LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2

MODUL 10

MIN & MAX

Oleh:

PANDIA ARYA BRATA

2311102076

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
}
```

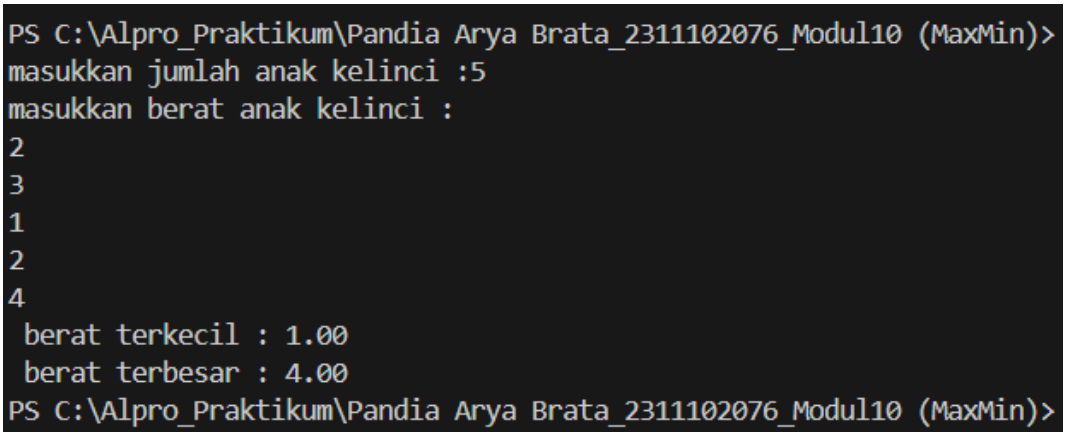
Sehingga melalui algoritma di atas, identitas mahasiswa dapat diperoleh, misalnya `T[idx].nama`, `T[idx].nim`, `T[idx].kelas`, hingga `T[idx].jurusan`.

I. GUIDED

1.GUIDED (No.1|2A)

```
package main

import (
    "fmt"
)

func main() {
    var N int
    var berat [1000]float64

    fmt.Print("masukkan jumlah anak kelinci :")
    fmt.Scan(&N)

    fmt.Println("masukkan berat anak kelinci :")
    for i := 0; i < N; i++ {
        fmt.Scan(&berat[i])
    }

    // inisialisasi nilai min dan max dg elemen pertama
    min := berat[0]
    max := berat[0]

    for i := 1; i < N; i++ {
        if berat[i] < min {
            min = berat[i]
        }
        if berat[i] > max {
            max = berat[i]
        }
    }
    fmt.Printf(" berat terkecil : %.2f\n", min)
    fmt.Printf(" berat terbesar : %.2f\n", max)
}
```

Screenshot program:

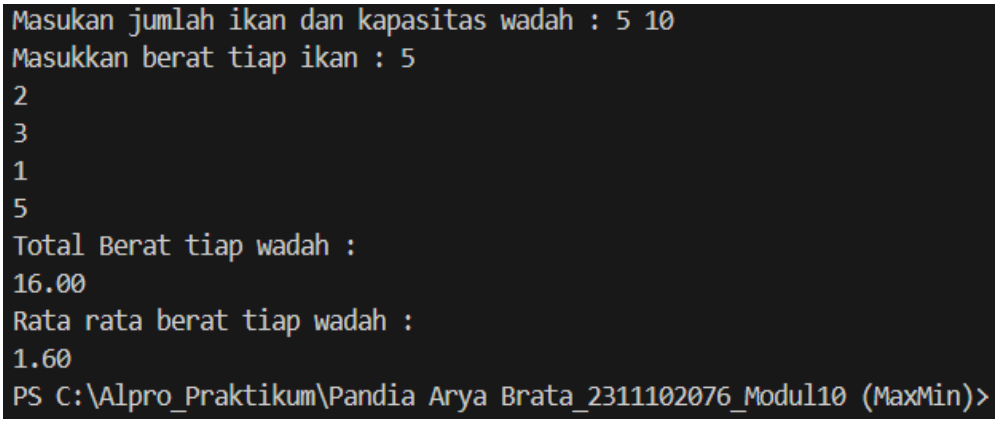


Deskripsi program :
Program ini berisi program yang menghitung dan menampilkan berat terkecil dan terbesar dari sejumlah anak kelinci berdasarkan input berat yang diberikan oleh pengguna.

2.GUIDED(No.2|2A)

```
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 program :



Deskripsi program :

Program ini menghitung total berat dan rata-rata berat ikan dalam wadah berdasarkan input jumlah ikan dan kapasitas wadah, serta menampilkan hasil perhitungan secara terstruktur dan informatif.

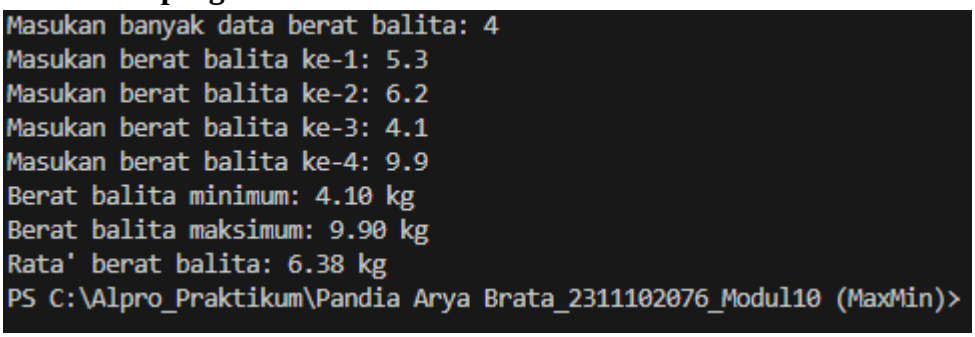
II. UNGUIDED

1.UNGUIDED

```
package main

import "fmt"

func reamur(cel float64) {
    r := (4.0 / 5.0) * cel
    fmt.Println("Derajat Reamur : ", r)
}

func fahrenheit(cel float64) {
    f := (9.0 / 5.0 * cel) + 32
    fmt.Println("Derajat Fahrenheit : ", f)
}

func kelvin(cel float64) {
    k := cel + 273.15
    fmt.Println("Derajat Kelvin : ", k)
}

func main() {
    var cel float64
    fmt.Print("Drajat Celcius : ")
    fmt.Scan(&cel)
    reamur(cel)
```

```
    fahrenheit(cel)
    kelvin(cel)
}
```

Screenshot program :



Deskripsi program :

Program ini memiliki 2 fungsi, fungsi pertama untuk menentukan minimum dan maksimum berat balita, dan fungsi kedua adalah menghitung rata-rata dari berat balita, program memerlukan inputan dari pengguna.