

**LAPORAN PRAKTIKUM  
ALGORITMA PEMROGRAMAN 2**

**MODUL 10**

**PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Oleh:

ANANDA BASKORO PUTRA

2311102187

IF 11 02

**S1 TEKNIK INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO 2024**

## I. DASAR TEORI

### a) Ide pencarian nilai max/min

Pencarian nilai maksimum dan minimum dalam himpunan data merupakan proses penting dalam analisis untuk menentukan elemen terbesar (maksimum) dan terkecil (minimum). Nilai maksimum adalah elemen yang tidak ada elemen lain dalam himpunan lebih besar darinya, sedangkan nilai minimum adalah elemen yang tidak ada elemen lain lebih kecil darinya. Algoritma pencarian dilakukan dengan membandingkan setiap elemen dalam himpunan data melalui perulangan. Prosesnya dimulai dengan menginisialisasi elemen pertama sebagai kandidat nilai maksimum dan minimum, kemudian setiap elemen dibandingkan; jika lebih besar dari nilai maksimum saat ini, nilai maksimum diperbarui, begitu pula untuk nilai minimum. Kompleksitas algoritma ini adalah  $O(n)O(n)O(n)$ , karena setiap elemen hanya diperiksa satu kali. Penerapan pencarian nilai ekstrem mencakup analisis data untuk menentukan batas atas dan bawah, pengolahan sinyal atau data numerik, serta deteksi outlier dalam statistik deskriptif. Selain itu, nilai maksimum dan minimum sering digunakan untuk menghitung range dan sebagai dasar pengambilan keputusan berdasarkan nilai-nilai ekstrem dalam data.

### b) Ide pencarian nilai ekstrim pada array bertipe dasar

Pencarian nilai ekstrem (maksimum dan minimum) dalam array bertipe dasar melibatkan operasi sederhana pada elemen-elemen yang memiliki tipe data primitif, seperti integer, float, atau karakter. Tujuan utamanya adalah menemukan elemen terbesar (maksimum) dan terkecil (minimum) dari sekumpulan data yang tersimpan dalam array. Prosesnya dimulai dengan menetapkan elemen pertama sebagai nilai awal maksimum dan minimum, kemudian menggunakan perulangan untuk membandingkan setiap elemen. Jika elemen yang diperiksa lebih besar dari nilai maksimum atau lebih kecil dari nilai minimum, nilainya diperbarui. Algoritma ini efisien dengan kompleksitas  $O(n)O(n)O(n)$ , karena setiap elemen hanya diperiksa satu kali. Array bertipe dasar mendukung akses elemen yang cepat dan sederhana, sehingga cocok untuk data numerik atau himpunan elemen sederhana. Penerapannya mencakup analisis data numerik, statistik deskriptif, deteksi anomali, serta pengendalian batas nilai dalam program, menjadikannya teknik fundamental dalam berbagai aplikasi pemrograman.

c) Ide pencarian nilai ekstrim pada array terstruktur

Pencarian nilai ekstrem (maksimum dan minimum) dalam array terstruktur dilakukan pada array yang elemen-elemennya terdiri dari tipe data kompleks, seperti struktur (struct) atau objek. Dalam kasus ini, pencarian nilai ekstrem biasanya dilakukan berdasarkan satu atau lebih atribut dalam struktur, bukan pada elemen secara keseluruhan. Proses pencarian dimulai dengan menentukan elemen pertama sebagai nilai awal maksimum dan minimum berdasarkan atribut tertentu, kemudian menggunakan perulangan untuk membandingkan setiap elemen berdasarkan atribut tersebut. Jika nilai atribut yang diperiksa lebih besar dari nilai maksimum atau lebih kecil dari nilai minimum, maka nilainya akan diperbarui. Algoritma ini dapat memiliki kompleksitas  $O(n)$ , karena setiap elemen dalam array terstruktur hanya perlu diperiksa satu kali. Pencarian nilai ekstrem dalam array terstruktur sering diterapkan dalam pengolahan data yang lebih kompleks, seperti sistem manajemen data, analisis statistik berdasarkan beberapa parameter, dan aplikasi yang melibatkan objek dengan banyak atribut.

## I. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

### Guided 1

```
package main

import (
    "fmt"
)

func main() {
    var N int

    var berat [1000]float64

    fmt.Print("Masukan jumlah anak kelinci: ")

    fmt.Scan(&N)

    fmt.Println("Masukan berat anak kelinci: ")

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

        fmt.Scan(&berat[i])

    }

    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]

}

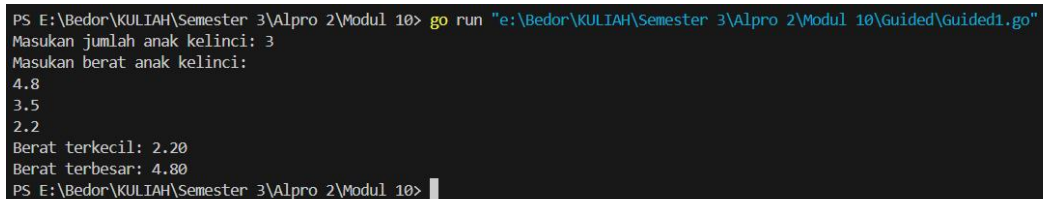
}

fmt.Printf("Berat terkecil: %.2f\n", min)

fmt.Printf("Berat terbesar: %.2f\n", max)

}
```

