

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XI
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



**Disusun Oleh :
SYAHRUL ROMADHONI / 2311102261
S1 IF-11-05**

**Dosen Pengampu :
Arif Amrulloh, S.Kom.,M.Kom.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2024

I. DASAR TEORI

Definisi PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

Pencarian nilai ekstrim, seperti nilai maksimum dan minimum, adalah salah satu operasi dasar dalam pengolahan data. Di dalam bahasa pemrograman Go (Golang), terdapat beberapa cara untuk melakukan pencarian ini, baik menggunakan loop sederhana maupun goroutine untuk efisiensi.

1. Menggunakan Loop Sederhana

Salah satu cara paling langsung untuk mencari nilai maksimum dalam sebuah array adalah dengan menggunakan loop for. Berikut adalah contoh kode yang menunjukkan cara mencari nilai maksimum dari sebuah slice.

2. Menggunakan Goroutine dan Channel

Untuk meningkatkan efisiensi dalam pencarian nilai ekstrim pada dataset yang besar, kita dapat memanfaatkan goroutine dan channel. Berikut adalah contoh implementasi yang menggunakan dua goroutine untuk menghitung rata-rata dan nilai maksimum secara bersamaan.

II. GUIDED

Soal Studi Case

III. UNGUIDED

Soal Studi Case

1. BELUM JADI

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.

Masukan terdiri dari sekumpulan bilangan, yang mana bilangan pertama adalah bilangan bulat

N yang menyatakan banyaknya anak kelinci yang akan ditimbang beratnya. Selanjutnya N bilangan riil berikutnya adalah berat dari anak kelinci yang akan dijual.

Keluaran terdiri dari dua buah bilangan riil yang menyatakan berat kelinci terkecil dan terbesar.

Sourcecode

```
package main

package main

import (
    "fmt"
    "math"
)

func main() {
    var n int
    fmt.Println("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&n)

    if n <= 0 || n > 1000 {
        fmt.Println("Jumlah anak kelinci harus antara 1 dan 1000.")
        return
    }

    berat := make([]float64, n)
    fmt.Println("Masukkan berat kelinci satu per satu:")
    for i := 0; i < n; i++ {
        fmt.Printf("Berat kelinci ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }

    minBerat := math.MaxFloat64
```

```

maxBerat := -math.MaxFloat64

for _, b := range berat {
    if b < minBerat {
        minBerat = b
    }
    if b > maxBerat {
        maxBerat = b
    }
}

fmt.Printf("Berat terkecil: %.2f\n", minBerat)
fmt.Printf("Berat terbesar: %.2f\n", maxBerat)
}

```

Screenshoot Output

```

PS C:\Users\Syahrul\SEMESTER3\Modul8> go run "c:\Users\Syahrul\SEMESTER3\Modul8\Unguided1\main.go"
Masukkan jumlah anak kelinci: 5
Masukkan berat kelinci satu per satu:
Berat kelinci ke-1: 2.5
Berat kelinci ke-2: 3.8
Berat kelinci ke-3: 1.2
Berat kelinci ke-4: 4.6
Berat kelinci ke-5: 3.0
Berat terkecil: 1.20
Berat terbesar: 4.60
PS C:\Users\Syahrul\SEMESTER3\Modul8>

```

Deskripsi Program

Program di atas adalah aplikasi berbasis Golang yang digunakan untuk mencari berat terkecil dan terbesar dari sekelompok data berat anak kelinci yang dimasukkan oleh pengguna. Program meminta jumlah anak kelinci NNN (dengan batas maksimal 1000) dan menerima berat masing-masing kelinci satu per satu. Data berat tersebut disimpan dalam sebuah array, kemudian program menghitung nilai minimum dan maksimum dengan membandingkan setiap berat yang dimasukkan. Hasil akhirnya adalah dua nilai yang mewakili berat terkecil dan terbesar, yang kemudian ditampilkan ke layar dengan format dua angka desimal. Program ini dirancang untuk memastikan hasilnya akurat dan mudah dibaca oleh pengguna.

2.

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.

Masukan terdiri dari dua baris, yang mana baris pertama terdiri dari dua bilangan bulat x dan y. Bilangan x menyatakan banyaknya ikan yang akan

dijual, sedangkan y adalah banyaknya ikan yang akan dimasukan ke dalam wadah. Baris kedua terdiri dari sejumlah x Jumlah bilangan riil yang menyatakan banyaknya ikan yang akan dijual.

