LAPORAN PRAKTIKUM
ALGORITME DAN PEMEROGRAMAN

MODUL 10

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

Oleh:

ERVAN HAPIZ

2311102206

IF – 11- 02

S1 TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

MODUL 10. PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

10.1 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. 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	`max ← 1`	`max = 0`
2	`i ← 2`	`i = 1`
3	`while i <= n do`	`for i < n {`
4	`    if a[i] > a[max]`	`    a[i] > a[max] {`
5	`        max ← i`	`        max = i`
6	`    endif`	`    }`
7	`    i ← i + 1`	`    i = i + 1`
8	`endwhile`	`}`

10.2 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!

```
...
```

```
5   type arrInt [2023]int
..  ...
15
16  func terkecil_1(tabInt arrInt, n int) int {
17  /* mengembalikan nilai terkecil yang terdapat di dalam tabInt yang berisi n
18  bilangan bulat */
19      var min int = tabInt[0]        // min berisi data pertama
20      var j int = 1                  // pencarian dimulai dari data berikutnya
21      for j < n {
22          if min > tabInt[j] {       // pengecekan apakah nilai minimum valid
23              min = tabInt[j]        // update nilai minimum dengan yang valid
24          }
25          j = j + 1
26      }
27      return min                     // returnkan nilai minimumnya
28  }
```

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!

```
5   ...
..  type arrInt [2023]int
15  ...
16
17  func terkecil_2(tabInt arrInt, n int) int {
18  /* mengembalikan indeks nilai terkecil yang terdapat di dalam tabInt yang berisi
19  n bilangan bulat */
20      var idx int = 0                // idx berisi indeks data pertama
21      var j int = 1                  // pencarian dimulai dari data berikutnya
22      for j < n {
23          if tabInt[idx] > tabInt[j] { // pengecekan apakah nilai minimum valid
24              idx = j                // update nilai minimum dengan yang valid
25          }
26          j = j + 1
27      }
28      return idx                     // returnkan indeks nilai minimumnya
    }
```

10.3 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

sudah diberikan sebelumnya, maka pencarian yang dilakukan bisa mengembalikan indeks mahasiswa dengan ipk tertinggi tersebut. Berikut ini adalah modifikasinya!

```
...
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 di atas, identitas mahasiswa dapat diperoleh, misalnya `T[idx].nama`, `T[idx].nim`, `T[idx].kelas`, hingga `T[idx].jurusan`.

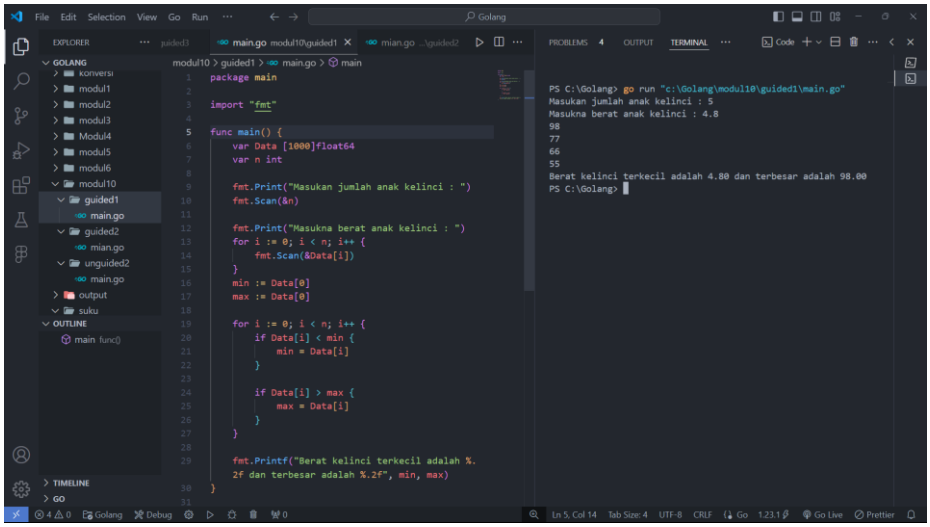
II. GUIDED

1. Guided 1

Source Code

