

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 11
NILAI EKSTREM**



Disusun Oleh :

Agnes Refilina Fiska / 2311102126

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 proses yang sering dilakukan dalam berbagai aktivitas sehari-hari, seperti mencari file di komputer, menemukan teks dalam dokumen, mencari buku di rak, dan banyak lagi. Algoritma pencarian nilai maksimum atau minimum sangat sederhana. Data diproses secara berurutan, dan nilai maksimum atau minimum yang ditemukan disimpan untuk dibandingkan dengan data berikutnya. Ketika algoritma selesai, nilai yang tersisa adalah nilai ekstrim (maksimum atau minimum) yang dicari.

Secara umum, langkah-langkah algoritma ini adalah:

1. Tetapkan data pertama sebagai nilai ekstrim awal.
2. Bandingkan nilai ekstrim dengan data berikutnya satu per satu hingga data terakhir.
3. Setelah semua data diperiksa, nilai ekstrim yang ditemukan adalah hasil pencarian.

B. Pencarian Nilai Ekstrim pada Array Bertipe Data Dasar

Sebagai contoh, sebuah array integer dengan kapasitas 2023 elemen berisi sejumlah N bilangan bulat. Pencarian nilai terkecil dapat dilakukan dengan memulai dari elemen pertama, kemudian membandingkan setiap elemen secara berurutan. Algoritma ini cocok untuk array dengan data bertipe sederhana seperti integer atau float.

C. Pencarian Nilai Ekstrim pada Array Bertipe Data Terstruktur

Pencarian nilai ekstrim juga dapat diterapkan pada array yang berisi data terstruktur, seperti mencari mahasiswa dengan nilai tertinggi, lagu dengan durasi terpanjang, atau pembalap dengan waktu balapan tercepat. Dalam kasus ini, algoritma tetap sama, tetapi membandingkan atribut tertentu dari setiap elemen data terstruktur.

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() {
    var n int
    var weights = make([]float64, 1000)

    // Input jumlah anak kelinci
    fmt.Print("Masukkan jumlah anak kelinci : ")
    fmt.Scan(&n)

    // Input berat masing-masing anak kelinci
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan berat anak kelinci ke-%d: ", i+1)
```

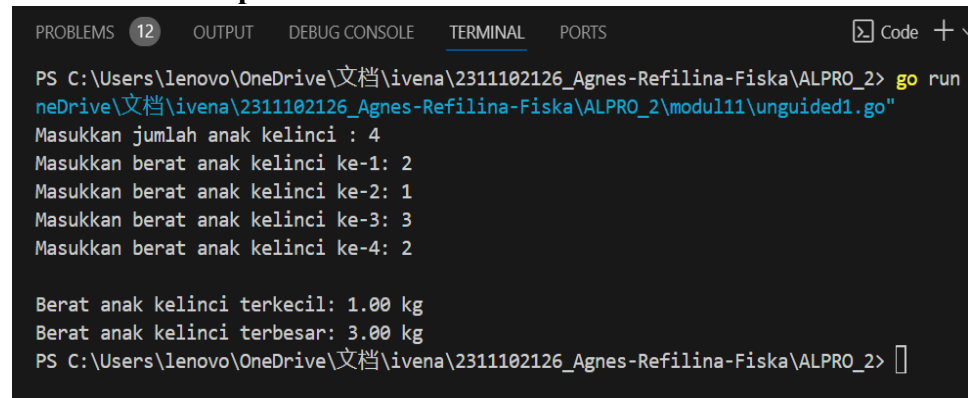
```
        fmt.Scan(&weights[i])
    }

    // Mencari berat terkecil dan terbesar
    var min, max float64 = weights[0], weights[0]
    for i := 1; i < n; i++ {
        if weights[i] < min {
            min = weights[i]
        }
        if weights[i] > max {
            max = weights[i]
        }
    }

    // Output berat terkecil dan terbesar
    fmt.Printf("\nBerat anak kelinci terkecil:
%.2f kg\n", min)

