

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XI**

**PENCARIAN NILAI EXTREME PADA HIMPUNAN DATA**



**Disusun Oleh :**

**Dimas Akal Hernanda/2311102249**

**IF-11-06**

**Dosen Pengampu :**

**ABEDNEGO DWI SEPTIADI**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. DASAR TEORI

### Pencarian Nilai Ekstrem pada Himpunan Data (Algoritma dan Pemrograman)

Pencarian nilai ekstrem bertujuan untuk menemukan nilai terbesar (maksimum) atau terkecil (minimum) dalam sebuah himpunan data. Algoritma yang umum digunakan adalah:

1. Pencarian Linear: Mengiterasi setiap elemen dalam himpunan untuk mencari nilai ekstrem. Kompleksitas waktu algoritma ini adalah  $O(n)$ , di mana  $n$  adalah jumlah elemen dalam data.
2. Struktur Data Terurut: Jika data sudah terurut, nilai maksimum atau minimum bisa ditemukan langsung di elemen pertama atau terakhir dengan waktu  $O(1)$ .
3. Pencarian dalam Struktur Lain: Dalam struktur seperti graf atau matriks, pencarian nilai ekstrem dilakukan dengan memeriksa elemen-elemen terkait sesuai kriteria tertentu.

Aplikasi: Pencarian nilai ekstrem digunakan dalam analisis data, pengoptimalan, dan pemrosesan citra. Algoritma sederhana seperti pencarian linear cukup efisien untuk banyak kasus, tetapi untuk data yang lebih besar atau terstruktur, teknik lain mungkin diperlukan.

## II. Guided

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

### Source Code

```
package main

import "fmt"

// Mendeklarasikan tipe data array arrInt dengan panjang 2023
type arrInt [2023]int

// Fungsi untuk mencari indeks elemen terkecil dalam array
func terkecil(tabInt arrInt, n int) int {
    var idx int = 0 // idx menyimpan indeks elemen terkecil
    var j int = 1
    for j < n {
        if tabInt[idx] > tabInt[j] {
            idx = j // Simpan indeks j jika elemen di indeks j lebih kecil
        }
        j = j + 1
    }
    return idx
}
```

```
// Fungsi main untuk menguji fungsi terkecil

func main() {

    var n int

    var tab arrInt


    // Meminta input jumlah elemen array

    fmt.Print("Masukkan jumlah elemen (maks 2023): ")

    fmt.Scan(&n)


    // Validasi input jumlah elemen

    if n < 1 || n > 2023 {

        fmt.Println("Jumlah elemen harus antara 1 dan 2023.")

        return

    }


    // Memasukkan elemen-elemen array

    fmt.Println("Masukkan elemen-elemen array:")

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

        fmt.Print("Elemen ke-", i+1, ": ")

        fmt.Scan(&tab[i])

    }


    // Memanggil fungsi terkecil untuk menemukan indeks elemen terkecil

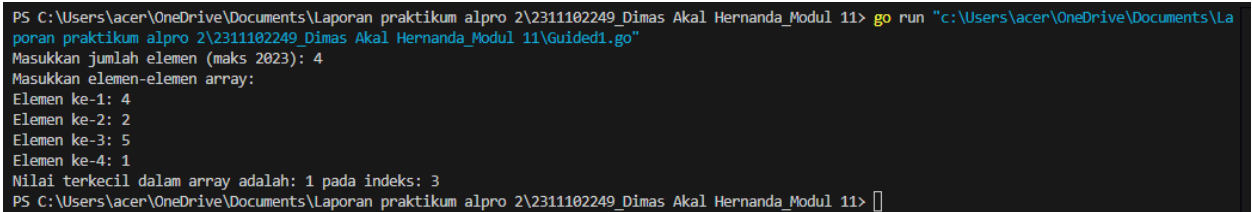
    idxMin := terkecil(tab, n)
```

```
// Menampilkan nilai dan indeks terkecil

    fmt.Println("Nilai terkecil dalam array adalah:", tab[idxMin], "pada indeks:",
idxMin)

}
```

## Screenshoot Output



```
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> go run "c:\Users\acer\OneDrive\Documents\La
poran praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11\Guided1.go"
Masukkan jumlah elemen (maks 2023): 4
Masukkan elemen-elemen array:
Elemen ke-1: 4
Elemen ke-2: 2
Elemen ke-3: 5
Elemen ke-4: 1
Nilai terkecil dalam array adalah: 1 pada indeks: 3
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> █
```

