LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 7

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

Oleh:

BERTHA ADELA

103112400041

IF-12-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

Type: Bahasa pemrograman pada umumnya mengizinkan pemrograman untuk mengubah nama suatu tipe data dengan nama baru yang lebih ringkas dan familiar. Sebagai contoh "integer" dapat dirubah dengan nama alias "bilangan ". Caranya dengan menggunakan kata kunci "type".

Struct atau Record: Stucture memungkinkan pemrograman untuk mengelompokkan beberapa data atau nilai yang memiliki relasi atau keterkaitan tertentu menjadi suatu kesatuan. Masing-masing nilai tersimpan dalam field dari stucture tersebut. Berbeda dengan bahasa pemrograman lain. kesamaan tipe dari dua variabel berjenis stucture bukan karena namanya tetapi karena strukturnya. Dua variabel dengan nama-nama field dan tipe field yang sama (dan dalam urutan yang sama) dianggap mempunyai tipe yang sama. Tentunya akan lebih memudahkan jika stucture tersebut didefinisikan sebagai sebuah tipe baru, sehingga deklarasi stucture tidak perlu lagi seluruh field-nya ditulis ulang berkali-kali.

Array: Array mempunyai ukuran (jumlah elemen) yang tetap (statis) selama eksekusi program, sehingga jumlah elemen array menjadi bagian dari deklarasi variabel dengan tipe array.

Slice: Array dalam Go juga dapat mempunyai ukuran yang dinamik. (Tidak digunakan di kelas Algoritma Pemrograman). Deklarasinya mirip dengan deklarasi array, tetapi jumlah elemennya dikosongkan.

Fungsi built-in **len** dapat digunakan untuk mengetahui ukuran slice. Fungsi lain, **cap**, dapat digunakan untuk mengetahui total tempat yang disediakan untuk slice tersebut. Fungsi built-in **append** dapat digunakan untuk menambahkan elemen ke suatu slice, dan bila perlu memperbesar tempat untuk slice tersebut.

Ide 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 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.

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.

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.

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.

II. GUIDED

- **GUIDED 1**

Code:

SMT2 > Pertemuan7 > 103112400041_Guided1.go > main

```go
package main
import "fmt"
func main() {
    var n int
    fmt.Print("Masukkan jumlah tanaman: ")
    fmt.Scan(&n)

    var tinggiTanaman [500]float64
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan tinggi tanaman ke-%d (cm): ", i+1)
        fmt.Scan(&tinggiTanaman[i])
    }

    min, max := tinggiTanaman[0], tinggiTanaman[0]
    for i := 1; i < n; i++ {
        if tinggiTanaman[i] < min {
            min = tinggiTanaman[i]
        }
        if tinggiTanaman[i] > max {
            max = tinggiTanaman[i]
        }
    }
    fmt.Printf("\nTinggi tanaman tertinggi: %.2f cm\n", max)
    fmt.Printf("Tinggi tanaman terpendek: %.2f cm\n", min)
}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c
uided1.go"
Masukkan jumlah tanaman: 4
Masukkan tinggi tanaman ke-1 (cm): 21.21
Masukkan tinggi tanaman ke-2 (cm): 19.23
Masukkan tinggi tanaman ke-3 (cm): 23.53
Masukkan tinggi tanaman ke-4 (cm): 23.33

Tinggi tanaman tertinggi: 23.53 cm
Tinggi tanaman terpendek: 19.23 cm
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program diatas berguna untuk mencari tanaman tertinggi dan terpendeknya dari semua input pengguna ke dalam array.

- **GUIDED 2**

Code:

SMT2 > Pertemuan7 > 103112400041_Guided2.go > main

```go
package main
import "fmt"
func main() {
    var x, y int
    fmt.Print("Masukkan jumlah buku dan jumlah buku per rak: ")
    fmt.Scan(&x, &y)

    var hargaBuku [500]float64
    fmt.Println("\nMasukkan harga setiap buku (dalam ribuan Rp): ")
    for i:= 0; i<x; i++ {
        fmt.Scan(&hargaBuku[i])
    }

    var hargaRataRata []float64
    for i:=0; i<x; i+= y {
        total := 0.0
        for j := i; j < i+y && j < x; j++{
            total += hargaBuku[j]
        }
        hargaRataRata = append(hargaRataRata, total/float64(y))
    }
    min,max := hargaBuku[0], hargaBuku[0]
    for _, harga := range hargaBuku[:x] {
        if harga < min {
            min = harga
        }
        if harga > max {
            max = harga
        }
    }
    fmt.Printf("\nRata-rata harga per rak: ")
    for _, avg:= range hargaRataRata {
        fmt.Printf("%.2f ", avg)

