

LAPORAN PRAKTIKUM
Algoritma Pemrograman
MODUL X
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Oleh:

Ilhan Sahal Mansiz

2311102029

IF-11-02

S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

1. Ide Pencarian Nilai Maksimum/Minimum

Pencarian nilai maksimum atau minimum adalah operasi dasar untuk menemukan elemen terbesar atau terkecil dalam suatu himpunan data. Operasi ini dilakukan dengan membandingkan setiap elemen dalam himpunan dengan nilai maksimum atau minimum sementara hingga seluruh elemen diproses.

Langkah utama:

- Inisialisasi nilai ekstrem (maksimum atau minimum) dengan elemen pertama.
- Bandingkan elemen berikutnya dengan nilai ekstrem sementara.
- Perbarui nilai ekstrem jika ditemukan elemen yang lebih besar atau lebih kecil.

2. Pencarian Nilai Ekstrem pada Array Bertipe Data Dasar

Pada array bertipe data dasar seperti int, float, atau string, algoritma pencarian nilai ekstrem dapat dilakukan dengan menggunakan iterasi sederhana. Contoh dalam bahasa Go:

```
package main

import (
    "fmt"
)

func findExtremes(data []int) (int, int) {
    min := data[0]
    max := data[0]

    for _, value := range data {
        if value < min {
            min = value
        }
        if value > max {
            max = value
        }
    }

    return min, max
}
```

```

    }

    func main() {
        data := []int{3, 7, 2, 9, 5}
        min, max := findExtremes(data)
        fmt.Printf("Nilai minimum: %d\n", min)
        fmt.Printf("Nilai maksimum: %d\n", max)
    }

```

3. Pencarian Nilai Ekstrem pada Array Bertipe Data Terstruktur

Jika array berisi tipe data terstruktur seperti struct, pencarian nilai ekstrem dilakukan dengan membandingkan salah satu atribut struct. Contoh:

```

package main

import (
    "fmt"
)

type Person struct {
    Name string
    Age  int
}

func findOldestAndYoungest(people []Person) (Person, Person) {
    youngest := people[0]
    oldest := people[0]

    for _, person := range people {
        if person.Age < youngest.Age {
            youngest = person
        }
        if person.Age > oldest.Age {
            oldest = person
        }
    }

    return youngest, oldest
}

```

```

    }

    func main() {
        people := []Person{
            {"Alice", 25},
            {"Bob", 30},
            {"Charlie", 20},
            {"Diana", 35},
        }

        youngest, oldest := findOldestAndYoungest(people)
        fmt.Printf("Termuda: %s (%d tahun)\n", youngest.Name,
youngest.Age)
        fmt.Printf("Tertua: %s (%d tahun)\n", oldest.Name,
oldest.Age)
    }

```

Kesimpulan

- Pencarian nilai ekstrem dilakukan dengan membandingkan setiap elemen dalam array dengan nilai sementara.
- Untuk array bertipe data dasar, cukup membandingkan langsung nilainya.
- Untuk array bertipe data terstruktur, pencarian dilakukan berdasarkan atribut tertentu dalam struct.

II. GUIDED

1. Guided 1

Source Code :

```
package main

import (
    "fmt"
)

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

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

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

    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 :

```
Masukan jumlah anak kelinci: 4
Masukan berat anak kelinci:
3 5 2 8
Berat terkecil: 2.00
Berat terbesar: 8.00
```

Deskripsi Program :

Program ini digunakan untuk mencari berat terkecil dan terbesar dari sejumlah anak kelinci berdasarkan data berat yang dimasukkan oleh pengguna. Program dimulai dengan meminta jumlah anak kelinci, kemudian menerima input berat masing-masing kelinci ke dalam array. Setelah semua data berat dimasukkan, program melakukan pencarian nilai ekstrem dengan membandingkan setiap berat kelinci untuk menentukan nilai minimum dan maksimum. Hasil akhirnya adalah berat terkecil dan terbesar yang ditampilkan dalam format dua angka desimal. Program ini berguna untuk analisis sederhana data numerik dalam bentuk array.

2. Guided 2

Source Code :