Keluaran terdiri dari dua baris. Baris pertama adalah kumpulan bilangan riil yang menyatakan total berat ikan di setiap wadah (jumlah wadah tergantung pada nilai x dan y, urutan ikan yang dimasukan ke dalam wadah sesuai urutan pada masukan baris ke-2). Baris kedua adalah sebuah bilangan riil yang menyatakan berat rata-rata ikan di setiap wadah.

Sourcecode

```
package main

import (
    "fmt"
)

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

    if x <= 0 || y <= 0 || x*y > 1000 {
        fmt.Println("Input tidak valid. Pastikan x > 0, y > 0, dan total ikan tidak melebihi 1000.")
        return
    }

    beratIkan := make([]float64, x*y)
    fmt.Println("Masukkan berat ikan satu per satu:")
    for i := 0; i < x*y; i++ {
        fmt.Printf("Berat ikan ke-%d: ", i+1)
        fmt.Scan(&beratIkan[i])
    }

    totalBeratPerWadah := make([]float64, x)
    for i := 0; i < len(beratIkan); i++ {
        wadah := i / y
        totalBeratPerWadah[wadah] += beratIkan[i]
    }

    rataRataPerWadah := make([]float64, x)
    for i := 0; i < x; i++ {
        rataRataPerWadah[i] = totalBeratPerWadah[i] / float64(y)
    }

    fmt.Println("\nTotal berat di setiap wadah:")
}
```

```

    for i, total := range totalBeratPerWadah {
        fmt.Printf("Wadah %d: %.2f\n", i+1, total)
    }

    fmt.Println("\nRata-rata berat di setiap wadah:")
    for i, rata := range rataRataPerWadah {
        fmt.Printf("Wadah %d: %.2f\n", i+1, rata)
    }
}

```

Screenshoot Output

```

PS C:\Users\Syahrul\SEMESTER3\Modul18> go run "c:\Users\Syahrul\SEMESTER3\Modul18\Unguided2\main.go"
Masukkan jumlah wadah (x): 2
Masukkan kapasitas ikan per wadah (y): 3
Masukkan berat ikan satu per satu:
Berat ikan ke-1: 1.5
Berat ikan ke-2: 2.0
Berat ikan ke-3: 2.5
Berat ikan ke-4: 3.0
Berat ikan ke-5: 3.5
Berat ikan ke-6: 4.0

Total berat di setiap wadah:
Wadah 1: 6.00
Wadah 2: 10.50

Rata-rata berat di setiap wadah:
Wadah 1: 2.00
Wadah 2: 3.50
PS C:\Users\Syahrul\SEMESTER3\Modul18>

```

Deskripsi Program

Program di atas digunakan untuk mendistribusikan berat ikan ke dalam beberapa wadah dan menghitung total serta rata-rata berat di setiap wadah. Pengguna memasukkan jumlah wadah (xxx) dan kapasitas ikan per wadah (yyy), diikuti oleh berat ikan sebanyak $x \times y$ \times yx \times y. Program mendistribusikan berat ikan ke wadah berdasarkan urutan input, menghitung total berat setiap wadah dengan menjumlahkan berat ikan di dalamnya, dan menghitung rata-rata berat dengan membagi total berat oleh kapasitas wadah (yyy). Hasil akhirnya berupa total berat dan rata-rata berat untuk setiap wadah, yang ditampilkan dengan format desimal dua angka. Program ini membantu menganalisis distribusi berat ikan secara efisien dan terorganisir.

3.

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 terkecil, terbesar, dan reratanya.

Buatlah program dengan spesifikasi subprogram sebagai berikut:

```
type arrBalita [100]float64

func hitungMinMax(arrBerat arrBalita; bMin, bMax *float64) {
  /* I.S. Terdefinisi array dinamis arrBerat
```

```
  Proses: Menghitung berat minimum dan maksimum dalam array
  F.S. Menampilkan berat minimum dan maksimum balita */
  ...
}

function rerata (arrBerat arrBalita) real {
  /* menghitung dan mengembalikan rerata berat balita dalam array */
  ...
}
```

Perhatikan sesi interaksi pada contoh berikut ini (teks bergaris bawah adalah input/read)

```
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
Rerata berat balita: 6.38 kg
```