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

Screenshoot Output

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The command 'go run' is executed in the terminal. The program prompts the user to enter the number of children (4), followed by the weight of each child (2, 1, 3, 2 kg). The program then outputs the minimum weight (1.00 kg) and the maximum weight (3.00 kg).

```
PS C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2> go run
neDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2\modul11\unguided1.go"
Masukkan jumlah anak kelinci : 4
Masukkan berat anak kelinci ke-1: 2
Masukkan berat anak kelinci ke-2: 1
Masukkan berat anak kelinci ke-3: 3
Masukkan berat anak kelinci ke-4: 2

Berat anak kelinci terkecil: 1.00 kg
Berat anak kelinci terbesar: 3.00 kg
PS C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2> 
```

Deskripsi Program :

Program dalam bahasa Go (Golang) untuk memproses data berat badan anak kelinci. Program dapat menentukan berat terkecil dan terbesar dari sejumlah data berat yang dimasukkan oleh pengguna. Berat masing-masing kelinci disimpan dalam array dengan kapasitas maksimal 1000 elemen.

Alur Program

1. Inisialisasi dan Deklarasi:

- Variabel `n` digunakan untuk menyimpan jumlah anak kelinci yang datanya akan dimasukkan.
- Array `weights` dengan kapasitas 1000 elemen tipe `float64` digunakan untuk menyimpan data berat badan anak kelinci.

2. Input Data:

- Pengguna diminta memasukkan jumlah anak kelinci (`n`).
- Program meminta berat badan masing-masing kelinci sebanyak jumlah `n` yang telah dimasukkan.

3. Penghitungan Berat Terkecil dan Terbesar:

- Nilai awal untuk berat terkecil (`min`) dan terbesar (`max`) diambil dari berat anak kelinci pertama.
- Program memeriksa setiap elemen dalam array menggunakan perulangan:
 - Jika berat lebih kecil dari `min`, nilai `min` diperbarui.
 - Jika berat lebih besar dari `max`, nilai `max` diperbarui.

4. Output Hasil:

- Program mencetak nilai berat terkecil dan terbesar ke layar dengan format dua angka desimal.

Penjelasan Manfaat

Program ini dapat digunakan untuk menganalisis data berat badan anak kelinci, misalnya untuk keperluan monitoring pertumbuhan atau seleksi berdasarkan berat. Program sederhana ini juga mempermudah pengguna dalam mendapatkan informasi berat terkecil dan terbesar dengan cepat.

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() {
    var x, y int
    var weights = make([]float64,
1000)          // Array untuk berat ikan
    var containers = make([][]float64,
1000) // Array 2D untuk wadah

    // Input jumlah ikan (x)
    fmt.Print("Masukkan jumlah ikan yang
akan dijual (x): ")
    fmt.Scan(&x)

    // Input jumlah wadah (y)
    fmt.Print("Masukkan jumlah wadah yang
tersedia (y): ")
    fmt.Scan(&y)

    // Input berat untuk setiap ikan
    fmt.Println("\nMasukkan berat untuk
setiap ikan (kg):")
    for i := 0; i < x; i++ {
        fmt.Printf("Berat ikan ke-%d: ",
i+1)
        fmt.Scan(&weights[i])
    }

    // Inisialisasi array wadah
    for i := 0; i < y; i++ {
        containers[i] = make([]float64, 0)
    }

    // Distribusi ikan ke wadah
    currentContainer := 0
    for i := 0; i < x; i++ {
```

```

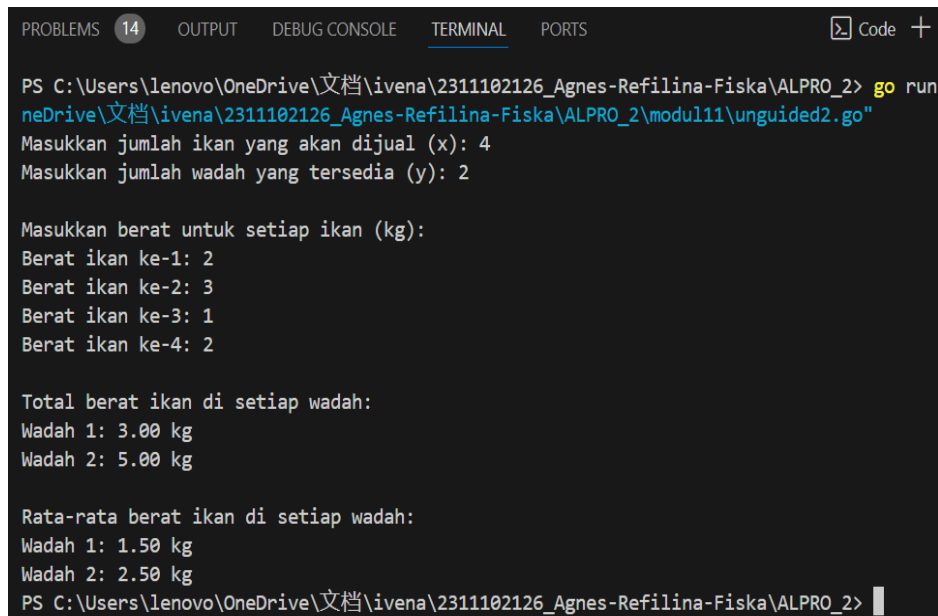
        containers[currentContainer] =
append(containers[currentContainer],
weights[i])
        currentContainer =
(currentContainer + 1) % y
    }