```
package main

import "fmt"

func main() {
    var Data [1000]float64
    var n int

    fmt.Print("Masukan jumlah anak kelinci : ")
    fmt.Scan(&n)

    fmt.Print("Masukna berat anak kelinci : ")
    for i := 0; i < n; i++ {
        fmt.Scan(&Data[i])
    }
    min := Data[0]
    max := Data[0]

    for i := 0; i < n; i++ {
        if Data[i] < min {
            min = Data[i]
        }

        if Data[i] > max {
            max = Data[i]
        }
    }

    fmt.Printf("Berat kelinci terkecil adalah
%.2f dan terbesar adalah %.2f", min, max)
}
```

Screenshot



Deskripsi

Program ini adalah program untuk mencari berat maksimal kelinci. Pada fungsi main terdapat deklarasi variable array data dengan size 1000 type data float. Kemudian variable n int untuk inputan jumlah kelinci. Selanjutnya program akan menerima input dari user jumlah anak kelinci. Selanjutnya array akan diisi. Terdapat delarasi var min dan max dengan nilai yaitu array indek 0. Kemudian untuk mencari nilai min dan max dilakukan perulangan dengan i dari 0 ghingga n kemudian terdapat if data[i] < min maka nilai min akan diganti dengan data[i]. begitu juga dengan max if data [i] > max maka max = data[i]. kemudian program akan menampilkan nilai min dan max.

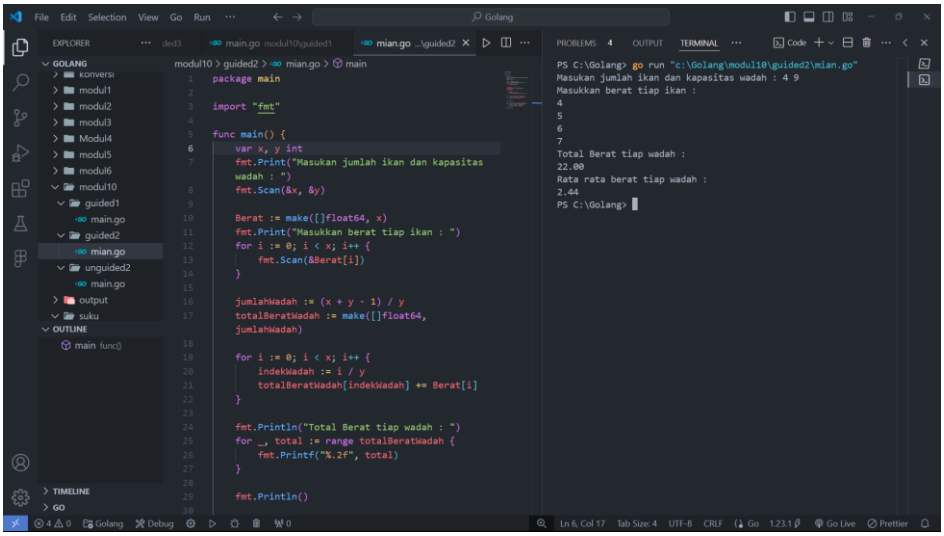
2. **Guided 2**

Source Code

```
package main

import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukan jumlah ikan dan kapasitas
wadah : ")
    fmt.Scan(&x, &y)

    Berat := make([]float64, x)
    fmt.Print("Masukkan berat tiap ikan : ")
    for i := 0; i < x; i++ {
        fmt.Scan(&Berat[i])
    }

    jumlahWadah := (x + y - 1) / y
    totalBeratWadah := make([]float64,
jumlahWadah)

    for i := 0; i < x; i++ {
```

```
            indekWadah := i / y
            totalBeratWadah[indekWadah] += Berat[i]
        }

        fmt.Println("Total Berat tiap wadah : ")
        for _, total := range totalBeratWadah {
            fmt.Printf("%.2f", total)
        }

        fmt.Println()

