

**LAPORAN PRAKTIKUM  
PEMROGRAMAN BERORIENTASI OBJEK  
MODUL X  
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Oleh:

OKTAVANIA AYU RAHMADANTY

2311102240

S1IF-11-02

**S1 TEKNIK INFORMATIKA  
UNIVERSITAS TELKOM PURWOKERTO**

**2024**

## **I. DASAR TEORI**

### **I.1 Ide Nilai Max/Min**

Pencarian adalah suatu proses yang lazim dilakukan di dalam kehidupan sehari-hari. Contoh penggunaannya dalam kehidupan nyata sangat beragam, misalnya pencarian file di dalam directory komputer, pencarian suatu teks di dalam sebuah dokumen, pencarian buku pada rak buku, dan contoh lainnya. Pertama pada modul ini akan dipelajari salah satu algoritma pencarian nilai terkecil atau terbesar pada sekumpulan data, atau biasa disebut pencarian nilai ekstrim.

Ide algoritma sederhana sekali. Karena data harus diproses secara sekuensial, maka nilai atau indeks ke nilai maksimum dari data yang telah diproses disimpan untuk dibandingkan dengan data berikutnya. Nilai yang berhasil disimpan sampai algoritma tersebut berakhir adalah nilai maksimum yang dicari. Adapun algoritmanya secara umum adalah sebagai berikut:

- 1) Jadikan data pertama sebagai nilai ekstrim
- 2) Lakukan validasi nilai ekstrim dari data kedua hingga data terakhir.

Apabila nilai ekstrim tidak valid, maka update nilai ekstrims tersebut t dengan dengan 1 data data data yang yang  
dicek.

Telkom University

- 3) Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid, matics lab

Berikut ini adalah notasi dalam pseudocode dan bahasa Go, misalnya untuk pencarian nilai terbesar atau maksimum:

	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	}

## I.2 Pencarian Nilai Ekstrim pada Array Bertipe Data Dasar

Misalnya terdefinisi sebuah array of integer dengan kapasitas 2023, dan array terisi sejumlah N bilangan bulat, kemudian pencarian nilai terkecil dilakukan pada array tersebut. Perhatikan potongan program dalam bahasa Go berikut ini!

```

5  type arrInt [2023]int
..  ...
15
16 func terkecil_1(tabInt arrInt, n int) int {
17  /* mengembalikan nilai terkecil yang terdapat di dalam tabInt yang berisi n
18  bilangan bulat */
19      var min int = tabInt[0]          // min berisi data pertama
20      var j int = 1                    // pencarian dimulai dari data berikutnya
21      for j < n {
22          if min > tabInt[j] {          // pengecekan apakah nilai minimum valid
23              min = tabInt[j]          // update nilai minimum dengan yang valid
24          }
25          j = j + 1
26      }
27      return min                       // returnkan nilai minimumnya
28  }

```

Potongan program di atas sedikit berbeda dengan sebelumnya karena penggunaan indeks array pada bahasa Go di mulai dari nol atau "0" seperti penjelasan pada modul 9. Selanjutnya, pada penjelasan di awal bab 3 telah disampaikan bahwa pada pencarian yang terpenting adalah posisi atau indeks dari nilai yang dicari dalam kumpulan data atau array. Oleh karena itu modifikasi pada program di atas dapat dilihat pada potongan program berikut ini!

## I.3. Pencarian Nilai Ekstrim pada Array Bertipe Data Tersruktur

Pada kasus yang lebih kompleks pencarian ekstrim dapat juga dilakukan, misalnya mencari data mahasiswa dengan nilai terbesar, mencari lagu dengan durasi terlama, mencari pembalap yang memiliki catatan waktu balap tercepat, dan sebagainya. Sebagai contoh misalnya terdapat array yang digunakan untuk menyimpan data mahasiswa, kemudian terdapat fungsi IPK yang digunakan untuk mencari data mahasiswa dengan IPK tertinggi.

```
..  ...
5  type mahasiswa struct {
..    nama, nim, kelas, jurusan string
..    ipk float64
.. }
.. type arrMhs [2023]mahasiswa
.. ...
15
16 func IPK_1(T arrMhs, n int) float64 {
17     /* mengembalikan ipk terkecil yang dimiliki mahasiswa pada array T yang berisi
18     n mahasiswa */
19     var tertinggi float64 = T[0].ipk
20     var j int = 1
21     for j < n {
22         if tertinggi < T[j].ipk {
23             tertinggi = T[j].ipk
24         }
25         j = j + 1
26     }
27     return tertinggi
28 }
```

