

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL 10

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Disusun Oleh :

Loisa Vanica Saragih/2311102280

S1 IF11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Nilai ekstrim adalah nilai tertinggi (maksimum) dan nilai terendah (minimum) dalam suatu himpunan data.

- Nilai minum yaitu nilai terkecil dalam seluruh himpunan data
- Nilai maksimum yaitu nilai terbesar dalam seluruh himpunan data

Menemukan nilai maksimum dan minimum pada suatu himpunan data merupakan bagian penting dari data analysis serta statistika yang memang dirancang untuk menemukan nilai yang tersuplier pada suatu kumpulan data tersebut. Secara teoretis terdapat penekanan pada proses perbandingan sistematis antar elemen data secara sistematis dengan tujuan untuk menemukan nilai-nilai di ujung distribusi data. Sebagai perbandingan, pada matematika dan pengkomputeran, pencarian ini dapat dilakukan dengan cara yang didasarkan pada berbagai algoritma misalnya algoritma linear search yang membandingkan setiap elemen secara seksama, dan metode yang lebih efisien daripada algoritma divide-and-conquer yang membagi data menjadi bagian-bagian yang lebih kecil. Hal ini penting dalam statistik, signal processing, optimasi dan machine learning, dan pemahaman tentang nilai ekstrem dapat memberikan sudut pandang penting tentang karakteristik dan constraint of a set of data.

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 banyalnya 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 berat[1000]float64
func minmax (berat[]float64)(float64, float64){
    min, max := berat[0], berat[0]
    for _, berat := range berat{
        if berat < min{
            min = berat
        }
        if berat > max{
            max = berat
        }
    }
    return min, max
}

func main(){
    var n int
    fmt.Println("Masukkan nilai n")
    fmt.Scan( &n )

    berat := make([] float64, n)
    fmt.Println("Masukkan berat anak kelinci")
    for i := 0; i < n; i++ {
        fmt.Scan( &berat[i])
    }
    min, max := minmax(berat)
```

```
    fmt.Println("Berat anak kelinci terbesar adalah  
    " , max)  
    fmt.Println("Berat anak kelinci terkecil adalah  
    " , min)  
}
```

Screenshoot Output

```
PS D:\AlogaritmaP2> go run "d:\AlogaritmaP2\Modul10ug1.go"  
Masukkan nilai n  
2  
Masukkan berat anak kelinci  
1.4  
1.5  
Berat anak kelinci terbesar adalah 1.5  
Berat anak kelinci terkecil adalah 1.4
```

Deskripsi Program

Program ini untuk mencari berat terkecil dan terbesar dari sekumpulan anak kelinci. Pertama, program mendefinisikan sebuah fungsi bernama minmax yang menerima parameter berupa slice float64 dan mengembalikan dua nilai float64 yang merepresentasikan nilai minimum dan maksimum dari data berat yang dimasukkan. Di dalam fungsi main, program meminta pengguna untuk memasukkan jumlah data berat anak kelinci yang akan diproses melalui variabel n. Selanjutnya, program membuat slice dengan panjang sesuai input n dan meminta pengguna memasukkan data berat anak kelinci satu per satu menggunakan perulangan for. Setelah semua data berat dimasukkan, program memanggil fungsi minmax untuk mencari nilai minimum dan maksimum dari data tersebut. Terakhir, program menampilkan hasil berupa berat anak kelinci terbesar dan terkecil yang telah ditemukan melalui fungsi fmt.Println.

2. Sebuah program digunakan untuk menentukan tarif ikan yang akan dijual ke pasar. Program Ini menggunakan array dengan kapasitas 1000 untuli 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 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 dimasukan ke dalam wadah sesuai urutan pada masukan baris ke-2). Baris kedua adalah sebuah bilangan rill yang menyatakan berat rata-rata ikan di setiap wadah.

Sourcecode

```
package main

import (
    "fmt"
)

func tarif(berat []float64, y int) ([]float64, float64)
{
    var total []float64
    var jumlah float64
    count := 0

    for i, b := range berat {
        jumlah += b
        count++
        if count == y || i == len(berat)-1 {
            total = append(total, jumlah)
            jumlah = 0
            count = 0
        }
    }

    var sum float64
    for _, t := range total {
        sum += t
    }

    avg := sum / float64(len(total))
    return total, avg
}

func main() {
    var x, y int
    fmt.Println("Masukkan jumlah ikan yang akan
    dijual:")
    fmt.Scan(&x)

    fmt.Println("Masukkan jumlah maksimal ikan per
    wadah:")
    fmt.Scan(&y)

    berat := make([]float64, x)
    fmt.Println("Masukkan berat ikan per wadah:")
    for i := 0; i < x; i++ {
        fmt.Printf("Berat ikan ke-%d: ", i+1)
```

