

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VIII**

**PENCARIAN NILAI EKSTRIM PADA  
HIMPUNAN DATA**



**Disusun Oleh :**

**AFRIZAL DWI NUGRAHA / 2311102136**

**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

### 1. Ide Pencarian 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 dengan data yang dicek.
- 3) Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid.

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	}

### 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!

```

..  ...
5  type arrInt [2023]int
..  ...
15
16 func terkecil_2(tabInt arrInt, n int) int {
17     /* mengembalikan indeks nilai terkecil yang terdapat di dalam tabInt yang berisi
18     n bilangan bulat */
19     var idx int = 0           // idx berisi indeks data pertama
20     var j int = 1           // pencarian dimulai dari data berikutnya
21     for j < n {
22         if tabInt[idx] > tabInt[j] { // pengecekan apakah nilai minimum valid
23             idx = j               // update nilai minimum dengan yang valid
24         }
25         j = j + 1
26     }
27     return idx               // returnkan indeks nilai minimumnya
28 }

```

### 3. Pencarian Nilai Ekstrim pada Array Bertipe Data Terstruktur

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!

```
.. ...
5  type mahasiswa struct {
..     nama, nim, kelas, jurusan string
..     ipk float64
.. }
.. type arrMhs [2023]mahasiswa
.. ...
15
16 func IPK_2(T arrMhs, n int) int {
17     /* mengembalikan indeks mahasiswa yang memiliki ipk tertinggi pada array T yang
18     berisi n mahasiswa */
19     var idx int = 0
20     var j int = 1
21     for j < n {
22         if T[idx].ipk < T[j].ipk {
23             idx = j
24         }
25         j = j + 1
26     }
27     return idx
28 }
```

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

## II. UNGUIDED

1. 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

import (
    "fmt"
    "math"
)

func main() {
    var N int
    fmt.Print("Masukkan jumlah kelinci (N): ")
    fmt.Scan(&N)

    // Validasi jumlah kelinci
    if N <= 0 || N > 1000 {
        fmt.Println("Jumlah kelinci harus antara 1 hingga 1000.")
        return
    }

    // Deklarasi array untuk menyimpan berat kelinci
    weights := make([]float64, N)

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

    // Inisialisasi nilai minimum dan maksimum
    minWeight := math.MaxFloat64
```

```

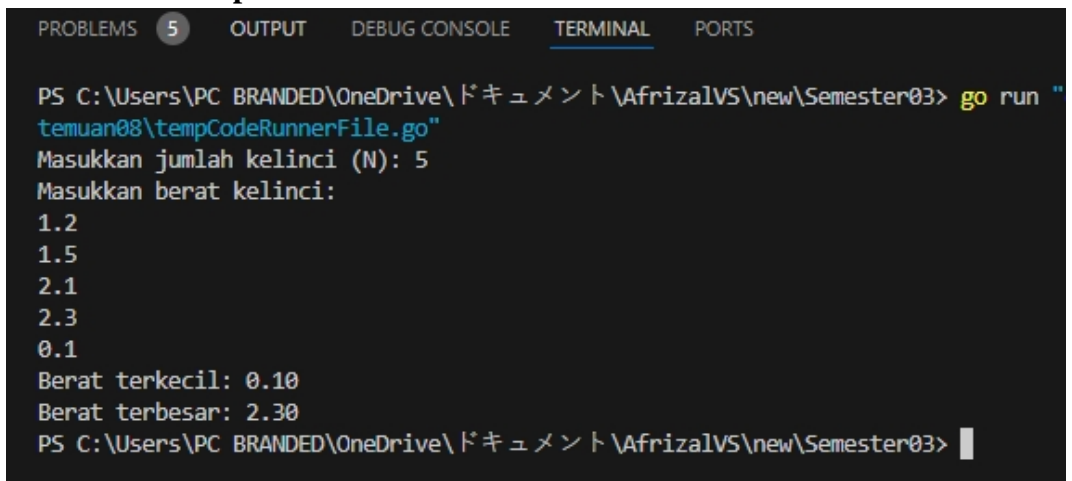
maxWeight := -math.MaxFloat64

// Cari nilai terkecil dan terbesar
for _, weight := range weights {
    if weight < minWeight {
        minWeight = weight
    }
    if weight > maxWeight {
        maxWeight = weight
    }
}

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

```

### Screenshot Output