Apabila diperhatikan potongan program di atas, maka kita akan memperoleh ipk tertinggi, tetapi kita tidak memperoleh identitas mahasiswa dengan ipk tertinggi tersebut. Maka seperti penjelasan yang sudah diberikan sebelumnya, maka pencarian yang dilakukan bisa mengembalikan indeks mahasiswa dengan ipk tertinggi tersebut. Berikut ini adalah modifikasinya!

```

. ...
i type mahasiswa struct {
.     nama, nim, kelas, jurusan string
.     ipk float64
. }
. type arrMhs [2023]mahasiswa
. ...
5
6 func IPK_2(T arrMhs, n int) int {
7     /* mengembalikan indeks mahasiswa yang memiliki ipk tertinggi pada array T yang
8     berisi n mahasiswa */
9     var idx int = 0
10    var j int = 1
11    for j < n {
12        if T[idx].ipk < T[j].ipk {
13            idx = j
14        }
15        j = j + 1
16    }
17    return idx
18 }

```

Sehingga melalui algoritma di atas, identitas mahasiswa dapat diperoleh, misalnya T[idx].nama, T[idx].kelas, hingga T[idx].jurusan.

## 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])
    }

    min := berat[0]
    max := berat[0]

    for i := 1; i < N; i++ {
        if berat[i] < min {
            min = berat[i]
        }
    }
}
```

```

    }

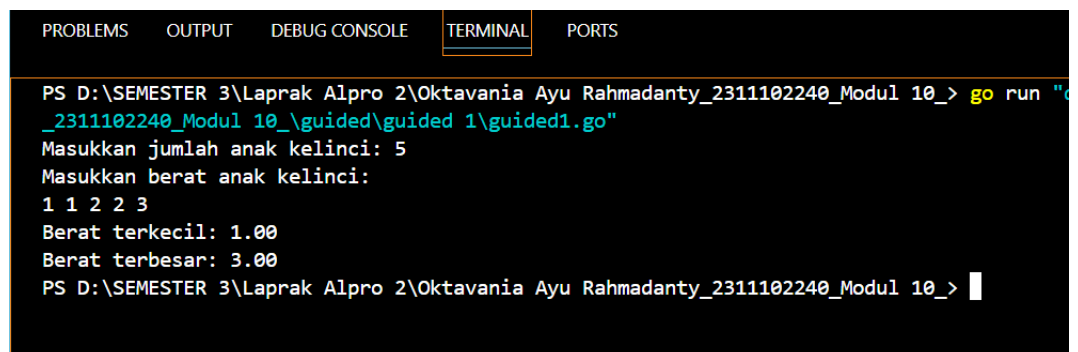
    if berat[i] > max {
        max = berat[i]
    }
}

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

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

```

### Screenshot hasil program



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240_Modul 10_> go run "d
_2311102240_Modul 10_\guided\guided 1\guided1.go"
Masukkan jumlah anak kelinci: 5
Masukkan berat anak kelinci:
1 1 2 2 3
Berat terkecil: 1.00
Berat terbesar: 3.00
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240_Modul 10_>

```

### Penjelasan:

Program di atas ditulis dalam bahasa Go untuk menentukan berat terkecil dan terbesar dari sejumlah anak kelinci. Pengguna diminta untuk memasukkan jumlah anak kelinci dan berat masing-masing anak kelinci. Program kemudian menggunakan perulangan untuk memeriksa setiap berat yang dimasukkan, membandingkannya dengan nilai minimum dan maksimum yang awalnya diinisialisasi dengan berat anak kelinci pertama. Hasil akhirnya adalah berat terkecil dan terbesar yang ditampilkan dengan format desimal dua angka di belakang koma.

## Guided 2

### Source Code

```
package main

import (
    "fmt"
)

func main() {
    var x, y int

    fmt.Printf("masukkan jumlah ikan dan kapasitas wadah: ")

    fmt.Scan(&x, &y)

    berat := make([]float64, x)

    fmt.Printf("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]
    }
}
```



```

    fmt.Println("total berat tiap wadah: ")

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

    fmt.Println()

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

    for _, total := range totalBeratWadah {
        ratarata := total / float64(y)

        fmt.Printf("%.2f", ratarata)
    }

    fmt.Println()
}

