

LAPORAN
PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL7
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Oleh:

NAMA: NUFAIL ALAUDDIN TSAQIF

NIM: 103112400084

KELAS: IF-12-01

S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

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.

2. Pencarian Nilai Ekstrem di Array Bertipe Data Dasar

Pencarian nilai ekstrim pada array bertipe data dasar adalah proses untuk menemukan nilai maksimum atau minimum dalam sebuah kumpulan data. Algoritma ini dimulai dengan menganggap elemen pertama sebagai nilai ekstrim sementara, lalu membandingkan setiap elemen berikutnya dengan nilai tersebut. Jika ditemukan nilai yang lebih besar (untuk maksimum) atau lebih kecil (untuk minimum), nilai ekstrim akan diperbarui. Proses ini dilakukan secara sekuensial hingga seluruh elemen array diperiksa, dan nilai ekstrim yang ditemukan adalah hasil akhir dari pencarian. Dengan kompleksitas waktu $O(n)$, algoritma ini efisien dan sering digunakan dalam berbagai aplikasi seperti analisis data dan pengolahan informasi numerik.

3. Pencarian Nilai Ekstrem di Array Bertipe Data Terstruktur

Pencarian nilai ekstrim pada array bertipe data terstruktur digunakan untuk menemukan nilai tertinggi atau terendah dari atribut tertentu dalam objek atau record. Contohnya, dalam mencari mahasiswa dengan IPK tertinggi, kita akan membandingkan nilai IPK setiap elemen dalam array.

GUIDED 1

SOURCE CODE:

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("masukan jumlah tanaman: ")
    fmt.Scan(&n)

    var tinggitanaman [500]float64
    for i := 0; i < n; i++ {
        fmt.Printf("masukan tinggi tanaman ke -%d(cm) : ", i+1)
        fmt.Scan(&tinggitanaman[i])
    }

    min, max := tinggitanaman[0], tinggitanaman[0]
    for i := 1; i < n; i++ {
        if tinggitanaman[i] < min {
            min = tinggitanaman[i]
        }
        if tinggitanaman[i] > max {
            max = tinggitanaman[i]
        }
    }
    fmt.Printf("\nTinggi Tanaman Tertinggi: %.2f cm/n", max)
    fmt.Printf("Tinggi tanaman teerpendek: %.2f cm/n", min)
}
```

OUTPUT:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALGORITMA
masukan jumlah tanaman: 3
masukan tinggi tanaman ke -1(cm) : 4
masukan tinggi tanaman ke -2(cm) : 5
masukan tinggi tanaman ke -3(cm) : 6

Tinggi Tanaman Tertinggi: 6.00 cm/nTinggi tanaman teerpendek: 4.00 cm/n
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> █
```

DEKSRIPSI:

Program ini meminta pengguna memasukkan jumlah tanaman dan tinggi masing-masing tanaman dalam cm. Kemudian, program menghitung tinggi tanaman tertinggi dan terendah dengan membandingkan nilai-nilai yang dimasukkan, dan menampilkan hasilnya dalam format dua angka desimal. Data disimpan dalam array dengan kapasitas maksimum 500, dan pencarian nilai minimum serta maksimum dilakukan menggunakan perulangan.

GUIDED 2

SOURCE CODE:

```
package main

import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukan jumlah buku dan jumlah buku per rak: ")
    fmt.Scan(&x, &y)

    var hargaBuku [500]float64
    fmt.Println("\nMasukan harga setiap buku (dalam ribuan Rp): ")
    for i := 0; i < x; i++ {
        fmt.Scan(&hargaBuku[i])
    }

    var hargaRataRata []float64
    for i := 0; i < x; i += y {
        total := 0.0
        for j := i; j < i+y && j < x; j++ {
            total += hargaBuku[j]
        }
        hargaRataRata = append(hargaRataRata, total/float64(y))
    }

    min, max := hargaBuku[0], hargaBuku[0]
    for _, harga := range hargaBuku[:x] {
        if harga < min {
            min = harga
        }
        if harga > max {
            max = harga
        }
    }

    fmt.Printf("\nRata-rata harga per rak: ")
    for _, avg := range hargaRataRata {
        fmt.Printf("%.2f ", avg)
    }
    fmt.Printf("\nHarga termahal: %.2f Rp\n", max)
    fmt.Printf("Harga termurah: %.2f Rp\n", min)
}
```