## Screenshoot Output



```
PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10> go run "e:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10\Guided\Guided1.go"
Masukan jumlah anak kelinci: 3
Masukan berat anak kelinci:
4.8
3.5
2.2
Berat terkecil: 2.20
Berat terbesar: 4.80
PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10> |
```

## Deskripsi Program

Program diatas digunakan untuk menentukan berat terkecil dan terbesar dari sejumlah anak kelinci. Program meminta pengguna untuk memasukkan jumlah anak kelinci dan berat masing-masing kelinci. Setelah itu, program menggunakan perulangan untuk membandingkan setiap berat dan menentukan nilai minimum (berat terkecil) dan maksimum (berat terbesar). Hasil akhirnya ditampilkan dalam format desimal dengan dua angka di belakang koma.

## Guided 2

```
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 // pembulatan ke atas jika x tidak habis dibagi 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 rata-rata berat tiap wadah

fmt.Println("Rata-rata berat tiap wadah:")

for _, total := range totalBeratWadah {

rataRata := total / float64(y)

fmt.Printf("%.2f ", rataRata)

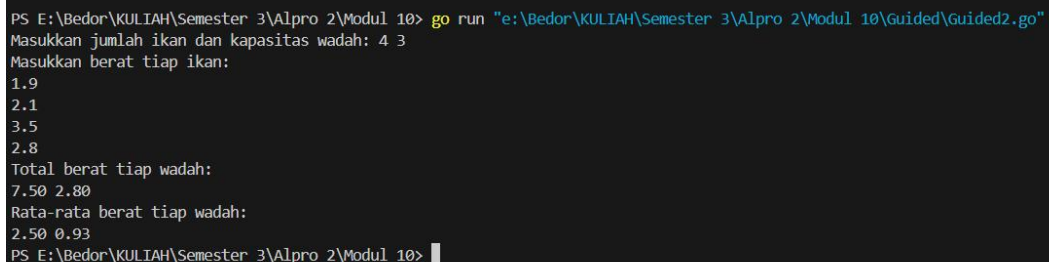
}

fmt.Println()

}

```

## Screenshoot Output



```

PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10> go run "e:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10\Guided\Guided2.go"
Masukkan jumlah ikan dan kapasitas wadah: 4 3
Masukkan berat tiap ikan:
1.9
2.1
3.5
2.8
Total berat tiap wadah:
7.50 2.80
Rata-rata berat tiap wadah:
2.50 0.93
PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10> █

```

## Deskripsi Program

Program ini untuk mengelompokkan ikan ke dalam sejumlah wadah berdasarkan kapasitas tertentu dan menghitung total serta rata-rata berat ikan di setiap wadah. Pengguna diminta memasukkan jumlah ikan, kapasitas wadah, dan berat masing-masing ikan. Program menghitung jumlah wadah yang diperlukan dengan pembulatan ke atas, mendistribusikan ikan ke wadah berdasarkan indeks, lalu menghitung total berat dan rata-rata berat ikan di setiap wadah. Hasil akhirnya berupa total dan rata-rata berat ikan per wadah yang ditampilkan dengan format desimal dua angka di belakang koma.

## II. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

### Unguided 1

```
package main
import (
    "fmt"
)
type arrBalita [100]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 rataRata(arrBerat arrBalita, n int) float64 {
    var total float64
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}
func main() {
    var n int
    var berat arrBalita
```



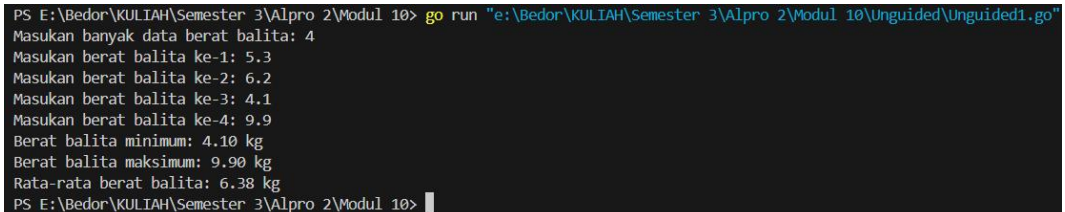
```

var bMin, bMax float64

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(&berat[i])
}
hitungMinMax(berat, n, &bMin, &bMax)
rata := rataRata(berat, n)
fmt.Printf("Berat balita minimum: %.2f kg\n", bMin)
fmt.Printf("Berat balita maksimum: %.2f kg\n", bMax)
fmt.Printf("Rata-rata berat balita: %.2f kg\n", rata)
}

```

## Screenshoot Output



```

PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10> go run "e:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10\Unguided\Unguided1.go"
Masukan banyak data berat balita: 4
Masukan berat balita ke-1: 5.3
Masukan berat balita ke-2: 6.2
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
Rata-rata berat balita: 6.38 kg
PS E:\Bedor\KULIAH\Semester 3\Alpro 2\Modul 10>

```

## Deskripsi Program

Program ini menganalisis data berat badan balita dengan menghitung berat minimum, maksimum, dan rata-rata. Pengguna memasukkan jumlah balita dan berat masing-masing balita. Program menggunakan fungsi `hitungMinMax` untuk mencari berat minimum dan maksimum, serta fungsi `rataRata` untuk menghitung rata-rata berat.