```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03> go run "
temuan08\tempCodeRunnerFile.go"
Masukkan jumlah kelinci (N): 5
Masukkan berat kelinci:
1.2
1.5
2.1
2.3
0.1
Berat terkecil: 0.10
Berat terbesar: 2.30
PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03>

```

### Deskripsi Program

Program ini digunakan untuk mengolah data berat anak kelinci yang akan dijual. Ini menerima informasi tentang jumlah anak kelinci dan berat masing-masing kelinci, dan kemudian menghitung berat terkecil dan terbesar dari data yang dimasukkan. Program ini dapat menangani hingga seribu data berat kelinci.

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 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

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

    // Validasi input
    if x <= 0 || x > 1000 || y <= 0 {
        fmt.Println("Input tidak valid. Jumlah ikan harus 1-1000 dan kapasitas wadah harus lebih dari 0.")
        return
    }

    // Input berat ikan
    weights := make([]float64, x)
    fmt.Println("Masukkan berat ikan:")
    for i := 0; i < x; i++ {
```

```

        fmt.Scan(&weights[i])
    }

    // Hitung total berat ikan di setiap wadah
    var totalWeights []float64
    var currentWeight float64
    for i, weight := range weights {
        currentWeight += weight
        if (i+1)%y == 0 || i == x-1 {
            totalWeights = append(totalWeights,
currentWeight)
            currentWeight = 0
        }
    }

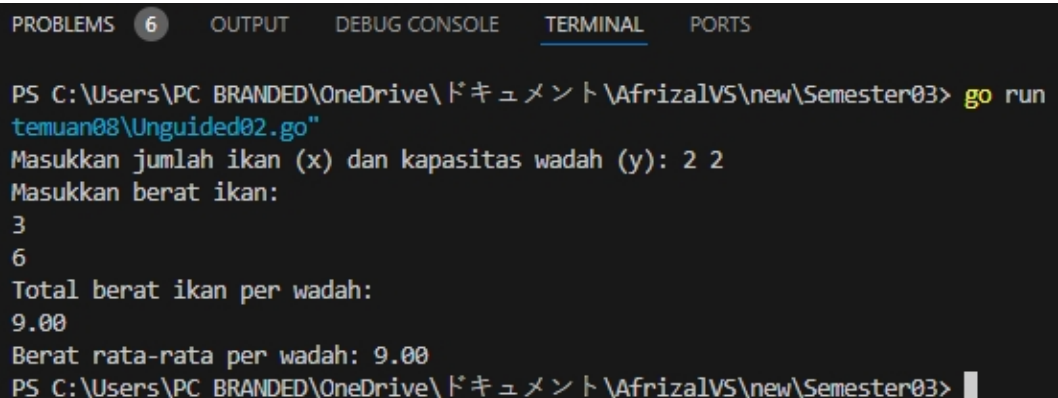
    // Hitung rata-rata berat ikan per wadah
    totalSum := 0.0
    for _, total := range totalWeights {
        totalSum += total
    }
    averageWeight := totalSum / float64(len(totalWeights))

    // Output total berat ikan per wadah
    fmt.Println("Total berat ikan per wadah:")
    for _, total := range totalWeights {
        fmt.Printf("%.2f ", total)
    }
    fmt.Println()

    // Output rata-rata berat per wadah
    fmt.Printf("Berat rata-rata per wadah: %.2f\n",
averageWeight)
}

```

## Screenshoot Output



```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03> go run
temuan08\Unguided02.go
Masukkan jumlah ikan (x) dan kapasitas wadah (y): 2 2
Masukkan berat ikan:
3
6
Total berat ikan per wadah:
9.00
Berat rata-rata per wadah: 9.00
PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03>

```



**Deskripsi Program**

Berdasarkan berat ikan yang akan dijual ke pasar, program ini menghitung berat total ikan di setiap wadah dan berat rata-rata ikan per wadah dengan menggunakan array, yang dapat menyimpan hingga 1000 ikan. Jumlah ikan dan kapasitas setiap wadah yang diberikan pengguna adalah dasar pemrosesan data program.

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"  
)  
  
// Definisi tipe data untuk array berat balita  
type arrBalita [100]float64  
  
// Fungsi untuk menghitung nilai minimum dan maksimum dalam array  
func hitungMinMax(arrBerat arrBalita, N int, Min, Max *float64) {  
    *Min = arrBerat[0]
```

```

    *Max = arrBerat[0]

    for i := 1; i < N; i++ {
        if arrBerat[i] < *Min {
            *Min = arrBerat[i]
        }
        if arrBerat[i] > *Max {
            *Max = arrBerat[i]
        }
    }
}

// Fungsi untuk menghitung rata-rata berat balita dalam array
func rata(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 Min, Max float64

    // Input jumlah balita
    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&N)

    // Validasi jumlah balita
    if N <= 0 || N > 100 {
        fmt.Println("Jumlah balita harus antara 1 hingga 100.")
        return
    }

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

    // Hitung nilai minimum, maksimum, dan rata-rata

```

```

hitungMinMax(berat, N, &Min, &Max)
rerata := rata(berat, N)

// Output hasil
fmt.Printf("Berat balita minimum: %.2f kg\n", Min)
fmt.Printf("Berat balita maksimum: %.2f kg\n", Max)
fmt.Printf("Berat balita rata-rata: %.2f kg\n", rerata)
}

```

### Screenshoot Output

```

PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03> go run "
temuan08\tempCodeRunnerFile.go"
Masukkan banyak data berat balita: 4
Masukkan berat balita ke-1: 5.7
Masukkan berat balita ke-2: 5.9
Masukkan berat balita ke-3: 6.4
Masukkan berat balita ke-4: 7.2
Berat balita minimum: 5.70 kg
Berat balita maksimum: 7.20 kg
Berat balita rata-rata: 6.30 kg
PS C:\Users\PC BRANDED\OneDrive\ドキュメント\AfrizalVS\new\Semester03>

```

### Deskripsi Program

Petugas Pos Pelayanan Terpadu (Posyandu) dapat mencatat, mengolah, dan menganalisis berat balita (dalam kilogram) dengan program ini. Program ini dapat menghitung berat balita terkecil, terbesar, dan rata-rata dari sekumpulan data yang dimasukkan dengan menggunakan array.

### **Daftar Pustaka :**

1. MODUL 10. PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA