

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL VIII

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Disusun Oleh :

ALTHAF TEGAR SOFYAN / 2311102217

Kelas IF 11 06

Dosen Pengampu :

Abednego Dwi Septiadi, S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

10.1 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	max ← 1	max = 0
2	i ← 2	i =
3	while i ≤ n do	for i < n {
4	if a[i] > a[max] then	if a[i] > a[max] {
5	max ← i	max = i
6	endif	}
7	i ← i + 1	i = i +
8	endwhile	}

II. GUIDED

1. Tipe Dasar IDX

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)

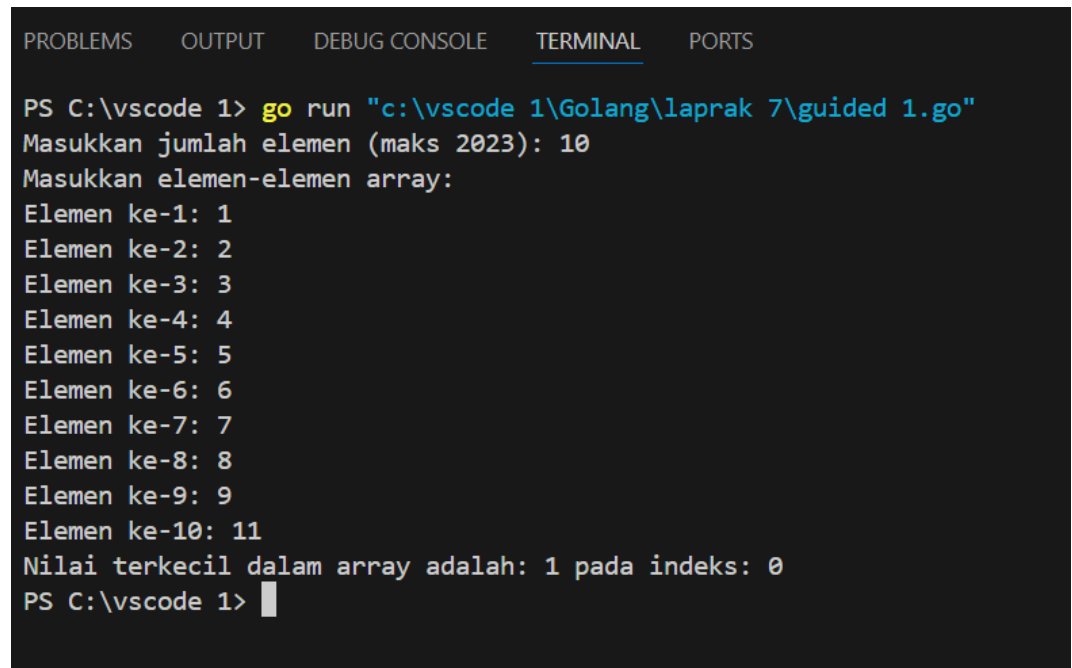
    // Validasi input jumlah elemen
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah elemen harus antara 1 dan 2023.")
        return
    }

    // 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



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\vscode 1> go run "c:\vscode 1\Golang\laprak 7\guided 1.go"
Masukkan jumlah elemen (maks 2023): 10
Masukkan elemen-elemen array:
Elemen ke-1: 1
Elemen ke-2: 2
Elemen ke-3: 3
Elemen ke-4: 4
Elemen ke-5: 5
Elemen ke-6: 6
Elemen ke-7: 7
Elemen ke-8: 8
Elemen ke-9: 9
Elemen ke-10: 11
Nilai terkecil dalam array adalah: 1 pada indeks: 0
PS C:\vscode 1> 
```

Deskripsi Program: Program diatas digunakan untuk menemukan elemen terkecil dalam array. Program akan mendeklarasikan tipe data array dengan Panjang maksimum 2023. Kemudian akan menerima array dan jumlah elemen untuk mengembalikan indeks elemen terkecil dalam array. Fungsi main akan memvalidasi dari jumlah elemen yang dimasukkan oleh pengguna, mengumpulkan elemen, memanggil fungsi terkecil untuk menemukan indeks elemen terkecil, dan terakhir akan menampilkan nilai terkecil beserta indeks nilai.

2. Tipe struktur IPK

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

```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\vscode 1> go run "c:\vscode 1\Golang\laprak 7\guided 2.go"
Masukkan jumlah mahasiswa (maks 2023): 4

Masukkan data mahasiswa ke-1
Nama: RENA
NIM: 23111022188
Kelas: IF
Jurusan: INFORMATIKA
IPK: 3.8

Masukkan data mahasiswa ke-2
Nama: ALTAPP
NIM: 2311102217
Kelas: IF
Jurusan: INFORMATIKA
IPK: 3.7

Masukkan data mahasiswa ke-3
Nama: NISSA
NIM: 2311102218
Kelas: IF
Jurusan: INFORMATIKA
IPK: 3.9

Masukkan data mahasiswa ke-4
Nama: ANDI
NIM: 23111022189
Kelas: IF
Jurusan: INFORMATIKA
IPK: 3.6

IPK tertinggi dari 4 mahasiswa adalah: 3.90
PS C:\vscode 1>

```

Deskripsi Program: Program diatas berfungsi untuk mengelola data mahasiswa. Program ini mendefinisikan struktur mahasiswa yang berisi atribut nama, nim, kelas, jurusan, dan ipk. Fungsi akan menerima array mahasiswa dan jumlah mahasiswa yang terdaftar kemudian mencari dan mengembalikan nilai IPK tertinggi diantara mahasiswa. Fungsi main akan melakukan validasi dari jumlah mahasiswa yang dimasukkan dan akan mengumpulkan data mahasiswa satu persatu. Kemudian akan mencari nilai.

III. UNGUIDED

1. Buatlah sebuah program yang digunakan untuk mendata berat anak kelinci yang akan dijual kepasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat anak kelinci yang akan dijual. Masukkan 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.

Source Code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    const maxKapasitas = 1000
    var berat2217 [maxKapasitas]float64
    var n217 int

    fmt.Print("Masukkan jumlah anak kelinci: ")
    _, err := fmt.Scan(&n217)
    if err != nil || n217 < 1 || n217 > maxKapasitas {
        fmt.Println("Input tidak valid. Pastikan jumlah anak
kelinci antara 1 hingga 1000.")
        return
    }

    fmt.Print("Masukkan berat anak kelinci:")
    for i := 0; i < n217; i++ {
        _, err := fmt.Scan(&berat2217[i])
        if err != nil {
            fmt.Println("Input tidak valid. Pastikan memasukkan
bilangan riil untuk berat.")
            return
        }
    }

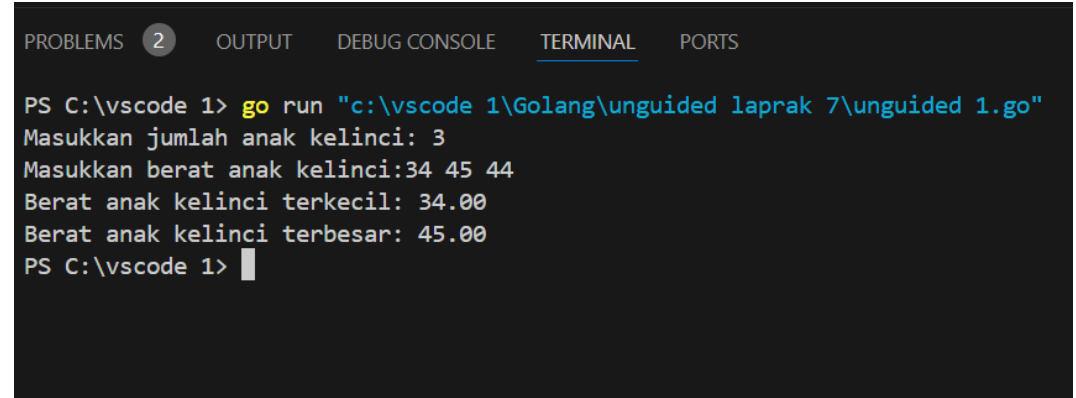
    // Inisialisasi nilai terendah dan tertinggi
    minWeight := math.Inf(1)
    maxWeight := math.Inf(-1)

    // Mencari berat terkecil dan terbesar
    for i := 0; i < n217; i++ {
        if berat2217[i] < minWeight {
            minWeight = berat2217[i]
        }
        if berat2217[i] > maxWeight {
            maxWeight = berat2217[i]
        }
    }

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

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

Screenshoot Output