    // Output total berat ikan di setiap
wadah
    fmt.Println("\nTotal berat ikan di
setiap wadah:")
    for i := 0; i < y; i++ {
        var totalWeight float64
        for _, weight := range
containers[i] {
            totalWeight += weight
        }
        fmt.Printf("Wadah %d: %.2f kg\n",
i+1, totalWeight)
    }

    // Hitung dan output rata-rata berat
ikan di setiap wadah
    fmt.Println("\nRata-rata berat ikan di
setiap wadah:")
    for i := 0; i < y; i++ {
        var totalWeight float64
        numFish := len(containers[i])
        for _, weight := range
containers[i] {
            totalWeight += weight
        }
        if numFish > 0 {
            average := totalWeight /
float64(numFish)
            fmt.Printf("Wadah %d: %.2f
kg\n", i+1, average)
        } else {
            fmt.Printf("Wadah %d: 0.00 kg
(kosong)\n", i+1)
        }
    }
}

```


Screenshoot Output

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The command 'go run' is used to execute a file named 'unguided2.go'. The program prompts the user for the number of fish (x) and the number of ponds (y). The user enters 4 for fish and 2 for ponds. Then, the program asks for the weight of each fish in kg, and the user enters 2, 3, 1, and 2 for the four fish. Finally, the program calculates and displays the total weight of fish in each pond (3.00 kg and 5.00 kg) and the average weight of fish in each pond (1.50 kg and 2.50 kg).

```
PS C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2> go run
neDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2\modul11\unguided2.go"
Masukkan jumlah ikan yang akan dijual (x): 4
Masukkan jumlah wadah yang tersedia (y): 2

Masukkan berat untuk setiap ikan (kg):
Berat ikan ke-1: 2
Berat ikan ke-2: 3
Berat ikan ke-3: 1
Berat ikan ke-4: 2

Total berat ikan di setiap wadah:
Wadah 1: 3.00 kg
Wadah 2: 5.00 kg

Rata-rata berat ikan di setiap wadah:
Wadah 1: 1.50 kg
Wadah 2: 2.50 kg
PS C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2>
```

Deskripsi Program :

Program ini dibuat menggunakan bahasa Go (Golang) untuk membantu mengelola distribusi ikan ke beberapa wadah serta menghitung statistik berat ikan. Program ini memakai array untuk menyimpan data berat ikan dan mengelompokkan ikan ke dalam wadah.

Alur Program

1. Input Awal:

- Pengguna diminta memasukkan jumlah ikan (x) dan jumlah wadah (y) yang tersedia.
- Berat masing-masing ikan kemudian dimasukkan satu per satu.

2. Distribusi Ikan ke Wadah:

- Ikan-ikan didistribusikan ke wadah secara bergantian (rotasi). Misalnya, ikan pertama masuk ke wadah pertama, ikan kedua ke wadah kedua, dan seterusnya. Jika sudah mencapai wadah terakhir, pengisian kembali ke wadah pertama.
- Proses ini dilakukan menggunakan operator modulo (%), sehingga distribusi ikan tetap seimbang di semua wadah.

3. Penghitungan Statistik:

- **Total Berat per Wadah:** Berat seluruh ikan dalam setiap wadah dijumlahkan.
- **Rata-Rata Berat per Wadah:** Total berat ikan di wadah dibagi jumlah ikan dalam wadah tersebut.
- Jika ada wadah kosong (tidak ada ikan), program akan menampilkan pesan bahwa wadah tersebut kosong.

4. Output:

- Hasil ditampilkan dengan format rapi, menampilkan total berat dan rata-rata berat per wadah hingga dua angka desimal, lengkap dengan satuan kilogram (kg).

Manfaat Program

Program ini sangat membantu dalam mengatur dan memantau distribusi ikan, terutama di pasar atau tempat penjualan. Dengan pembagian yang merata, setiap wadah terorganisir dengan baik, dan informasi tentang berat ikan di tiap wadah mudah diakses.

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"
)

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

// Fungsi untuk menghitung berat minimum dan maksimum
func hitungMinMax(arrBerat arrBalita, bMin, bMax *float64) {
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]
```

```

// Mencari nilai minimum dan maksimum
for i := 1; i < len(arrBerat); i++ {
    if arrBerat[i] != 0 {
        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
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}

func main() {
    var dataBerat arrBalita
    var jumlahData int
    var min, max float64

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

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

    // Hitung min dan max
    hitungMinMax(dataBerat, &min, &max)
}

