LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL XI

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

Disusun Oleh :

Raihan Ramadhan/2311102040

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 nilai ekstrim maksimum dan minimum dalam sebuah himpunan data atau fungsi adalah proses untuk menemukan nilai tertinggi (maksimum) dan nilai terendah (minimum). Himpunan data ini bisa berupa angka, karakter, atau elemen lain yang dapat dibandingkan. Nilai maksimum adalah nilai terbesar dalam suatu himpunan data, sedangkan nilai minimum adalah nilai terkecil dalam himpunan tersebut.

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] then`	`    if a[i] > a[max] {`
5	`        max ← i`	`        max = i`
6	`    endif`	`    }`
7	`    i ← i + 1`	`    i = i + 1`
8	`endwhile`	`}`

Pendekatan algoritma ini sangat sederhana. Data akan diproses secara berurutan, di mana nilai atau indeks yang merupakan nilai maksimum dari data yang telah diproses akan disimpan untuk dibandingkan dengan data berikutnya. Nilai yang tersimpan hingga akhir algoritma merupakan nilai maksimum yang dicari.

Secara umum, langkah-langkah algoritma ini adalah sebagai berikut:

1. Tetapkan data pertama sebagai nilai ekstrim awal.
2. Periksa setiap data, mulai dari elemen kedua hingga elemen terakhir:
    - Jika nilai ekstrim saat ini tidak valid, perbarui nilai ekstrim dengan data yang sedang diperiksa.
3. Setelah semua data selesai diperiksa, nilai ekstrim yang diperoleh merupakan nilai yang valid.

II. UNGUIDED

1. 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.

Source Code

```
package main

import (
    "fmt"
)

func main() {
    const kapasitas_2311102040 = 1000
    var berat [kapasitas_2311102040]float64 // Array dengan kapasitas tetap
1000

    var n int
    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&n)

    // Validasi jumlah anak kelinci
    if n <= 0 || n > kapasitas_2311102040  {
        fmt.Printf("Jumlah anak kelinci harus antara 1 dan %d.\n",
kapasitas_2311102040 )
        return
    }

    fmt.Println("Masukkan berat anak kelinci:")

    for i := 0; i < n; i++ {
        fmt.Printf("Berat kelinci ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }

    // Mencari berat terkecil dan terbesar
    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 Output

```
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
Masukkan jumlah anak kelinci: 5
Masukkan berat anak kelinci:
Berat kelinci ke-1: 2.3
Berat kelinci ke-2: 2.5
Berat kelinci ke-3: 2.2
Berat kelinci ke-4: 3.5
Berat kelinci ke-5: 4.2
Berat terkecil: 2.20
Berat terbesar: 4.20
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
```

Deskripsi Program

Program ini dibuat untuk mencatat dan mencari berat terkecil dan terbesar dari anak kelinci yang akan dijual, dengan kapasitas data hingga 1000 berat. Pengguna diminta memasukkan jumlah kelinci yang akan ditimbang, dan program akan mengecek apakah jumlah tersebut sudah sesuai, yaitu antara 1 sampai 1000. Setelah itu, pengguna memasukkan berat tiap kelinci satu per satu, yang akan disimpan di dalam daftar data (array). Walaupun kapasitasnya bisa sampai 1000, hanya jumlah yang dimasukkan oleh pengguna yang dipakai. Program kemudian mencari berat paling ringan dan paling berat dengan cara membandingkan semua data, lalu menampilkan hasilnya berupa dua angka yang menunjukkan berat terkecil dan terbesar.

2. UNGUIDED 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 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 riil yang menyatakan berat rata-rata ikan di setiap wadah.

Source Code

```
package main

import (
    "fmt"
)

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

    // Membuat array untuk menyimpan berat ikan
    beratIkan_2311102040  := make([]float64, x)
    fmt.Println("Masukkan berat ikan satu per satu:")

    // Memasukkan berat ikan ke dalam array
    for i := 0; i < x; i++ {
        fmt.Printf("Berat ikan ke-%d: ", i+1)
        fmt.Scan(&beratIkan_2311102040 [i])
    }

    // Menghitung jumlah wadah
    jumlahWadah := (x + y - 1) / y // Menggunakan pembulatan ke atas

    // Menyimpan total berat di setiap wadah
    totalBeratPerWadah := make([]float64, jumlahWadah)
    for i := 0; i < x; i++ {
        wadahIndex := i / y
        totalBeratPerWadah[wadahIndex] += beratIkan_2311102040 [i]
    }
```

```
    // Menampilkan total berat per wadah
    fmt.Println("Total berat ikan di setiap wadah:")
    for i := 0; i < jumlahWadah; i++ {
        fmt.Printf("Wadah %d: %.2f\n", i+1, totalBeratPerWadah[i])
    }