    }
    fmt.Printf("\nHarga termahal: %.2f Rp\n", max)
    fmt.Printf("\nHarga termurah: %.2f Rp\n", min)
}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\
uided2.go"
Masukkan jumlah buku dan jumlah buku per rak: 6 2

Masukkan harga setiap buku (dalam ribuan Rp):
20000
50000
75000
90000
121000
125000

Rata-rata harga per rak: 35000.00 82500.00 123000.00
Harga termahal: 125000.00 Rp

Harga termurah: 20000.00 Rp
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program diatas akan menghitung rata rata harga buku di setiap rak kemudian mencari harga buku termahal dan harga buku termurahnya.

- **GUIDED 3**

Code:

SMT2 > Pertemuan7 > 103112400041_Guided3.go > main

```go
package main
import "fmt"
func main() {
    var n int
    fmt.Print("Masukkan jumlah siswa: ")
    fmt.Scan(&n)

    var nilaiSiswa [200]float64
    var totalNilai float64 = 0

    fmt.Println("\nMasukkan nilai ujian masing-masing siswa: ")
    for i:= 0; i<n; i++ {
        fmt.Scan(&nilaiSiswa[i])
        totalNilai += nilaiSiswa[i]
    }
    rataRata := totalNilai / float64(n)
```

```go
    min, max := nilaiSiswa[0], nilaiSiswa[0]
    var diAtasRataRata int = 0
    for _, nilai := range nilaiSiswa[:n] {
        if nilai < min {
            min = nilai
        }
        if nilai > max {
            max = nilai
        }
        if nilai > rataRata {
            diAtasRataRata++
        }
    }

    fmt.Printf("\nNilai terendah: %.0f\n", min)
    fmt.Printf("Nilai tertinggi: %.0f\n", max)
    fmt.Printf("Rata-rata kelas: %.2f\n", rataRata)
    fmt.Printf("Jumlah siswa di atas rata-rata: %d\n", diAtasRataRata)
}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\O
uided3.go"
Masukkan jumlah siswa: 4

Masukkan nilai ujian masing-masing siswa:
88
78
79
84

Nilai terendah: 78
Nilai tertinggi: 88
Rata-rata kelas: 82.25
Jumlah siswa di atas rata-rata: 2
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program diatas akan mencari nilai terendah, nilai tertinggi, dan rata rata nilai ujian siswa.

III. UNGUIDED

- **UNGUIDED 1**

Code:

SMT2 > Pertemuan7 > 103112400041_Unguided1.go > ...

```go
//BERTHA ADELA
//103112400041
package main

import "fmt"

func main() {
    var banyakAnakKelinci int
    var beratKelinci float64
    var dataBerat[1000]float64
    fmt.Print("Banyak anak kelinci: ")
    fmt.Scanln(&banyakAnakKelinci)
    beratMin := dataBerat[0]
    beratMax := dataBerat[0]
    hitung := 0
    if banyakAnakKelinci <= len(dataBerat) {
        for i := 0; i < banyakAnakKelinci; i++ {
            hitung += 1
            fmt.Printf("Masukkan berat anak kelinci (%d/%d): ", hitung, banyakAnakKelinci)
            fmt.Scanln(&beratKelinci)
            dataBerat[i] = beratKelinci
            if i == 0 {
                beratMin = beratKelinci
                beratMax = beratKelinci
            } else {
                if beratKelinci > beratMax {
                    beratMax = beratKelinci
                }
                if beratKelinci < beratMin {
                    beratMin = beratKelinci
                }
            }
        }
        fmt.Printf("Berat anak kelinci terkecilnya adalah %.2f", beratMin)
        fmt.Printf("\nBerat anak kelinci terbesarnya adalah %.2f", beratMax)
    } else {
        fmt.Println("Jumlah anak kelinci melebihi batas maksimal (1000)")
    }
}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDr
nguided1.go"
Banyak anak kelinci: 5
Masukkan berat anak kelinci (1/5): 11.23
Masukkan berat anak kelinci (2/5): 10.87
Masukkan berat anak kelinci (3/5): 9.73
Masukkan berat anak kelinci (4/5): 6.54
Masukkan berat anak kelinci (5/5): 9.32
Berat anak kelinci terkecilnya adalah 6.54
Berat anak kelinci terbesarnya adalah 11.23
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program ini berguna untuk mencari berat terkecil dan berat terbesar anak kelinci.