OUTPUT:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALC
Masukan jumlah buku dan jumlah buku per rak: 3 3

Masukan harga setiap buku (dalam ribuan Rp):
20000
30000
40000

Rata-rata harga per rak: 30000.00
Harga termahal: 40000.00 Rp
Harga termurah: 20000.00 Rp
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> |
```

DEKSRIPSI:

Program ini meminta pengguna memasukkan jumlah buku dan jumlah buku per rak, kemudian input harga setiap buku dalam ribuan rupiah. Program menghitung rata-rata harga buku per rak, harga termahal, dan harga termurah dari seluruh daftar harga yang dimasukkan. Hasilnya ditampilkan dalam format dua angka desimal.

GUIDED 3

SOURCE CODE:

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukan jumlah siswa: ")
    fmt.Scan(&n)

    var nilaiSiswa [200]float64
    var totalNilai float64 = 0

    fmt.Println("\nMasukan nilai ujian masing-masing siswa:")
    for i := 0; i < n; i++ {
        fmt.Scan(&nilaiSiswa[i])
        totalNilai += nilaiSiswa[i]
    }

    rataRata := totalNilai / float64(n)

    min, max := nilaiSiswa[0], nilaiSiswa[0]
    var diAtasRataRata int = 0
    for _, nilai := range nilaiSiswa[:n] {
        if nilai < min {
            min = nilai
        }
        if nilai > max {
            max = nilai
        }
        if nilai > rataRata {
            diAtasRataRata++
        }
    }

    fmt.Printf("\nNilai terendah: %.0f\n", min)
    fmt.Printf("Nilai terendah: %.0f\n", max)
    fmt.Printf("Rata-rata kelas: %.2f\n", rataRata)
    fmt.Printf("Jumlah siswa di atas rata-rata: %d\n", diAtasRataRata)
}
```

OUTPUT:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALGORITMA
Masukan jumlah siswa: 3

Masukan nilai ujian masing-masing siswa:
80
88
78

Nilai terendah: 78
Nilai terendah: 88
Rata-rata kelas: 82.00
Jumlah siswa di atas rata-rata: 1
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> |
```

DEKSRIPSI:

Program ini meminta pengguna untuk memasukkan jumlah siswa dan nilai ujian masing-masing siswa. Program kemudian menghitung total nilai, rata-rata kelas, serta nilai terendah dan tertinggi. Selain itu, program menghitung jumlah siswa yang nilai ujian mereka berada di atas rata-rata kelas. Hasilnya ditampilkan dalam format yang sesuai, termasuk nilai terendah, nilai tertinggi, rata-rata kelas, dan jumlah siswa di atas rata-rata.

II. UNGUIDED

UNGUIDED 1

SOURCE CODE

```
package main
// 103112400084 NUFAIL ALAUDDIN TSAQIF
import "fmt"

func main() {
    var jumlahKelinci int
    var beratKelinci [1000]float64

    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&jumlahKelinci)

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

    min, max := beratKelinci[0], beratKelinci[0]

    for i := 1; i < jumlahKelinci; i++ {
        if beratKelinci[i] < min {
            min = beratKelinci[i]
        }
        if beratKelinci[i] > max {
            max = beratKelinci[i]
        }
    }

    fmt.Printf("Berat kelinci terkecil: %.2f kg\n", min)
    fmt.Printf("Berat kelinci terbesar: %.2f kg\n", max)
}
```

OUTPUT

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALGO
Masukkan jumlah anak kelinci: 3
Masukkan berat kelinci ke-1 (kg): 1
Masukkan berat kelinci ke-2 (kg): 2
Masukkan berat kelinci ke-3 (kg): 3
Berat kelinci terkecil: 1.00 kg
Berat kelinci terbesar: 3.00 kg
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> █
```

