

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 11
NILAI EKSTREM**



Disusun Oleh :

Shiva Indah Kurnia

2311102035

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 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"
)

func main() {
    // 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
    }

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

    // Input berat kelinci dan inisialisasi nilai min dan
max
    var beratMin, beratMax float64
    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])
```

```

        if i == 0 {
            beratMin, beratMax = beratKelinci[i],
beratKelinci[i]
        } else {
            if beratKelinci[i] < beratMin {
                beratMin = beratKelinci[i]
            }
            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

```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Alpro sem 2> go run "c:\Alpro sem 2\minggu11\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 C:\Alpro sem 2> 

```

Deskripsi Program :

Program ini, ditulis dalam Go, membantu menganalisis berat kelinci. Pengguna memasukkan jumlah kelinci (1-1000) dan berat masing-masing. Program menghitung berat terkecil dan terbesar dengan membandingkan setiap data yang dimasukkan, lalu menampilkan hasil analisis dalam format yang jelas.

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

    // Validasi input
    if x <= 0 || y <= 0 {
        fmt.Println("Jumlah ikan dan kapasitas wadah harus lebih dari 0")
        return
    }

    // Deklarasi slice untuk berat ikan dan wadah
    beratIkan := make([]float64, x)
    jumWadah := (x + y - 1) / y // Hitung jumlah wadah
    beratWadah := make([]float64, jumWadah)

    // 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])

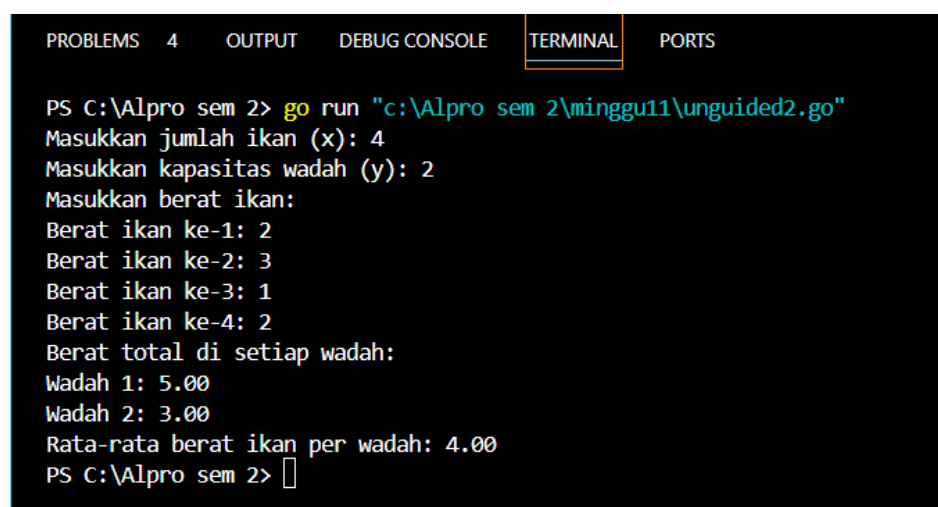
        // Distribusi berat ikan ke wadah
        wadahKe := i / y
        beratWadah[wadahKe] += beratIkan[i]
    }

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

    // Hitung rata-rata berat ikan per wadah
    var totalBerat float64
    for _, berat := range beratWadah {
        totalBerat += berat
    }
    fmt.Printf("Rata-rata berat ikan per wadah: %.2f\n",
totalBerat/float64(jumWadah))
}

```

Screenshoot Output