- **UNGUIDED 2**

Code:

```go
//BERTHA ADELA
//103112400041
package main

import "fmt"

func main() {
    var dataBerat[1000] float64
    var ikanAkanDiJual, ikanTiapWadah int
    var maxWadah[1000] float64
    fmt.Print("Berapa ikan yang mau dijual dan berapa ikan tiap wadahnya: ")
    fmt.Scanln(&ikanAkanDiJual, &ikanTiapWadah)
    if ikanAkanDiJual <= len(dataBerat) {
        wadah := (ikanAkanDiJual+ikanTiapWadah-1)/ikanTiapWadah
        for i := 0; i < ikanAkanDiJual; i++ {
                fmt.Printf("Berat ikan (%d/%d): ", i+1, ikanAkanDiJual)
                fmt.Scanln(&dataBerat[i])
        }
        for i := 0; i < wadah; i++ {
            mulai := i * ikanTiapWadah
            akhir := mulai + ikanTiapWadah
            if akhir > ikanAkanDiJual {
                akhir = ikanAkanDiJual
            }
            beratTiapWadah := 0.0
            for j := mulai; j < akhir; j++ {
                beratTiapWadah += dataBerat[j]
            }
            maxWadah[i] = beratTiapWadah
        }
        fmt.Println("Total berat per wadah:")
        for i := 0; i < wadah; i++ {
            fmt.Printf("%.2f ", maxWadah[i])
        }
        fmt.Println()
        total := 0.0
        for i := 0; i < wadah; i++ {
            total += maxWadah[i]
        }
        rataRata := total / float64(wadah)
        fmt.Printf("Rata-rata berat setiap wadah: %.2f\n", rataRata)
    } else {
        fmt.Println("Jumlah ikan melebihi batas maksimal(1000)")
    }

}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\
nguided2.go"
Berapa ikan yang mau dijual dan berapa ikan tiap wadahnya: 5 2
Berat ikan (1/5): 28.22
Berat ikan (2/5): 19.23
Berat ikan (3/5): 39.33
Berat ikan (4/5): 21.03
Berat ikan (5/5): 42.21
Total berat per wadah:
47.45 60.36 42.21
Rata-rata berat setiap wadah: 50.01
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program ini berguna untuk menghitung rata-rata ikan tiap wadah, ikan terkecil, dan ikan terbesar.

- **UNGUIDED 3**

Code:

SMT2 > Pertemuan7 > 103112400041_Unguided3.go > rerata

```go
//BERTHA ADELA
//103112400041

package main

import "fmt"

type arrBalita [100]float64
func hitungMinMax(arrBerat arrBalita, bMin, bMax *float64) {
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]
    for i := 1; arrBerat[i] != -1.0; i++ {
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
    }
}
func rerata(arrBerat arrBalita) float64 {
    total := 0.0
    jumlah := 0
    for i := 0; arrBerat[i] != -1.0; i++ {
        total += arrBerat[i]
        jumlah=jumlah+1
    }
    if jumlah == 0 {
        return 0
    }
    return total / float64(jumlah)
}
func main() {
    var bb arrBalita
    var jumlahBalita int
    var berat float64
    var beratMin, beratMax float64
    fmt.Print("Masukan banyak data berat balita: ")
    fmt.Scanln(&jumlahBalita)

    for i := 0; i < jumlahBalita; i++ {
        fmt.Printf("Masukkan berat balita ke-%d: ", i+1)
        fmt.Scanln(&berat)
        bb[i] = berat
    }
    bb[jumlahBalita] = -1.0
    hitungMinMax(bb, &beratMin, &beratMax)
    rerata := rerata(bb)
    fmt.Printf("Berat balita minimum: %.2f\n kg", beratMin)
    fmt.Printf("Berat balita maksimum: %.2f\n kg", beratMax)
    fmt.Printf("Rerata berat balita: %.2f\n kg", rerata)
}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Document
nguided3.go"
Masukan 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:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program ini dapat menghitung berat minimum balita, berat maksimum balita, serta rata-rata berat balita.

IV. KESIMPULAN

Manfaat array bermacam-macam, yaitu memudahkan dalam menyusun suatu data, membuat data menjadi lebih rapi dan mudah dilihat, mudah di akses, lebih sederhana namun tetap estetik, dan lebih efisien dalam penggunaannya.

REFERENSI

MODUL 6 STRUCT & ARRAY
MODUL 7 PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA