

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 11
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Disusun Oleh:

Boutefhika Nuha Ziyadatul Khair/2311102316

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

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	$\text{max} \leftarrow 1$	$\text{max} = 0$
2	$i \leftarrow 2$	$i = 1$
3	while $i \leq n$ do	for $i < n$ {
4	if $a[i] > a[\text{max}]$	if $a[i] > a[\text{max}]$ {
5	then $\text{max} \leftarrow i$	$\text{max} = i$
6	endif	}
7	$i \leftarrow i + 1$	$i = i + 1$
8	endwhile	}

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. Perhatikan potongan program dalam bahasa Go berikut ini!

```
type arrInt [2023]int
...

func terkecil_1(tabInt arrInt, n int) int {
    /* mengembalikan nilai terkecil yang terdapat di dalam tabInt yang berisi n
    bilangan bulat */
    var min int = tabInt[0]           // min berisi data pertama
    var j int = 1                     // pencarian dimulai dari data berikutnya
    for j < n {
        if min > tabInt[j] {          // pengecekan apakah nilai minimum valid
            min = tabInt[j]           // update nilai minimum dengan yang valid
        }
        j = j + 1
    }
    return min                        // returnkan nilai minimumnya
}
```

Potongan program di atas sedikit berbeda dengan sebelumnya karena penggunaan indeks array pada bahasa Go di mulai dari nol atau "0" seperti penjelasan pada modul 9. Selanjutnya, pada penjelasan di awal bab 3 telah disampaikan bahwa pada pencarian yang terpenting adalah posisi atau indeks dari nilai yang dicari dalam kumpulan data atau array. Oleh karena itu modifikasi pada program di atas dapat dilihat pada potongan program berikut ini!

```
...
type arrInt [2023]int
...

func terkecil_2(tabInt arrInt, n int) int {
    /* mengembalikan indeks nilai terkecil yang terdapat di dalam tabInt yang berisi
    n bilangan bulat */
    var idx int = 0                   // idx berisi indeks data pertama
    var j int = 1                     // pencarian dimulai dari data berikutnya
    for j < n {
        if tabInt[idx] > tabInt[j] { // pengecekan apakah nilai minimum valid
            idx = j                   // update nilai minimum dengan yang valid
        }
        j = j + 1
    }
    return idx                        // returnkan indeks nilai minimumnya
}
```

Pencarian Nilai Ekstrim pada Array Bertipe Data Terstruktur

Pada kasus yang lebih kompleks pencarian ekstrim dapat juga dilakukan, misalnya mencari data mahasiswa dengan nilai terbesar, mencari lagu dengan durasi terlama, mencari pembalap yang memiliki catatan waktu balap tercepat, dan sebagainya. Sebagai contoh misalnya terdapat array yang digunakan untuk menyimpan data mahasiswa, kemudian terdapat fungsi IPK yang digunakan untuk mencari data mahasiswa dengan IPK tertinggi.

```
---
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk float64
}
type arrMhs [2023]mahasiswa
---

func IPK_1(T arrMhs, n int) float64 {
    /* mengembalikan ipk terkecil yang dimiliki mahasiswa pada array T yang berisi
    n mahasiswa */
    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
}
```

Apabila diperhatikan potongan program di atas, maka kita akan memperoleh ipk tertinggi, tetapi kita tidak memperoleh identitas mahasiswa dengan ipk tertinggi tersebut. Maka seperti penjelasan yang sudah diberikan sebelumnya, maka pencarian yang dilakukan bisa mengembalikan indeks mahasiswa dengan ipk tertinggi tersebut. Berikut ini adalah modifikasinya! Sehingga melalui algoritma diatas, identitas mahasiswa dapat diperoleh, misalnya

```
---
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk float64
}
type arrMhs [2023]mahasiswa
---

func IPK_2(T arrMhs, n int) int {
    /* mengembalikan indeks mahasiswa yang memiliki ipk tertinggi pada array T yang
    berisi n mahasiswa */
    var idx int = 0
    var j int = 1
    for j < n {
        if T[idx].ipk < T[j].ipk {
            idx = j
        }
        j = j + 1
    }
    return idx
}
```

Sehingga melalui algoritma diatas, identitas mahasiswa dapat diperoleh, misalnya `T[idx].nama`, `T[idx].nim`, `T[idx].kelas`, hingga `T[idx].jurusan`.