```

PROBLEMS  4    OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Alpro sem 2> go run "c:\Alpro sem 2\minggu11\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 C:\Alpro sem 2> 

```

Deskripsi Program :

Program ini, ditulis dalam Go, menghitung dan menganalisis distribusi berat ikan ke dalam wadah berdasarkan kapasitas tertentu. Pengguna memasukkan jumlah ikan, kapasitas wadah, dan berat tiap ikan. Program menghitung jumlah wadah yang dibutuhkan, mengelompokkan berat ikan ke wadah sesuai kapasitas, lalu menampilkan total berat per wadah dan rata-rata berat ikan per wadah. Program ini mempermudah analisis data berat ikan secara efisien.

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"
)

func main() {
    // Input jumlah data
    var jumlahData int
    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&jumlahData)

    // Validasi input
    if jumlahData <= 0 || jumlahData > 100 {
        fmt.Println("Jumlah data harus antara 1-100")
        return
    }

    // Deklarasi slice untuk menyimpan berat balita
    beratBalita := make([]float64, jumlahData)

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

    // Inisialisasi nilai min, max, dan total
    beratMin := math.MaxFloat64
    beratMax := -math.MaxFloat64
    var total float64

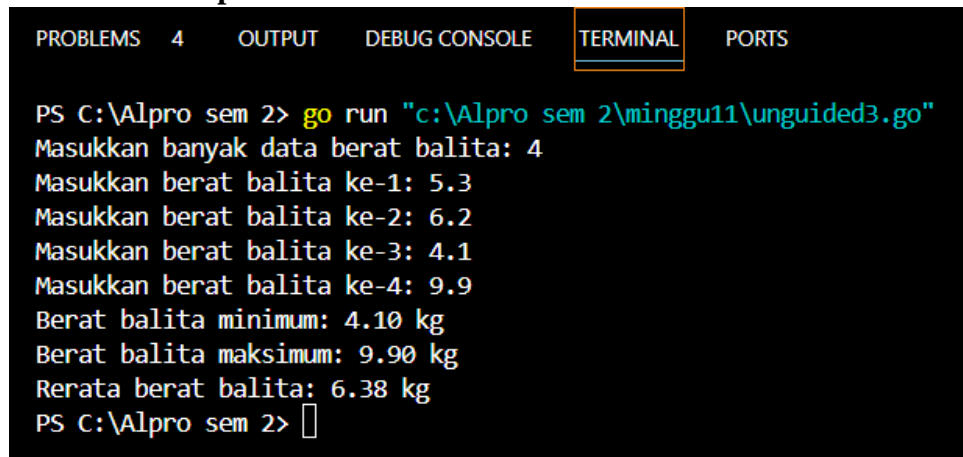
    // Hitung berat minimum, maksimum, dan total
    for _, berat := range beratBalita {
        if berat < beratMin {
            beratMin = berat
        }
        if berat > beratMax {
            beratMax = berat
        }
        total += berat
    }
}
```



```
// Hitung rerata
rerataBalita := total / float64(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

A screenshot of a terminal window showing the execution of a Go program. The window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is active), and PORTS. The terminal shows the command to run the program, followed by user input for the number of data points and individual weights. The program then outputs the minimum, maximum, and average weight.

```
PS C:\Alpro sem 2> go run "c:\Alpro sem 2\minggu11\unguided3.go"
Masukkan banyak data berat balita: 4
Masukkan berat balita ke-1: 5.3
Masukkan berat balita ke-2: 6.2
Masukkan berat balita ke-3: 4.1
Masukkan 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 C:\Alpro sem 2> 
```

Deskripsi Program :

Program Go ini menghitung dan menganalisis berat balita hingga 100 data. Pengguna memasukkan jumlah data dan berat masing-masing balita. Program menghitung berat minimum, maksimum, dan rata-rata dengan memanfaatkan fungsi **hitungMinMax** dan **rerata**. Hasil analisis berupa berat minimum, maksimum, dan rata-rata ditampilkan untuk membantu memantau kesehatan dan pertumbuhan anak secara efisien.

III. KESIMPULAN

Pencarian nilai maksimum dan minimum dalam dataset dapat dilakukan secara sederhana menggunakan array. Namun, untuk efisiensi yang lebih baik, terutama pada dataset besar, struktur data khusus dapat digunakan. Jika dataset kecil dan sederhana, array sudah cukup. Tetapi pada dataset besar atau saat performa menjadi prioritas, struktur data seperti pohon biner atau heap dapat mempercepat pencarian dan mendukung operasi lain, seperti penyisipan dan penghapusan, dengan lebih efisien.

Pengembang dan analis data perlu mempertimbangkan ukuran, karakteristik dataset, dan tujuan analisis saat memilih metode pencarian nilai ekstrem. Dengan pendekatan yang tepat, performa aplikasi dapat dioptimalkan, dan analisis menjadi lebih cepat serta akurat.

IV. REFERENSI

[1] Modul 11 Praktikum Algoritma 2