```

```

        // Hitung rerata
        rata := rerata(dataBerat, jumlahData)

        // Output hasil
        fmt.Printf("\nBerat balita minimum: %.2f kg\n",
min)
        fmt.Printf("Berat balita maksimum: %.2f kg\n",
max)
        fmt.Printf("Rerata berat balita: %.2f kg\n",
rata)
    }

```

Screenshot Output

```

PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2> go run
-Refilina-Fiska\ALPRO_2\modul11\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 C:\Users\lenovo\OneDrive\文档\ivena\2311102126_Agnes-Refilina-Fiska\ALPRO_2> 

```

Deskripsi Program :

Program ini ditulis dalam bahasa Go (Golang) dan dirancang untuk memproses data berat badan balita. Program dapat menghitung berat minimum, berat maksimum, dan rata-rata berat balita berdasarkan data yang dimasukkan pengguna. Data berat disimpan dalam array, dan hasil perhitungan ditampilkan dengan format rapi.

Alur Program

1. Inisialisasi dan Deklarasi:

- Program mendeklarasikan tipe data `arrBalita` sebagai array dengan kapasitas 100 elemen tipe `float64` untuk menyimpan berat badan balita.

- Variabel lain, seperti jumlah data yang dimasukkan (jumlahData) dan nilai minimum serta maksimum (min dan max), juga diinisialisasi.
2. **Input Data Berat:**
 - Pengguna diminta memasukkan jumlah balita yang datanya akan diproses.
 - Berat badan balita dimasukkan satu per satu hingga sesuai jumlah yang telah ditentukan.
 3. **Penghitungan:**
 - Fungsi hitungMinMax menghitung nilai berat minimum dan maksimum dalam array. Proses ini dilakukan dengan membandingkan setiap elemen dalam array.
 - Fungsi rerata menghitung rata-rata berat badan balita dengan menjumlahkan seluruh data berat dalam array, lalu membaginya dengan jumlah data.
 4. **Output Hasil:**
 - Program mencetak nilai berat minimum, berat maksimum, dan rata-rata berat balita ke layar dengan format dua angka desimal.

III. KESIMPULAN

Pencarian nilai maksimum atau minimum dalam data bisa dilakukan dengan cara sederhana menggunakan array biasa. Metode ini cocok untuk dataset kecil yang tidak membutuhkan proses rumit.

Namun, jika dataset lebih besar atau kecepatan sangat penting, kita bisa menggunakan struktur data khusus seperti pohon biner atau heap. Struktur data ini membuat pencarian, penambahan, dan penghapusan data jadi lebih cepat dan efisien.

Karena itu, penting untuk memilih cara yang sesuai dengan ukuran dan kebutuhan data. Dengan metode yang tepat, kita bisa meningkatkan kinerja aplikasi dan mendapatkan hasil lebih cepat dan akurat.

IV. REFERENSI

[1] Modul 11 Praktikum Algoritma 2

[2]<https://gabrieleromanato.name/go-how-to-find-the-minimum-and-maximum-value-in-a-slice>