II. UNGUIDED

1. 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"
)

type arrBerat [1000]float64

// Fungsi untuk mencari berat terkecil dan terbesar
func cariBeratTerkecilDanTerbesar(berat arrBerat,
n int) (float64, float64) {
    var minBerat, maxBerat float64
    minBerat = berat[0]
    maxBerat = berat[0]

    for i := 1; i < n; i++ {
        if berat[i] < minBerat {
            minBerat = berat[i]
        }
        if berat[i] > maxBerat {
            maxBerat = berat[i]
        }
    }

    return minBerat, maxBerat
}
```

```

func main() {
    var i, n int
    var berat arrBerat

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

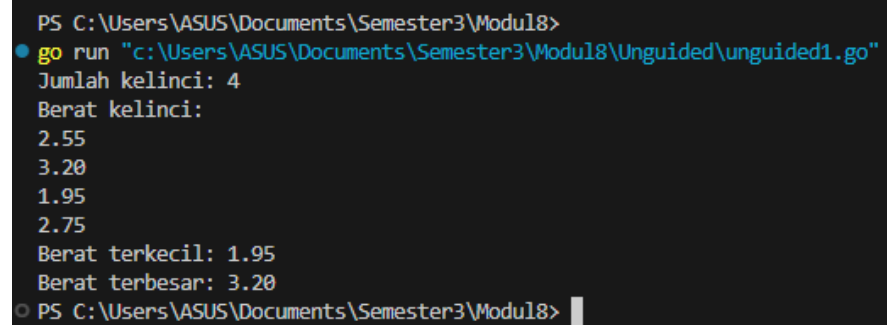
    // Input berat kelinci
    fmt.Println("Berat kelinci:")
    for i = 0; i < n; i++ {
        fmt.Scan(&berat[i])
    }

    // Memanggil fungsi untuk mencari berat terkecil
    dan terbesar
    minBerat, maxBerat :=
    cariBeratTerkecilDanTerbesar(berat, n)

    // Menampilkan hasil
    fmt.Printf("Berat terkecil: %.2f\n", minBerat)
    fmt.Printf("Berat terbesar: %.2f\n", maxBerat)
}

```

Screenshoot Program



```

PS C:\Users\ASUS\Documents\Semester3\Modul8>
go run "c:\Users\ASUS\Documents\Semester3\Modul8\Unguided\unguided1.go"
Jumlah kelinci: 4
Berat kelinci:
2.55
3.20
1.95
2.75
Berat terkecil: 1.95
Berat terbesar: 3.20
PS C:\Users\ASUS\Documents\Semester3\Modul8>

```

Deskripsi Program

Program diatas mencari berat terkecil dan terbesar dari sejumlah kelinci berdasarkan input pengguna. Program menggunakan array arrBerat bertipe float64 untuk menyimpan berat kelinci, dengan ukuran maksimal 1000 elemen. Fungsi cariBeratTerkecilDanTerbesar menerima array berat dan jumlah kelinci sebagai parameter, kemudian mencari nilai berat terkecil dan terbesar dengan iterasi melalui array. Setelah pengguna memasukkan jumlah kelinci dan berat masing-masing kelinci, fungsi ini dipanggil untuk menentukan berat terkecil dan

terbesar. Hasilnya kemudian ditampilkan dengan format dua angka desimal.

2. 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 yang 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"
)

// Definisi tipe array
type arrBerat [1000]float64

// Fungsi untuk menghitung jumlah wadah
func hitungJumlahWadah(x, y int) int {
    if x%y == 0 {
        return x / y
    }
    return x/y + 1
}

// Fungsi untuk menghitung total berat ikan di
// setiap wadah
func hitungTotalBerat(berat arrBerat, totalBerat
*arrBerat, x, y int) {
    for i := 0; i < x; i++ {
        j := i / y // Menentukan indeks wadah
        berdasarkan pembagian
        totalBerat[j] += berat[i]
```

```

    }
}

// Fungsi untuk menghitung rata-rata berat ikan per wadah
func hitungRataRata(totalBerat arrBerat,
jumlahWadah int) float64 {
    var total float64
    for i := 0; i < jumlahWadah; i++ {
        total += totalBerat[i]
    }
    return total / float64(jumlahWadah)
}

func main() {
    var x, y int
    var berat arrBerat
    var totalBerat arrBerat

    // Membaca input banyaknya ikan dan kapasitas wadah
    fmt.Print("Masukkan banyaknya ikan yang akan dijual (x): ")
    fmt.Scan(&x)
    fmt.Print("Masukkan banyaknya ikan yang akan dimasukkan ke dalam setiap wadah (y): ")
    fmt.Scan(&y)

    // Membaca berat masing-masing ikan
    fmt.Print("Masukkan berat ikan (sejumlah x ikan): ")
    for i := 0; i < x; i++ {
        fmt.Scan(&berat[i])
    }

    // Menghitung jumlah wadah
    jumlahWadah := hitungJumlahWadah(x, y)

    // Menghitung total berat ikan di setiap wadah
    hitungTotalBerat(berat, &totalBerat, x, y)

    // Menampilkan total berat ikan per wadah
    fmt.Println("Total berat ikan per wadah: ")
    for i := 0; i < jumlahWadah; i++ {