    // Menghitung berat rata-rata
    totalBeratSemuaWadah := 0.0
    for _, berat := range totalBeratPerWadah {
        totalBeratSemuaWadah += berat
    }
    beratRataRata := totalBeratSemuaWadah / float64(jumlahWadah)

    // Menampilkan berat rata-rata
    fmt.Printf("Berat rata-rata di setiap wadah: %.2f\n", beratRataRata)
}
```

Screenshot Output

```
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
Masukkan jumlah anak kelinci: 5
Masukkan berat anak kelinci:
Berat kelinci ke-1: 2.3
Berat kelinci ke-2: 2.5
Berat kelinci ke-3: 2.2
Berat kelinci ke-4: 3.5
Berat kelinci ke-5: 4.2
Berat terkecil: 2.20
Berat terbesar: 4.20
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
```

Deskripsi Program

Program ini digunakan untuk menghitung total berat ikan di setiap wadah dan berat rata-rata ikan di semua wadah, berdasarkan jumlah ikan yang akan dijual (x) dan berapa banyak ikan yang bisa dimasukkan ke dalam satu wadah (y). Pertama, pengguna diminta untuk memasukkan jumlah ikan dan kapasitas ikan per wadah, lalu program akan menerima berat masing-masing ikan dan menyimpannya dalam sebuah array. Setelah itu, ikan dibagi ke dalam beberapa wadah dengan menjumlahkan berat ikan satu per satu hingga setiap wadah terisi sesuai kapasitas maksimal. Program kemudian menampilkan total berat ikan di setiap wadah dan menghitung berat rata-rata ikan di semua wadah dengan cara membagi total berat dengan jumlah wadah. Dengan cara ini, program membantu mengorganisir dan menghitung berat ikan dengan mudah untuk kebutuhan distribusi atau penjualan.

3. UNGUIDED 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. Buatlah program dengan spesifikasi subprogram sebagai berikut:

```
type arrBalita [100]float64

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

```
  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
```

Source Code

```go
package main

import (
    "fmt"
)

// Fungsi untuk menghitung nilai minimum dan maksimum dari array
func hitungMinMax(arrBerat []float64, bMin *float64, bMax *float64) {
```

```
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]

    for _, berat := range arrBerat {
        if berat < *bMin {
            *bMin = berat
        }
        if berat > *bMax {
            *bMax = berat
        }
    }
}

// Fungsi untuk menghitung rata-rata berat balita
func hitungRatarata(arrBerat []float64) float64 {
    total_2311102040 := 0.0
    for _, berat := range arrBerat {
        total_2311102040 += berat
    }
    return total_2311102040 / float64(len(arrBerat))
}

func main() {
    var n int
    fmt.Print("Masukan banyak data berat balita: ")
    fmt.Scan(&n)

    // Inisialisasi array untuk menyimpan berat balita
    arrBerat := make([]float64, n)

    // Input berat balita
    for i := 0; i < n; i++ {
        fmt.Printf("Masukan berat balita ke-%d: ", i+1)
        fmt.Scan(&arrBerat[i])
    }

    // Deklarasi variabel untuk menyimpan nilai minimum dan maksimum
    var bMin, bMax float64

    // Hitung nilai minimum dan maksimum
    hitungMinMax(arrBerat, &bMin, &bMax)

    // Hitung rata-rata berat balita
    rerata := hitungRatarata(arrBerat)

    // Tampilkan 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", rerata)
}
```

Screenshot Output

```
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
Masukan banyak data berat balita: 5
Masukan berat balita ke-1: 10
Masukan berat balita ke-2: 11.7
Masukan berat balita ke-3: 12.5
Masukan berat balita ke-4: 13
Masukan berat balita ke-5: 12.8
Berat balita minimum: 10.00 kg
Berat balita maksimum: 13.00 kg
Rerata berat balita: 12.00 kg
PS D:\Kuliah\Matkul\SEMESTER 3\ALPRO PRAKTEK\PRAKTIKUM\MODUL XI>
```

Deskripsi Program

Program ini dibuat untuk menghitung berat balita dengan cara memasukkan beberapa data berat yang berupa angka desimal. Pengguna diminta untuk menentukan jumlah data, lalu berat setiap balita dimasukkan ke dalam sebuah array dengan tipe float64. Fungsi hitungMinMax bertugas untuk mencari nilai terkecil dan terbesar dari array dengan cara membandingkan setiap elemen dalam array. Sementara itu, fungsi hitungRerata digunakan untuk menghitung rata-rata berat balita dengan menjumlahkan seluruh nilai dalam array dan membaginya dengan total data yang ada. Setelah selesai melakukan perhitungan, program akan menampilkan berat balita terendah, tertinggi, dan rata-rata berat dari data yang dimasukkan pengguna.