```
package main

import (
    "fmt"
)

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

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

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

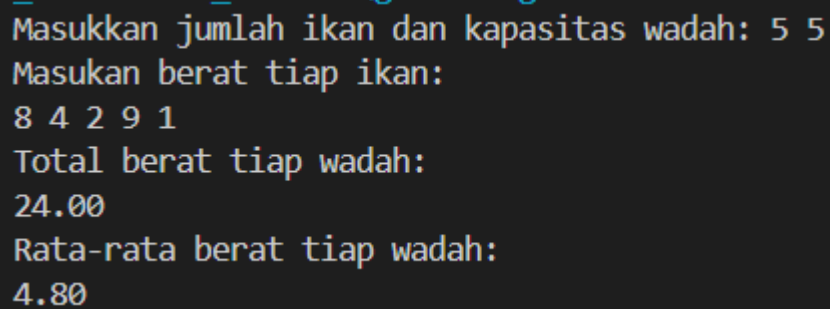
    for i := 0; i < x; i++ {
        indeksWadah := i / y
        totalBeratWadah[indeksWadah] += berat[i]
    }

    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 :



```
Masukkan jumlah ikan dan kapasitas wadah: 5 5  
Masukan berat tiap ikan:  
8 4 2 9 1  
Total berat tiap wadah:  
24.00  
Rata-rata berat tiap wadah:  
4.80
```

Deskripsi Program :

Program ini digunakan untuk menghitung total berat dan rata-rata berat ikan pada setiap wadah berdasarkan jumlah ikan dan kapasitas maksimal wadah yang ditentukan oleh pengguna. Program meminta input jumlah ikan, kapasitas wadah, dan berat masing-masing ikan. Data berat ikan dimasukkan ke dalam wadah secara berurutan berdasarkan kapasitas yang tersedia. Program kemudian menghitung total berat ikan dalam setiap wadah dan menampilkan rata-rata berat ikan per wadah. Program ini berguna untuk simulasi distribusi beban pada wadah secara efisien.

III. UNGUIDED

1. 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.

Source Code :

```
package main

import (
    "fmt"
)

type arrBalita [100]float64

func hitungMinMax(arrBerat arrBalita, n int, bMin, bMax *float64)
{
    *bMin = arrBerat[0]
    *bMax = arrBerat[0]

    for i := 1; i < n; i++ {
        if arrBerat[i] < *bMin {
            *bMin = arrBerat[i]
        }
        if arrBerat[i] > *bMax {
            *bMax = arrBerat[i]
        }
    }
}

func rerata(arrBerat arrBalita, n int) float64 {
    var total float64
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}
```

```
}

func main() {
    var n int
    var arr arrBalita
    var min, max float64

    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)

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

    hitungMinMax(arr, n, &min, &max)
    fmt.Printf("Berat balita minimum: %.2f kg\n", min)
    fmt.Printf("Berat balita maksimum: %.2f kg\n", max)

    average := rerata(arr, n)
    fmt.Printf("Rerata berat balita: %.2f kg\n", average)
}
```

Screenshot Program :

```
_2311102029_Modul10\unguided1.go"
Masukkan banyak data berat balita: 4
Masukkan berat balita:
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\ACER\Downloads\Semester 3\Lap
```

Deskripsi Program :

Program ini digunakan untuk menganalisis data berat balita dengan menghitung nilai minimum, maksimum, dan rata-rata dari sekumpulan data berat yang dimasukkan oleh pengguna. Program menerima jumlah data berat balita yang akan diolah, kemudian meminta input berat masing-masing balita. Dengan menggunakan fungsi `hitungMinMax`, program menentukan berat balita terkecil dan terbesar, sementara fungsi `rerata` digunakan untuk menghitung rata-rata berat balita. Hasil akhirnya menampilkan berat minimum, maksimum, dan rata-rata berat balita dalam satuan kilogram dengan format dua angka desimal. Program ini berguna untuk mempermudah analisis statistik sederhana pada data berat balita.