```

### Screenshot hasil program

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
ia Ayu Rahmadanty_2311102240_Modul 10_\guided\guided 2\guided2.go"
# command-line-arguments
guided\guided 2\guided2.go:18:1: syntax error: non-declaration statement outside function
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240_Modul 10_> go run "d:\
ia Ayu Rahmadanty_2311102240_Modul 10_\guided\guided 2\guided2.go"
masukkan jumlah ikan dan kapasitas wadah: 1 3
masukkan berat tiap ikan: 12
total berat tiap wadah:
12.00
rata rata berat tiap wadah:
4.00
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240_Modul 10_> 

```

### Penjelasan:

Program ini menghitung total berat ikan dan rata-rata berat ikan di setiap wadah dengan kapasitas tertentu. Pengguna diminta memasukkan jumlah ikan, kapasitas wadah, serta berat masing-masing ikan. Program menghitung jumlah wadah yang diperlukan, kemudian mendistribusikan

berat ikan ke wadah-wadah berdasarkan urutan pengisian. Setelah itu, program menampilkan total berat tiap wadah dan rata-rata berat ikan dalam setiap wadah dengan membagi total berat dengan kapasitas wadah. Output akhirnya berupa daftar total dan rata-rata berat untuk semua wadah yang digunakan.

### III. UNGUIDED

#### Unguided 1

##### Source Code

```
package main

import (
    "fmt"
)

func hitungMinMax(arrBalita []float64) (float64, float64) {
    min := arrBalita[0]
    max := arrBalita[0]

    for _, berat := range arrBalita {
        if berat < min {
            min = berat
        }
        if berat > max {
            max = berat
        }
    }

    return min, max
}

func rerata(arrBalita []float64) float64 {
    total := 0.0

    for _, berat := range arrBalita {
        total += berat
    }

    return total / float64(len(arrBalita))
}

func main() {
    var n int
    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)

    arrBalita := make([]float64, n)
```

```

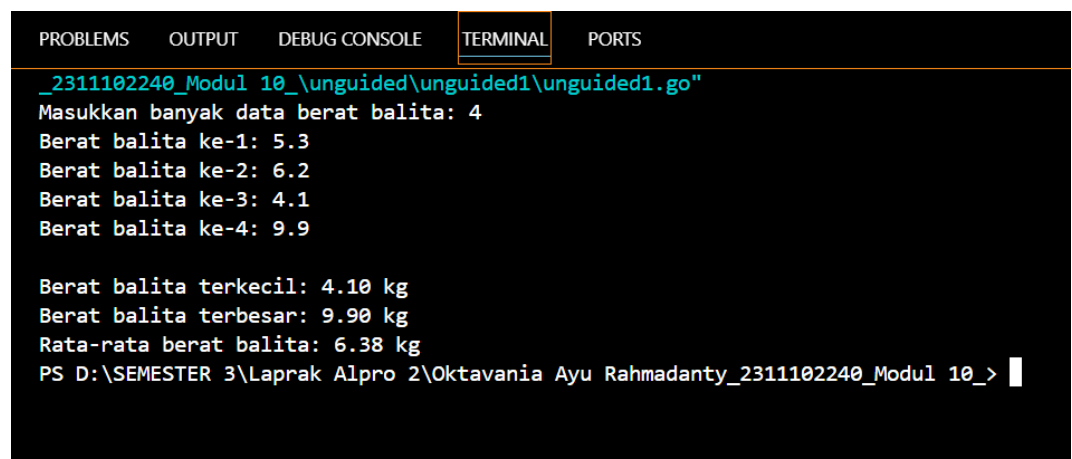
    for i := 0; i < n; i++ {
        fmt.Printf("Berat balita ke-%d: ", i+1)
        fmt.Scan(&arrBalita[i])
    }

    min, max := hitungMinMax(arrBalita)
    rata := rerata(arrBalita)

    fmt.Printf("\nBerat balita terkecil: %.2f kg\n",
min)
    fmt.Printf("Berat balita terbesar: %.2f kg\n",
max)
    fmt.Printf("Rata-rata berat balita: %.2f kg\n",
rata)
}

```

#### Screenshot hasil program



```

_2311102240_Modul 10_\unguided\unguided1\unguided1.go"
Masukkan banyak data berat balita: 4
Berat balita ke-1: 5.3
Berat balita ke-2: 6.2
Berat balita ke-3: 4.1
Berat balita ke-4: 9.9

Berat balita terkecil: 4.10 kg
Berat balita terbesar: 9.90 kg
Rata-rata berat balita: 6.38 kg
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240_Modul 10_>

```

#### Penjelasan:

Program ini bertujuan untuk menghitung berat balita terkecil, terbesar, dan rata-rata dari sejumlah data berat balita yang diinputkan. Pengguna diminta memasukkan jumlah balita dan berat masing-masing balita. Program menggunakan fungsi `hitungMinMax` untuk mencari berat terkecil dan terbesar, serta fungsi `rerata` untuk menghitung rata-rata berat. Setelah semua data diproses, program menampilkan hasil berupa berat terkecil, berat terbesar, dan rata-rata berat balita dengan format desimal dua angka di belakang koma.