## Deskripsi Program

**Program ini mencari nilai terkecil dan indeksnya dalam array menggunakan Go. Berikut alurnya:**

### 1. Input:

**Pengguna memasukkan jumlah elemen array (n, maksimal 2023) dan nilai setiap elemen.**

### 2. Fungsi terkecil:

**Mencari indeks nilai terkecil dalam array dengan membandingkan elemen satu per satu.**

### 3. Output:

**Program menampilkan nilai terkecil dan indeksnya (berbasis 0).**

**2. 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.**

#### **Source Code**

```
package main

import "fmt"

// Definisi struct mahasiswa dengan atribut nama, nim, kelas, jurusan, dan ipk
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk          float64
}

// Definisi tipe data array mahasiswa dengan kapasitas maksimal 2023
type arrMhs [2023]mahasiswa

// Fungsi untuk mencari IPK tertinggi dalam array mahasiswa
func ipk(T arrMhs, n int) float64 {
    var tertinggi float64 = T[0].ipk
    var j int = 1
    for j < n {
        if tertinggi < T[j].ipk {
            tertinggi = T[j].ipk
        }
    }
}
```

```

        j = j + 1
    }
    return tertinggi
}

// Fungsi main untuk mengisi data mahasiswa dan mencari IPK tertinggi
func main() {
    var n int
    var dataMhs arrMhs

    // Meminta input jumlah mahasiswa
    fmt.Print("Masukkan jumlah mahasiswa (maks 2023): ")
    fmt.Scan(&n)

    // Validasi jumlah mahasiswa yang dimasukkan
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah mahasiswa harus antara 1 dan 2023.")
        return
    }

    // Mengisi data mahasiswa
    for i := 0; i < n; i++ {
        fmt.Printf("\nMasukkan data mahasiswa ke-%d\n", i+1)
        fmt.Print("Nama: ")
        fmt.Scan(&dataMhs[i].nama)
    }
}

```

```

    fmt.Print("NIM: ")

    fmt.Scan(&dataMhs[i].nim)

    fmt.Print("Kelas: ")

    fmt.Scan(&dataMhs[i].kelas)

    fmt.Print("Jurusan: ")

    fmt.Scan(&dataMhs[i].jurusan)

    fmt.Print("IPK: ")

    fmt.Scan(&dataMhs[i].ipk)

}

// Mencari dan menampilkan IPK tertinggi

tertinggi := ipk(dataMhs, n)

fmt.Printf("\nIPK tertinggi dari %d mahasiswa adalah: %.2f\n", n, tertinggi)

}

```

## Screenshoot Output

```

PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11\Guided2.go"
Masukkan jumlah mahasiswa (maks 2023): 2

Masukkan data mahasiswa ke-1
Nama: Dimas
NIM: 2311102249
Kelas: IF-11-06
Jurusan: INFORMATIKA
IPK: 4.0

Masukkan data mahasiswa ke-2
Nama: Hernanda
NIM: 2311102149
Kelas: IF-11-06
Jurusan: INFORMATIKA
IPK: 4.0

IPK tertinggi dari 2 mahasiswa adalah: 4.00
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> 

```



### Deskripsi Program

Program ini mencari IPK tertinggi dari data mahasiswa yang diinput pengguna.

Alur:

1. Input jumlah mahasiswa (n) dan data mahasiswa (nama, NIM, kelas, jurusan, IPK).
2. Fungsi ipk mencari IPK tertinggi dengan membandingkan nilai dari setiap mahasiswa.
3. Output: Menampilkan IPK tertinggi.

Contoh: Input: Data 3 mahasiswa dengan IPK 3.5, 3.8, 3.7.

Output: IPK tertinggi: 3.80.

### III. Unguided

1. Sebuah program digunakan untuk mendaftarkan berat anak kelinci yang akan dijual ke pasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat anak belinci yang akan dijual.

#### Source Code

```
package main

import (
    "fmt"
    "math"
)

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

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

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

```
fmt.Println("Masukkan berat masing-masing anak kelinci:")
```

```
minWeight := math.MaxFloat64
```

```
maxWeight := -math.MaxFloat64
```

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

```
    fmt.Printf("Berat kelinci ke-%d: ", i+1)
```

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

```
    if weights[i] < minWeight {
```

```
        minWeight = weights[i]
```

```
    }
```

```
    if weights[i] > maxWeight {
```

```
        maxWeight = weights[i]
```

```
    }
```

```
}
```

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

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

```
}
```

## Screenshoot Output

```
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11\Unguided1.go"
Masukkan jumlah anak kelinci: 5
Masukkan berat masing-masing anak kelinci:
Berat kelinci ke-1: 3
Berat kelinci ke-2: 5
Berat kelinci ke-3: 4
Berat kelinci ke-4: 6
Berat kelinci ke-5: 2
Berat terkecil: 2.00
Berat terbesar: 6.00
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> █
```

## Deskripsi Program

Program di atas adalah program Go yang meminta pengguna untuk memasukkan jumlah anak kelinci (maksimal 1000), lalu menerima input berat masing-masing kelinci. Program ini kemudian menghitung dan menampilkan berat terkecil dan terbesar di antara semua kelinci yang dimasukkan.

2. Sebuah program digunakan untuk menentukan tarif ikan yang akan dijual be pasar. Program Ini menggunakan array dengan kapasitas 1000 untuk menampung data berat ikan yang akan dijual.

## Source Code