```



```

        fmt.Printf("Wadah  %d:  %.2f\n",  i+1,
totalBerat[i])
    }

    // Menghitung rata-rata berat ikan per wadah
    rataRata    :=    hitungRataRata(totalBerat,
jumlahWadah)

    // Menampilkan rata-rata berat ikan per wadah
    fmt.Printf("Rata-rata berat ikan per wadah:
%.2f\n", rataRata)
}

```

Screenshoot Program

```

PS C:\Users\ASUS\Documents\Semester3\Modul8>
go run "c:\Users\ASUS\Documents\Semester3\Modul8\Unguided\uno2\unguided2.go"
Masukkan banyaknya ikan yang akan dijual (x): 5
Masukkan banyaknya ikan yang akan dimasukkan ke dalam setiap wadah (y): 2
Masukkan berat ikan (sejumlah x ikan): 1.5 2.3 4.5 3.7 2.8
Total berat ikan per wadah:
Wadah 1: 3.80
Wadah 2: 8.20
Wadah 3: 2.80
Rata-rata berat ikan per wadah: 4.93
PS C:\Users\ASUS\Documents\Semester3\Modul8>

```

Deskripsi Program

Program diatas menghitung total berat ikan per wadah dan rata-rata berat ikan per wadah berdasarkan input jumlah ikan, kapasitas wadah, dan berat masing-masing ikan. Program menggunakan array arrBerat untuk menyimpan berat ikan dan total berat ikan di setiap wadah. Fungsi hitungJumlahWadah menghitung jumlah wadah yang diperlukan berdasarkan jumlah ikan dan kapasitas wadah. Fungsi hitungTotalBerat menghitung total berat ikan di setiap wadah dengan membagi ikan ke dalam wadah berdasarkan kapasitas. Fungsi hitungRataRata menghitung rata-rata berat ikan per wadah dengan menjumlahkan total berat dan membagi dengan jumlah wadah. Setelah input jumlah ikan, kapasitas wadah, dan berat masing-masing ikan dimasukkan, program menampilkan total berat ikan per wadah dan rata-rata berat ikan per wadah.

3. Pos Pelayanan Terpadu (posyandu) sebagai tempat pelayanan kesehatan perlu mencatat data berat balita (dalam kg). Petugas akan memasukkan data tersebut ke dalam array. Dari data yang diperoleh akan dicari berat balita terkecil, terbesar, dan reratanya.

