

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 11
NILAI EKSTREM**



**Disusun Oleh :
M. Faleno Albar Firjatulloh
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

A. 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 computer, pencarian suatu teks di dalam sebuah dokumen, pencarian buku pada rak buku, dan contoh lainnya.

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 sebagai berikut :

- Jadikan data pertama sebagai nilai ekstrim
- Lakukan validasi nilai ekstrim dari data kedua hingga data terakhir
- Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid.

B. 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.

C. Pencarian Nilai Ekstrim pada Array Bertipe Data Terstruktur

Pada kasus yang lebih kompleks pencarian ekstrim dapat juga dilakukan, misalnya mencari dari mahasiswa dengan nilai terbesar, mencari lagu dengan durasi terlama, mencari pembalap yang memiliki catatan waktu balap tercepat dan sebagainya.

II. UNGUIDED

1. Unguided 1

Soal Studi Case

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() {
    // Deklarasi array dengan kapasitas 1000
    var beratKelinci [1000]float64

    // Input jumlah kelinci
    var N int
    fmt.Print("Masukkan jumlah anak kelinci yang akan ditimbang: ")
    fmt.Scan(&N)

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

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

```

// Inisialisasi nilai minimum dan maksimum
beratMin := math.MaxFloat64
beratMax := -math.MaxFloat64

// Mencari berat minimum dan maksimum
for i := 0; i < N; i++ {
    // Update berat minimum
    if beratKelinci[i] < beratMin {
        beratMin = beratKelinci[i]
    }

    // Update berat maksimum
    if beratKelinci[i] > beratMax {
        beratMax = beratKelinci[i]
    }
}

// Output hasil
fmt.Printf("\nHasil    analisis    berat
kelinci:\n")
fmt.Printf("Berat    terkecil:    %.2f    kg\n",
beratMin)
fmt.Printf("Berat    terbesar:    %.2f    kg\n",
beratMax)
}

```

Screenshoot Output

```

PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\Pertemuan 8\unguided1.go"
Masukkan jumlah anak kelinci yang akan ditimbang: 4
Masukkan berat masing-masing kelinci:
Berat kelinci ke-1: 2
Berat kelinci ke-2: 1
Berat kelinci ke-3: 3
Berat kelinci ke-4: 2

Hasil analisis berat kelinci:
Berat terkecil: 1.00 kg
Berat terbesar: 3.00 kg
PS D:\smt 3\Praktikum Alpro 2>

```

Deskripsi Program :

Program Go ini berfungsi untuk mencatat dan menganalisis berat kelinci. Pengguna dapat memasukkan data berat untuk maksimal 1000 kelinci. Program akan memvalidasi input jumlah kelinci, mengumpulkan data berat satu per satu, lalu menghitung dan menampilkan berat minimum dan maksimum dari seluruh data yang dimasukkan.

2. Unguided 2

Soal Studi Case

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 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() {
    // Deklarasi array dengan kapasitas 1000
    var beratIkan [1000]float64
    var beratWadah [1000]float64

    // Input jumlah ikan (x)
    var x int
    fmt.Print("Masukkan jumlah ikan (x): ")
    fmt.Scan(&x)

    // Input kapasitas wadah (y)
    var y int
    fmt.Print("Masukkan kapasitas wadah (y): ")
    fmt.Scan(&y)

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

```

// Hitung jumlah wadah yang diperlukan
jumWadah := (x + y - 1) / y

// Hitung total berat per wadah
for i := 0; i < x; i++ {
    wadahKe := i / y
    beratWadah[wadahKe] += beratIkan[i]
}

// Output berat total di setiap wadah
fmt.Println("Berat total di setiap wadah:")
for i := 0; i < jumWadah; i++ {
    fmt.Printf("Wadah %d: %.2f\n", i+1,
beratWadah[i])
}

// Hitung dan output rata-rata
var totalBerat float64
for i := 0; i < jumWadah; i++ {
    totalBerat += beratWadah[i]
}
fmt.Printf("Rata-rata berat ikan per wadah:
%.2f\n", totalBerat/float64(jumWadah))
}