```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS  
  
PS C:\vscode 1> go run "c:\vscode 1\Golang\unguided laprak 7\unguided 1.go"  
Masukkan jumlah anak kelinci: 3  
Masukkan berat anak kelinci:34 45 44  
Berat anak kelinci terkecil: 34.00  
Berat anak kelinci terbesar: 45.00  
PS C:\vscode 1> █
```

Deskripsi Program: Program diatas menggunakan array dengan kapasitas maks yaitu 1000 untuk menyimpan berat anak kelinci. Pertama-tama program meminta pengguna untuk memasukkan jumlah anak kelinci yang akan ditimbang berat badannya. Kemudian, program akan meminta memasukkan berat tiap anak kelinci. Program menggunakan loop untuk membandingkan berat setiap kelinci untuk mencari nilai terbesar dan terkecil dari berat kelinci tersebut.

2. Buatlah sebuah program yang 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. Masukkan 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 dimasukkan ke dalam wadah. Baris ke dua 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 disetiap wadah. Jumlah wadah tergantung pada nilai x dan y, urutkan ikan yang dimasukkan ke dalam wadah sesuai urutan pada masukkan baris ke 2. Baris ke dua adalah sebuah bilangan riil yang akan menyatakan rata-rata ikan disetiap wadah.

Source Code

```
package main
package main

import "fmt"

func main() {
    var x22, y17 int
    fmt.Print("Masukkan jumlah ikan dan kapasitas wadah: ")
    fmt.Scan(&x22, &y17)

    berat_ikan := make([]float64, x22)
    fmt.Println("Masukkan berat tiap tiap ikan: ")
    for i := 0; i < x22; i++ {
        fmt.Scan(&berat_ikan[i])
    }

    jumlahWadah := x22 + y17 - 1/y17
    totalBeratWadah := make([]float64, jumlahWadah)

    for i := 0; i < x22; i++ {
        Wadah := i / y17
        totalBeratWadah[Wadah] += berat_ikan[i]
    }

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

    fmt.Print("Rata-rata berat tiap wadah: ")
    for _, total := range totalBeratWadah {
        rataRata := total / float64(y17)
        fmt.Printf("%.2f", rataRata)
    }
    fmt.Println()
}
```

Screenshoot Output

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\vscode 1> go run "c:\vscode 1\Golang\unguided laprak 7\unguided 2\unguided 2.go"
Masukkan jumlah ikan dan kapasitas wadah: 2
2
Masukkan berat tiap tiap ikan:
10
20
Berat total wadah: 30.0000000.0000000.0000000.0000000
Rata-rata berat tiap wadah: 15.0000000.0000000.0000000.0000000
PS C:\vscode 1> 
```

Deskripsi Program: Program tersebut digunakan untuk menghitung total berat ikan yang akan dijual dalam wadah berdasarkan jumlah ikan dan kapasitas wadah. Program menggunakan array untuk menyimpan berat masing masing ikan. Lalu program akan menginisialisasi array totalBeratWadah untuk menyimpan total berat ikan disetiap wadah. Total berat untuk setiap wadah akan dihitung dengan menambahkan berat ikan ynag sesuai berdasarkan indeks wadah. Program akan mencetak total berat ikan dalam wadah dan akan menghitung rata rata berat ikan dalam wadah.