Sourcecode

```
package main

import (
    "fmt"
)

// Tipe data array untuk berat balita
type arrBalita [100]float64

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

    for i := 1; i < n; i++ {
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
    }
}

// Fungsi untuk menghitung rata-rata 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() {
    // Deklarasi variabel
    var n int
    var arrBerat arrBalita
    var bMin, bMax, rataRata float64

    // Membaca banyak data berat balita
    fmt.Print("Masukan banyak data berat balita: ")
}
```

```

    fmt.Scan(&n)

    // Membaca data berat balita
    for i := 0; i < n; i++ {
        fmt.Printf("Masukan berat balita ke-%d: ",
i+1)

        fmt.Scan(&arrBerat[i])
    }

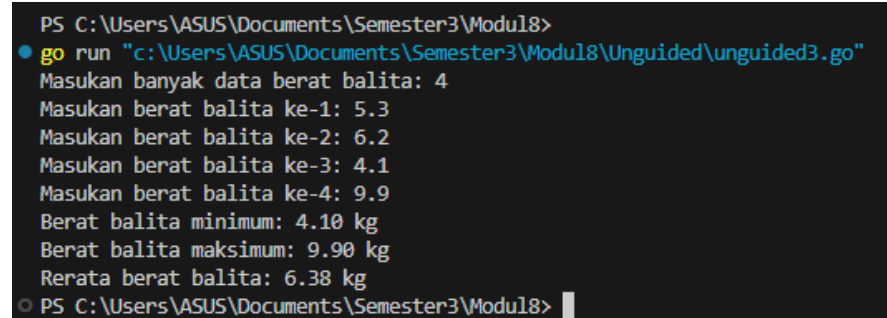
    // Menghitung berat minimum dan maksimum
    hitungMinMax(arrBerat, n, &bMin, &bMax)

    // Menghitung rata-rata berat balita
    rataRata = rerata(arrBerat, n)

    // Menampilkan hasil
    fmt.Printf("Berat balita minimum: %.2f kg\n",
bMin)
    fmt.Printf("Berat balita maksimum: %.2f kg\n",
bMax)
    fmt.Printf("Rerata berat balita: %.2f kg\n",
rataRata)
}

```

Screenshoot Program



```

PS C:\Users\ASUS\Documents\Semester3\Modul8>
go run "c:\Users\ASUS\Documents\Semester3\Modul8\Unguided\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\ASUS\Documents\Semester3\Modul8>

```

Deskripsi Program

Program diatas menghitung berat balita minimum, maksimum, dan rata-rata berdasarkan input data berat balita yang dimasukkan oleh pengguna. Program menggunakan array arrBalita untuk menyimpan berat balita, dengan ukuran maksimal 100 elemen. Fungsi hitungMinMax digunakan untuk mencari berat balita minimum dan maksimum, dimana nilai minimum dan maksimum diperbarui dengan membandingkan setiap elemen dalam array. Fungsi rerata menghitung rata-rata berat balita dengan menjumlahkan semua elemen dalam array dan membaginya dengan jumlah data yang dimasukkan. Setelah input

jumlah data berat balita dan berat masing-masing balita dimasukkan, program akan menampilkan berat balita minimum, maksimum, dan rata-rata.

III. KESIMPULAN

Kesimpulan dari program-program di atas adalah bahwa semua program tersebut menggunakan konsep dasar pencarian nilai ekstrim (terkecil dan terbesar) dalam berbagai konteks dengan memanfaatkan array untuk menyimpan data. Setiap program menghitung nilai minimum, maksimum, dan rata-rata dari sekumpulan data, seperti mencari berat kelinci, menghitung tarif ikan, dan mencatat data berat balita. Algoritma yang digunakan pada umumnya adalah iterasi sekuensial melalui array, dengan memperbarui nilai ekstrim (terkecil atau terbesar) saat memeriksa setiap elemen dalam array. Program-program ini juga menunjukkan pentingnya validasi data, serta penggunaan array dengan kapasitas yang sudah ditentukan untuk memanipulasi data. Secara keseluruhan, algoritma pencarian nilai ekstrim yang sederhana ini dapat diterapkan dalam berbagai kasus praktis dan memberikan gambaran yang jelas tentang cara mengolah data dalam pemrograman menggunakan bahasa Go.

IV. REFERENSI

- [1] Modul 11. Pencarian Nilai Ekstrim Pada Himpunan Data. Algoritma Pemrograman 2. Telkom University