DEKSRIPSI

Program ini meminta pengguna memasukkan jumlah anak kelinci dan berat masing-masing kelinci dalam kilogram. Program menyimpan data berat kelinci dalam array dan membandingkan setiap nilai untuk menemukan berat kelinci terkecil dan terbesar. Hasilnya akan ditampilkan dalam format dua angka desimal, efektif untuk memantau rentang berat anak kelinci yang dimasukkan.

\

UNGUIDED 2

SOURCE CODE

```
package main
// 103112400084 NUFAIL ALAUDDIN TSAQIF
import "fmt"

func main() {
    var totalIkan, ikanPerWadah int
    var beratIkan [1000]float64

    fmt.Print("Masukkan jumlah ikan (x) dan jumlah ikan per wadah (y): ")
    fmt.Scan(&totalIkan, &ikanPerWadah)

    fmt.Print("Masukkan berat setiap ikan: ")
    for i := 0; i < totalIkan; i++ {
        fmt.Scan(&beratIkan[i])
    }

    jumlahWadah := (totalIkan + ikanPerWadah - 1) / ikanPerWadah

    totalBeratPerWadah := make([]float64, jumlahWadah)
    jumlahIkanPerWadah := make([]int, jumlahWadah)

    for i := 0; i < totalIkan; i++ {
        wadahIdx := i / ikanPerWadah
        totalBeratPerWadah[wadahIdx] += beratIkan[i]
        jumlahIkanPerWadah[wadahIdx]++
    }

    fmt.Println("\nTotal berat tiap wadah:")
    for i, berat := range totalBeratPerWadah {
        fmt.Printf("| Wadah ke-%d: %.2f kg |", i+1, berat)
    }
    fmt.Println()

    fmt.Println("\nRata-rata berat tiap wadah:")
    for i := 0; i < jumlahWadah; i++ {
        if jumlahIkanPerWadah[i] > 0 {
            rataRata := totalBeratPerWadah[i] / float64(jumlahIkanPerWadah[i])
            fmt.Printf("| Berat rata-rata wadah ke-%d: %.2f kg |", i+1, rataRata)
        }
    }
    fmt.Println()
}
```

OUTPUT

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7\main.go"
Masukkan jumlah ikan (x) dan jumlah ikan per wadah (y): 4 4
Masukkan berat setiap ikan: 4
5
6
7

Total berat tiap wadah:
| Wadah ke-1: 22.00 kg |

Rata-rata berat tiap wadah:
| Berat rata-rata wadah ke-1: 5.50 kg |
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> |
```

DEKSRIPSI

Program ini meminta pengguna untuk memasukkan jumlah ikan dan jumlah ikan per wadah, kemudian memasukkan berat setiap ikan. Program menghitung jumlah wadah yang diperlukan dan total berat ikan dalam setiap wadah. Selain itu, program juga menghitung berat rata-rata ikan per wadah dan menampilkan hasilnya. Setiap wadah akan ditampilkan dengan total berat ikan, dan jika terdapat lebih dari satu ikan di wadah tersebut, berat rata-ratanya juga akan ditampilkan dalam format dua angka desimal.

UNGUIDED 3

SOURCE CODE

