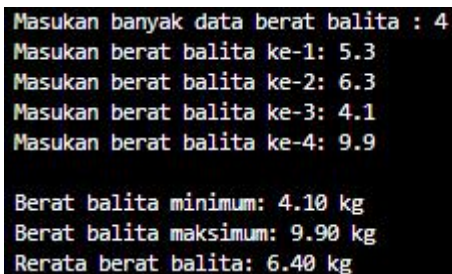
```
fmt.Printf("\nBerat balita minimum: %.2f kg\n", minBerat)

fmt.Printf("Berat balita maksimum: %.2f kg\n", maxBerat)

fmt.Printf("Rerata berat balita: %.2f kg\n", rerata(data, n))

}
```

Output Program:

A screenshot of a terminal window with a black background and green text. It shows the input and output of a Go program. The input consists of four lines of weights: 5.3, 6.3, 4.1, and 9.9. The output consists of three lines: minimum weight (4.10 kg), maximum weight (9.90 kg), and average weight (6.40 kg).

```
Masukan banyak data berat balita : 4
Masukan berat balita ke-1: 5.3
Masukan berat balita ke-2: 6.3
Masukan berat balita ke-3: 4.1
Masukan berat balita ke-4: 9.9

Berat balita minimum: 4.10 kg
Berat balita maksimum: 9.90 kg
Rerata berat balita: 6.40 kg
```

Deskripsi Program: Program ini digunakan untuk membantu Pos Pelayanan Terpadu (Posyandu) dalam mencatat dan mengolah data berat badan balita.

#### IV. KESIMPULAN

Modul ini menjelaskan konsep fundamental dalam pemrograman Go tentang tipe data bentukan dan struktur data array. Tipe bentukan mencakup alias untuk menyederhanakan nama tipe data yang ada dan struct untuk mengelompokkan data terkait menjadi satu kesatuan. Array dibahas sebagai struktur data statis dengan ukuran tetap, sementara slice diperkenalkan sebagai alternatif dinamis yang lebih fleksibel. Map dijelaskan sebagai struktur data asosiatif dengan kunci non-integer.

## **REFERENSI**