```
package main

import (
    "fmt"
)

func main() {
    var x, y int

    fmt.Println("Masukkan jumlah ikan (x) dan jumlah wadah (y): ")

    fmt.Scan(&x, &y)
```

```
if x <= 0 || x > 1000 || y <= 0 || y > 1000 {  
    fmt.Println("Jumlah ikan dan wadah harus antara 1 hingga 1000.")  
    return  
}
```

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

```
capacities := make([]int, y)  
fmt.Println("Masukkan kapasitas masing-masing wadah:")  
for i := 0; i < y; i++ {  
    fmt.Printf("Kapasitas wadah ke-%d: ", i+1)  
    fmt.Scan(&capacities[i])  
}
```

```
totalWeights := make([]float64, y)  
index := 0
```

```
for i := 0; i < y; i++ {  
    for j := 0; j < capacities[i]; j++ {
```

```
        if index >= x {
            break
        }

        totalWeights[i] += weights[index]

        index++
    }
}

var sumWeights float64

for _, weight := range totalWeights {
    sumWeights += weight
}

fmt.Println("Total berat di setiap wadah:")

for i, weight := range totalWeights {
    fmt.Printf("Wadah ke-%d: %.2f\n", i+1, weight)
}

averageWeight := sumWeights / float64(y)

fmt.Printf("Berat rata-rata di setiap wadah: %.2f\n", averageWeight)
}
```

**Screenshoot Output**

```

PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11\Unguided2.go"
Masukkan jumlah ikan (x) dan jumlah wadah (y): 7 3
Masukkan berat masing-masing ikan:
Berat ikan ke-1: 1.2
Berat ikan ke-2: 2.5
Berat ikan ke-3: 1.8
Berat ikan ke-4: 3.0
Berat ikan ke-5: 0.9
Berat ikan ke-6: 1.5
Berat ikan ke-7: 2.0
Masukkan kapasitas masing-masing wadah:
Kapasitas wadah ke-1: 3
Kapasitas wadah ke-2: 2
Kapasitas wadah ke-3: 2
Total berat di setiap wadah:
Wadah ke-1: 5.50
Wadah ke-2: 3.90
Wadah ke-3: 3.50
Berat rata-rata di setiap wadah: 4.30
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> 

```

## Deskripsi Program

Program di atas adalah program Go yang menghitung distribusi berat ikan ke sejumlah wadah berdasarkan kapasitas masing-masing wadah. Program meminta input jumlah ikan dan wadah, berat tiap ikan, serta kapasitas tiap wadah. Program kemudian mendistribusikan ikan ke wadah secara berurutan, menghitung total berat di setiap wadah, dan menampilkan berat rata-rata semua wadah.

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.

## Source Code

```

package main

import "fmt"

func main() {

    var jumlahBalita int

    var min, max, total float64

    fmt.Print("Masukkan jumlah balita: ")

```

```
fmt.Scan(&jumlahBalita)
```

```
if jumlahBalita <= 0 {
```

```
    fmt.Println("Jumlah balita harus lebih dari 0.")
```

```
    return
```

```
}
```

```
dataBalita := make([]float64, jumlahBalita)
```

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

```
    fmt.Printf("Masukkan berat balita ke-%d: ", i+1)
```

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

```
}
```

```
min = dataBalita[0]
```

```
max = dataBalita[0]
```

```
for _, berat := range dataBalita {
```

```
    if berat < min {
```

```
        min = berat
```

```
    }
```

```
    if berat > max {
```

```
        max = berat
```

```
    }
```

```
total += berat
```

```

    }

    rataRata := total / float64(jumlahBalita)

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

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

    fmt.Printf("Rata-rata berat: %.2f kg\n", rataRata)
}

```

## Screenshoot Output

```

PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11\Unguided3.go"
Masukkan jumlah balita: 4
Masukkan berat balita ke-1: 6.2
Masukkan berat balita ke-2: 7.2
Masukkan berat balita ke-3: 6.5
Masukkan berat balita ke-4: 8.2

Berat minimum: 6.20 kg
Berat maksimum: 8.20 kg
Rata-rata berat: 7.02 kg
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 11> 

```

## Deskripsi Program

Program di atas meminta pengguna untuk memasukkan jumlah balita dan berat masing-masing balita. Program kemudian menghitung dan menampilkan berat terkecil (minimum), terbesar (maksimum), serta rata-rata berat dari balita yang dimasukkan. Program menggunakan slice untuk menyimpan data berat balita dan melakukan perhitungan dengan iterasi.