        fmt.Println("Rata rata berat tiap wadah : ")
        for _, total := range totalBeratWadah {
            ratarata := total / float64(y)
            fmt.Printf("%.2f", ratarata)
        }
        fmt.Println()
    }
```

Screenshot



Deskripsi

Program ini adalah program untuk menghitung rata rata berat ikan tiap wadah. Terdapat var x untuk menyimpan jumlah ikan. y menyimpan kapasitas wadah (maksimal jumlah ikan per wadah). Slice Berat berisi berat masing-masing ikan. Pengguna memasukkan berat tiap ikan yang disimpan dalam slice Berat. Kemudian jumlahWadah menghitung jumlah wadah yang dibutuhkan menggunakan rumus (x + y - 1) / y untuk mengakomodasi semua ikan. totalBeratWadah adalah slice untuk menyimpan total berat ikan dalam setiap wadah. Loop mengiterasi setiap ikan dan menghitung indeks wadah (indekWadah) di mana ikan tersebut akan ditempatkan. Berat ikan ditambahkan ke total berat pada indeks wadah

yang sesuai. Kemudian dilakukan perulangan dengan for i range totalberatwadah menampilkan total berat setiap wadah. Perulngan selanjutnya menghitung dan menampilkan rata-rata berat ikan per wadah, dengan membagi total berat tiap wadah dengan kapasitas wadah (y).

III. UNGUIDED

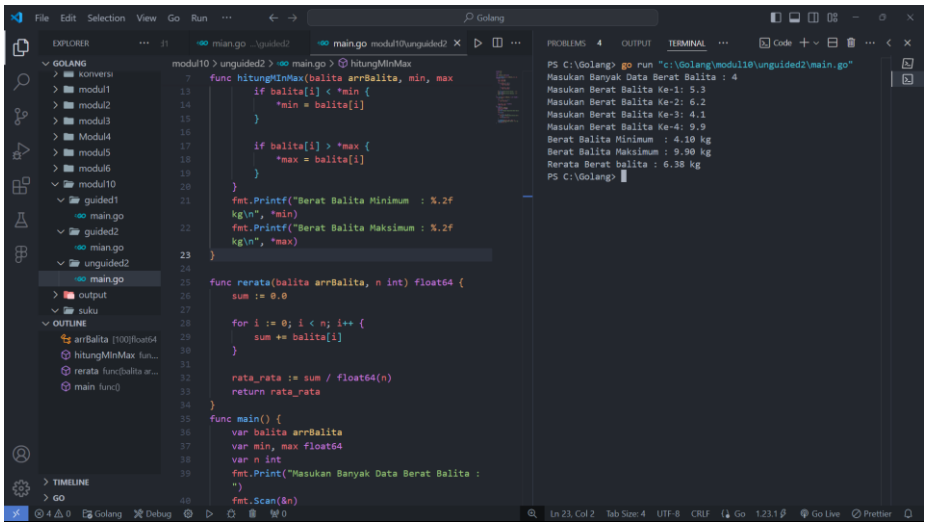
1. Unguided 1

Source Code

```
package main

import "fmt"

type arrBalita [100]float64

func hitungMInMax(balita arrBalita, min, max
*float64, n int) {

    *min = balita[0]
    *max = balita[0]

    for i := 0; i < n; i++ {
        if balita[i] < *min {
            *min = balita[i]
        }

        if balita[i] > *max {
            *max = balita[i]
        }
    }
    fmt.Printf("Berat Balita Minimum  : %.2f
kg\n", *min)
    fmt.Printf("Berat Balita Maksimum : %.2f
kg\n", *max)
}

func rerata(balita arrBalita, n int) float64 {
    sum := 0.0

    for i := 0; i < n; i++ {
        sum += balita[i]
    }

    rata_rata := sum / float64(n)
```

```
        return rata_rata
    }
    func main() {
        var balita arrBalita
        var min, max float64
        var n int
        fmt.Print("Masukan Banyak Data Berat Balita :
    ")
        fmt.Scan(&n)

        for i := 0; i < n; i++ {
            fmt.Printf("Masukan Berat Balita Ke-%v:
    ", i+1)
            fmt.Scan(&balita[i])
        }

        hitungMInMax(balita, &min, &max, n)
        fmt.Printf("Rerata Berat balita : %.2f kg",
    rerata(balita, n))
    }
```

Screenshot



Deskripsi

Program ini adalah program mencari nilai min, max dan rata-rata. Terdapat type arrbalita dengan size 100 dan tipe data float. Kemduian terdapat fungsi untuk menghitung nilai min dan max . pada fungsi var min dan max dideklerasikan dengan balita[0]. Kemudian untuk mencari nilai min dan max dilakukan perulangan sebanyak n kali, jika terdapat nilai data balita lebih kecil dari min maka nilai min akan diganti nmenjadi

nilai balita[i]. kemudian max ketika ada yang lebih besar maka nilai max akan diganti menjadi nbalita[i]. terdapat juga fungsi rerata untuk mengitung nilai rata rata dari array balita. Dalam fungsi terdapat perulangan dengan sum akan ditambah dengan nilai array balita. Sebanyak n kali. Kemudian sum akan dibagi dengan jumlah data untuk mencari rata rata. Fungsi kemudfian membalikan nilai rata rata. Dalam fungsi main terdapat deklarasiu variable balita dengan array, min, max dan n. kemudian program akan menerima inputan nilai n. kemdian program akan melakukan perulangan sebnayak n kali untuk menginput nilai array balita. Kemudian fsungdi minmax dipanggil dan fungsi rata rata untuk menampilkan nilai minimal dan maxsimal dan rata rata dari berat balita.