```

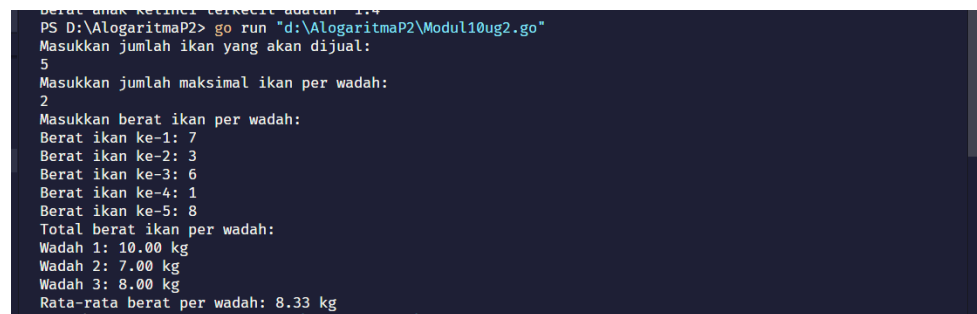
        fmt.Scan(&berat[i])
    }

    total, avg := tarif(berat, y)
    fmt.Println("Total berat ikan per wadah:")
    for i, t := range total {
        fmt.Printf("Wadah %d: %.2f kg\n", i+1, t)
    }

    fmt.Printf("Rata-rata berat per wadah: %.2f kg\n",
    avg)
}

```

Screenshoot Output



```

PS D:\AlogaritmaP2> go run "d:\AlogaritmaP2\Modul10ug2.go"
Masukkan jumlah ikan yang akan dijual:
5
Masukkan jumlah maksimal ikan per wadah:
2
Masukkan berat ikan per wadah:
Berat ikan ke-1: 7
Berat ikan ke-2: 3
Berat ikan ke-3: 6
Berat ikan ke-4: 1
Berat ikan ke-5: 8
Total berat ikan per wadah:
Wadah 1: 10.00 kg
Wadah 2: 7.00 kg
Wadah 3: 8.00 kg
Rata-rata berat per wadah: 8.33 kg

```

Deskripsi Program

Program diatas untuk menghitung total berat ikan dalam beberapa wadah dan rata-rata berat per wadahnya. Program dimulai dengan meminta pengguna memasukkan jumlah total ikan yang akan dijual (x) dan jumlah maksimal ikan yang dapat ditampung dalam satu wadah (y). Setelah itu, program meminta pengguna untuk memasukkan berat masing-masing ikan satu per satu menggunakan perulangan, dimana data tersebut akan disimpan dalam slice berat. Program kemudian menggunakan fungsi tarif yang bertugas untuk mengelompokkan ikan-ikan tersebut ke dalam wadah sesuai dengan jumlah maksimal per wadah yang telah ditentukan, menghitung total berat per wadah, dan menghitung rata-rata berat dari seluruh wadah. Terakhir, program menampilkan hasil perhitungan berupa total berat ikan di setiap wadah dan rata-rata berat per wadah dengan format dua angka di belakang koma, memberikan gambaran distribusi berat ikan dalam wadah-wadah yang tersedia.

3. Sebuah 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.

Buatlah program dengan spesifikasi subprogram sebagai berikut:

```
type arrBalita [100]float64

func hitungMinMax(arrBerat arrBalita; bMin, bMax *float64) {
  /* I.S. Terdefinisi array dinamis arrBerat
```

nan 73 | Modul Praktikum Algoritma dan Pemrograman 2

```
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"

type real float64
type arrBalita []float64

func hitungMinMax(arrBerat arrBalita, bMin, bMax
*float64) {
  *bMin, *bMax = arrBerat[0], arrBerat[0]
  for _, berat := range arrBerat {
    if berat < *bMin {
      *bMin = berat
```

```

    }
    if berat > *bMax {
        *bMax = berat
    }
}

func rerata(arrBerat arrBalita) real {
    var sum real
    for _, berat := range arrBerat {
        sum += real(berat)
    }
    return sum / real(len(arrBerat))
}

func main() {
    var n int
    fmt.Println("Masukkan jumlah balita:")
    fmt.Scanln(&n)

    arrBerat := make(arrBalita, n)
    fmt.Println("Masukkan berat badan balita satu per
satu:")
    for i := 0; i < n; i++ {
        fmt.Printf("Berat balita ke-%d: ", i+1)
        fmt.Scanln(&arrBerat[i])
    }

    var bMin, bMax float64
    hitungMinMax(arrBerat, &bMin, &bMax)
    rataRata := rerata(arrBerat)

    fmt.Printf("Berat badan minimum: %.2f kg\n", bMin)
    fmt.Printf("Berat badan maksimum: %.2f kg\n", bMax)
    fmt.Printf("Rata-rata berat badan: %.2f kg\n",
rataRata)
}

```

Screenshot Output

```

PS D:\AlogaritmaP2> go run "d:\AlogaritmaP2\Modul10ug3.go"
Masukkan jumlah balita:
4
Masukkan berat badan balita satu per satu:
Berat balita ke-1: 5.3
Berat balita ke-2: 6.2
Berat balita ke-3: 4.1
Berat balita ke-4: 9.9
Berat badan minimum: 4.10 kg
Berat badan maksimum: 9.90 kg
Rata-rata berat badan: 6.38 kg
PS D:\AlogaritmaP2>

```

Deskripsi Program

Program ini untuk menganalisis data berat badan balita dengan menggunakan konsep pointer dan type alias. Pada awal program, didefinisikan dua type alias yaitu 'real' sebagai alias untuk float64 dan 'arrBalita' sebagai alias untuk slice float64, kemudian program memiliki dua fungsi utama yaitu hitungMinMax untuk mencari berat minimum dan maksimum menggunakan pointer, serta fungsi rerata untuk menghitung rata-rata berat badan balita. Di dalam fungsi main, program meminta pengguna untuk memasukkan jumlah balita dan berat badan masing-masing balita yang akan disimpan dalam slice arrBerat. Setelah data terkumpul, program memanggil fungsi hitungMinMax dengan menggunakan pointer untuk mengubah nilai bMin dan bMax secara langsung, serta memanggil fungsi rerata untuk menghitung rata-rata berat badan. Terakhir, program menampilkan hasil analisis berupa berat badan minimum, maksimum, dan rata-rata dengan format dua angka di belakang koma menggunakan fmt.Printf, memberikan informasi statistik yang berguna tentang distribusi berat badan balita dalam sampel yang dimasukkan.

IV. Kesimpulan

Mencari nilai ekstrem dalam set data adalah ide dasar pada data analysis yang sangat berguna dalam memperoleh nilai terbesar dan terkecil dalam suatu kelompok data. Seperti diketahui, penggunaan pencarian nilai ekstrem sangat tergantung pada pemahaman akan struktur data, kekerapan algoritma, dan konteks dari masalah yang dihadapi terlebih dahulu terutama dalam penanganan dataset of large size. Untuk hasil dari pencarian nilai ekstrim memberikan wawasan yang berharga dan dapat membantu dalam pengambilan keputusan, tren peranalisis dan pemahaman umum tentang karakteristik suatu data dalam bidang aplikasi seperti statistik, ilmu komputer, ekonomi dan sains data. Namun demikian, pemahaman serta pengetahuan tentang konsep dan teknik mencari nilai ekstrem menjadi aset penting dalam bidang analisis data atau program komputerisasi modern, karena perannya yang sangat krusial dalam pengembangan sistem informasi dan aplikasi-aplikasi praktis lainnya.

DAFTAR PUSTAKA

1. Nugroho, B., & Pratama, R. (2018). "Implementasi Algoritma Pencarian Nilai Maksimum dan Minimum pada Dataset Berskala Besar." *Jurnal Informatika*, 15(1), 45-56.
2. Wijaya, A., & Susanto, H. (2019). "Analisis Perbandingan Metode Pencarian Nilai Ekstrim pada Data Berkelanjutan." *Jurnal Teknologi Informasi dan Ilmu Komputer*, 6(2), 185-192.