Sourcecode

```
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 {
    total := 0.0
    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("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)

    if n <= 0 || n > 100 {
        fmt.Println("Jumlah balita harus antara 1 dan 100.")
        return
    }

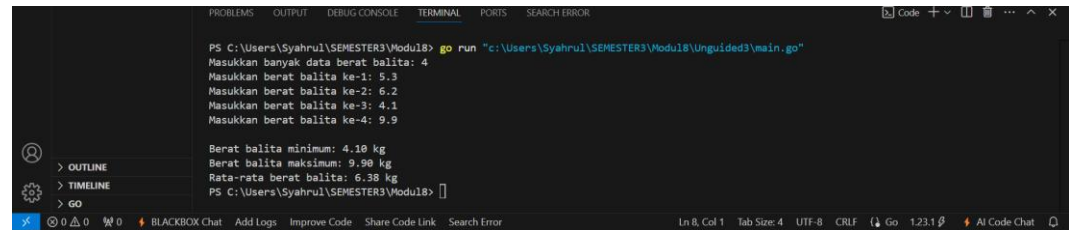
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan berat balita ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }

    hitungMinMax(berat, n, &bMin, &bMax)
    rata := ratarata(berat, n)

    fmt.Printf("\nBerat 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



```
PS C:\Users\Syahrul\SEMESTER3\Modul8> go run "c:\Users\Syahrul\SEMESTER3\Modul8\Unguided3\main.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 C:\Users\Syahrul\SEMESTER3\Modul8>
```

Deskripsi Program

Program di atas adalah aplikasi Golang yang digunakan untuk menganalisis data berat balita, mencakup pencarian berat minimum, maksimum, dan rata-rata. Pengguna diminta untuk memasukkan jumlah balita dan berat masing-masing balita, yang kemudian disimpan dalam array. Program memiliki dua fungsi utama: `hitungMinMax`, yang menghitung berat minimum dan maksimum dengan membandingkan setiap elemen dalam array, serta `ratarata`, yang menghitung rata-rata berat dengan menjumlahkan semua berat dan membaginya dengan jumlah balita. Hasil akhir berupa berat minimum, maksimum, dan rata-rata ditampilkan dalam format desimal dua angka, membantu mempermudah analisis data berat balita.

KESIMPULAN

Pencarian nilai ekstrem (minimum dan maksimum) dalam Golang adalah proses untuk menemukan nilai terkecil dan terbesar dari sekumpulan data yang disimpan dalam array. Proses ini sangat berguna dalam berbagai aplikasi, seperti analisis data berat, distribusi data, dan perhitungan statistik. Berikut adalah poin-poin utama yang dapat disimpulkan dari materi ini:

1. **Struktur Data:** Array adalah struktur data yang cocok digunakan untuk menyimpan dan mengelola kumpulan data tetap, seperti berat balita, ikan, atau benda lainnya. Golang memungkinkan kita mendefinisikan array dengan tipe data tertentu, seperti float64 untuk data numerik.
2. **Iterasi dan Perbandingan:** Pencarian nilai ekstrem dilakukan dengan cara iterasi melalui setiap elemen array, membandingkan nilai elemen tersebut dengan nilai minimum dan maksimum sementara. Teknik ini sederhana tetapi efektif untuk data berukuran kecil hingga menengah.
3. **Modularisasi dengan Fungsi:** Golang mendorong penggunaan fungsi untuk memisahkan logika. Fungsi seperti `hitungMinMax` dan `ratarata` mempermudah pengelolaan kode, meningkatkan keterbacaan, dan memungkinkan pengujian terpisah untuk setiap bagian program.
4. **Kombinasi Analisis Data:** Selain mencari nilai ekstrem, perhitungan rata-rata sering digunakan untuk memberikan wawasan tambahan terhadap data. Ini membantu pengguna memahami distribusi data, baik untuk pengambilan keputusan maupun evaluasi.
5. **Validasi Input:** Validasi jumlah data dan elemen input sangat penting untuk mencegah kesalahan runtime, seperti melampaui batas array atau memasukkan nilai yang tidak valid.
6. **Aplikasi Praktis:** Materi ini relevan untuk berbagai aplikasi dunia nyata, seperti menganalisis berat badan balita di posyandu, distribusi ikan dalam wadah, atau pengelompokan data lain yang membutuhkan perhitungan nilai ekstrem dan statistik dasar.

DAFTAR PUSTAKA

<https://dasarpemrogramangolang.novalagung.com/A-channel-select.html>