3. Buatlah program dengan permasalahan sebagai berikut ini. Posyandu sebagai tempat pelayanan Kesehatan perlu mencatat data berat balita. Petugas akan memasukkan data tersebut kedalam array. Dari data yang diperoleh akan dicari berat balita terkecil, terbesar, dan reratanya.

Source Code

```
package main
package main

import "fmt"

func main() {
    var n217 int
    fmt.Print("Masukkan jumlah balita: ")
    fmt.Scan(&n217)

    if n217 <= 0 {
        return
    }

    beratBalita := make([]float64, n217)
    for i := 0; i < n217; i++ {
        fmt.Printf("Masukkan berat balita ke-%d (dalam
kg): ", i+1)
        fmt.Scan(&beratBalita[i])
    }

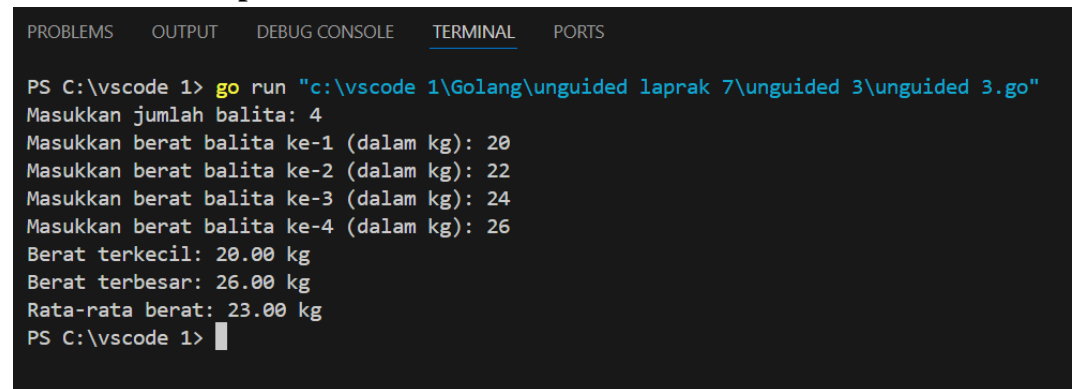
    beratTerkecil := beratBalita[0]
    beratTerbesar := beratBalita[0]
    totalBerat := 0.0

    for _, berat := range beratBalita {
        if berat < beratTerkecil {
            beratTerkecil = berat
        }
        if berat > beratTerbesar {
            beratTerbesar = berat
        }
        totalBerat += berat
    }

    rataRata := totalBerat / float64(n217)

    fmt.Printf("Berat terkecil: %.2f kg\n",
beratTerkecil)
    fmt.Printf("Berat terbesar: %.2f kg\n",
beratTerbesar)
    fmt.Printf("Rata-rata berat: %.2f kg\n", rataRata)
}
```

Screenshoot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\vscode 1> go run "c:\vscode 1\Golang\unguided laprak 7\unguided 3\unguided 3.go"
Masukkan jumlah balita: 4
Masukkan berat balita ke-1 (dalam kg): 20
Masukkan berat balita ke-2 (dalam kg): 22
Masukkan berat balita ke-3 (dalam kg): 24
Masukkan berat balita ke-4 (dalam kg): 26
Berat terkecil: 20.00 kg
Berat terbesar: 26.00 kg
Rata-rata berat: 23.00 kg
PS C:\vscode 1> 
```

Deskripsi Program: Program ini digunakan untuk mengetahui berat balita mulai dari yang terkecil, terberat, dan rata rata. Program meminta pengguna untuk memasukkan jumlah balita dan berat badan balita, untuk berat badan balita akan disimpan melalui array. Lalu, menggunakan loop untuk berat masing masing balita. Program akan menginisialisasi berat terkecil dan terbesar dengan nilai pertama. Kemudian, program melakukan iterasi untuk mencari berat nilai terkecil, terbesar, menghitung total berat, dan rata rata berat balita.