```

Screenshoot Output

```

PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\Pertemuan 8\unguided2.go"
unguided2.go
Masukkan jumlah ikan (x): 4
Masukkan kapasitas wadah (y): 2
Masukkan berat ikan:
Berat ikan ke-1: 2
Berat ikan ke-2: 3
Berat ikan ke-3: 1
Berat ikan ke-4: 2
Berat total di setiap wadah:
Wadah 1: 5.00
Wadah 2: 3.00
Rata-rata berat ikan per wadah: 4.00
PS D:\smt 3\Praktikum Alpro 2>

```

Deskripsi Program :

Program Go ini dibuat untuk mengelola berat ikan dalam wadah. Program menggunakan dua array untuk menyimpan berat individual ikan dan total berat per wadah. User memasukkan jumlah ikan dan kapasitas wadah, lalu mencatat berat setiap ikan. Program menghitung jumlah wadah yang dibutuhkan, mendistribusikan ikan ke wadah sesuai kapasitas, dan menampilkan total berat tiap wadah serta rata-rata berat

per wadah. Sistem ini memudahkan pengelolaan dan analisis distribusi berat ikan.

3. Unguided 3

Soal Studi Case

Pos Pelayanan Terpadu (posyandu) sebagai tempat pelayanan kesehatan perlu mencatat data berat balita (dalam kg). Petugas akan memasukkan data tersebut ke dalam array. Dan 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"
    "math"
)

// Mendefinisikan tipe array untuk berat balita
type arrBalita [100]float64

// Fungsi untuk menghitung berat minimum dan maksimum
```

```

func hitungMinMax(arrBerat arrBalita, bMin, bMax
*float64, n int) {
    *bMin = math.MaxFloat64
    *bMax = -math.MaxFloat64

    for i := 0; i < n; i++ {
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
    }
}

// Fungsi untuk menghitung rerata berat balita
func rerata(arrBerat arrBalita, n int) float64 {
    var total float64 = 0
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}

func main() {
    var beratBalita arrBalita
    var jumlahData int
    var beratMin, beratMax float64

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

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

    // Hitung berat minimum dan maksimum
    hitungMinMax(beratBalita, &beratMin,
&beratMax, jumlahData)

    // Hitung rerata

```



```

        rerataBalita := rerata(beratBalita,
jumlahData)

    // Output hasil
    fmt.Printf("Berat balita minimum: %.2f
kg\n", beratMin)
    fmt.Printf("Berat balita maksimum: %.2f
kg\n", beratMax)
    fmt.Printf("Rerata berat balita: %.2f kg\n",
rerataBalita)
}

```

Screenshot Output

```

PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\Pertemuan 8\unguided3.go"
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
PS D:\smt 3\Praktikum Alpro 2>

```

Deskripsi Program :

Program Go ini dirancang untuk analisis berat balita, dengan kapasitas 100 data. Program meminta user memasukkan jumlah dan berat balita, lalu menggunakan fungsi hitungMinMax dengan pointer untuk mencari berat terendah dan tertinggi. Fungsi rerata digunakan untuk menghitung rata-rata berat. Program menampilkan hasil analisis berupa berat minimum, maksimum, dan rata-rata, sehingga membantu pemantauan pertumbuhan balita secara sistematis.

III. KESIMPULAN

Pencarian nilai ekstrem, seperti nilai maksimum dan minimum dalam sebuah dataset, dapat dilakukan dengan cara sederhana menggunakan array dasar. Namun, untuk meningkatkan efisiensi, kita juga bisa memanfaatkan struktur data khusus. Pemilihan metode yang tepat tergantung pada kebutuhan dan karakteristik dataset yang dianalisis.

Jika dataset kecil dan tidak memerlukan pengolahan kompleks, array dasar mungkin sudah memadai. Namun, untuk dataset yang lebih besar atau ketika performa sangat penting, struktur data canggih seperti pohon biner atau heap dapat memberikan keunggulan signifikan. Struktur data ini tidak hanya mempercepat pencarian nilai ekstrem, tetapi juga memfasilitasi operasi lain, seperti penyisipan dan penghapusan data, dengan lebih efisien.

Oleh karena itu, penting bagi pengembang dan analis data untuk mempertimbangkan ukuran dan sifat dataset, serta tujuan analisis, saat memilih metode pencarian nilai ekstrem. Dengan pendekatan yang tepat, kita dapat mengoptimalkan kinerja aplikasi dan mendapatkan hasil analisis yang lebih cepat dan akurat.

IV. REFERENSI

[1] Modul 11 Praktikum Algoritma 2