```
package main
// 103112400084 NUFAIL ALAUDDIN TSAQIF
import "fmt"

func hitungMinMaxRata(arr []float64) (min, max, rata float64) {
    if len(arr) == 0 {
        return 0, 0, 0
    }

    min, max = arr[0], arr[0]
    var total float64

    for _, v := range arr {
        if v > 0 {
            if v < min {
                min = v
            }
            if v > max {
                max = v
            }
            total += v
        }
    }

    if len(arr) > 0 {
        rata = total / float64(len(arr))
    }
    return
}

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

    if n <= 0 || n > 100 {
        fmt.Println("Jumlah data tidak valid.")
        return
    }

    var dataBalita = make([]float64, n)
```

```

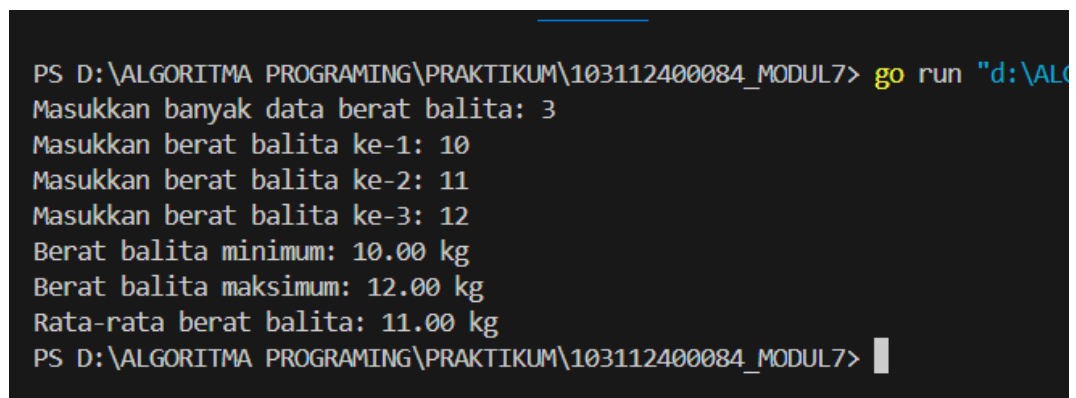
for i := 0; i < n; i++ {
    fmt.Printf("Masukkan berat balita ke-%d: ", i+1)
    fmt.Scan(&dataBalita[i])
}

min, max, rata := hitungMinMaxRata(dataBalita)

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

```

OUTPUT



```

PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7> go run "d:\ALC
Masukkan banyak data berat balita: 3
Masukkan berat balita ke-1: 10
Masukkan berat balita ke-2: 11
Masukkan berat balita ke-3: 12
Berat balita minimum: 10.00 kg
Berat balita maksimum: 12.00 kg
Rata-rata berat balita: 11.00 kg
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL7>

```

DEKSRIPSI

Program ini meminta pengguna untuk memasukkan jumlah data berat balita, kemudian memasukkan berat balita satu per satu. Program menyimpan data tersebut dalam array dan menghitung berat balita minimum, maksimum, serta rata-rata berat balita yang valid (lebih dari 0). Fungsi `hitungMinMaxRata` digunakan untuk menghitung nilai-nilai tersebut. Setelah semua data dimasukkan, program akan menampilkan hasil berat balita minimum, maksimum, dan rata-rata dengan format dua angka desimal, serta melakukan validasi agar jumlah data yang dimasukkan berada dalam rentang yang sesuai (1 hingga 100).

III. KESIMPULAN

Kesimpulan dari laporan ini adalah bahwa program-program yang dibahas berhasil mengimplementasikan algoritma pencarian nilai ekstrim (maksimum dan minimum) pada berbagai jenis data, seperti tanaman, buku, siswa, kelinci, ikan, dan balita. Setiap program menggunakan array untuk menyimpan data dan perulangan untuk membandingkan nilai-nilai dalam array tersebut. Selain itu, beberapa program juga menghitung nilai rata-rata dan melakukan validasi input dari pengguna. Algoritma yang diterapkan memiliki kompleksitas waktu $O(n)$ dan efektif dalam mengolah data bertipe dasar maupun terstruktur, menghasilkan output yang akurat dan relevan dengan konteks penggunaannya.

REFERENSI

MODUL 7 PENCARIAN NILAI EKSTRIM PADA HIMPUNAN
DATA ALGORITMA PEMOGRAMAN 2