

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL 10

PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Disusun Oleh :

PRIESTY AMEILIANA MAULIDAH / 2311102175

IF-11-05

Dosen Pengampu :

ARIF AMRULLOH, S.KOM.,M.KOM

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

1.) DASAR TEORI

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.

2. Pencarian Nilai Ekstrim pada Array Bertipe Data Dasar Misalnya terdefinisi sebuah array of integer dengan kapasitas 2023, dan array berisi sejumlah N bilangan bulat, kemudian pencarian nilai terkecil dilakukan pada array tersebut. P

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.

2.) GUIDED

Soal Studi Case

1)

Sourcecode

```
modul 7 > -60 guided 1.go > ...
1  package main
2  import "fmt"
3  type waktu struct {
4      jam, menit, detik int
5  }
6
7  func main(){
8      var wParkir, wPulang, durasi waktu // awal = 2 45 14
9      var dParkir, dPulang, lParkir int // pulang = 4 30 12
10
11     fmt.Scan(&wParkir.jam, &wParkir.menit, &wParkir.detik)
12     fmt.Scan(&wPulang.jam, &wPulang.menit, &wPulang.detik)
13
14     dParkir = wParkir.detik + wParkir.menit*60 + wParkir.jam*3600 // konversimke detik
15     dPulang = wPulang.detik + wPulang.menit*60 + wPulang.jam*3600 // detik
16     lParkir = dPulang - dParkir // detik dari pulang-datang
17
18     durasi.jam = lParkir / 3600
19     durasi.menit = lParkir % 3600 / 60
20     durasi.detik = lParkir % 3600 % 60 // 17
21     fmt.Printf("Lama parkir: %d jam %d menit %d detik", durasi.jam, durasi.menit, durasi.detik)
22 }
```

Screenshoot Output

```
PROBLEMS 77 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 7\guided 1.go"
2 45 14
4 30 12
Lama parkir: 1 jam 44 menit 58 detik
PS D:\1-Priesty AM\alpro 3 semester 3> |
```

Deskripsi Program

Program menghitung lama waktu parkir

- Input : waktu masuk dan keluar (jam menit detik)
- Proses : konversi ke detik, hitung selisih, konversi balik ke format jam-menit-detik
- Output : menampilkan durasi parkir

Hasil output :

- Masuk : 2 45 14
- Keluar : 4 30 12
- Program hitung : selisih waktu dalam format jam : menit : detik

GUIDED

Soal Studi Case

2.

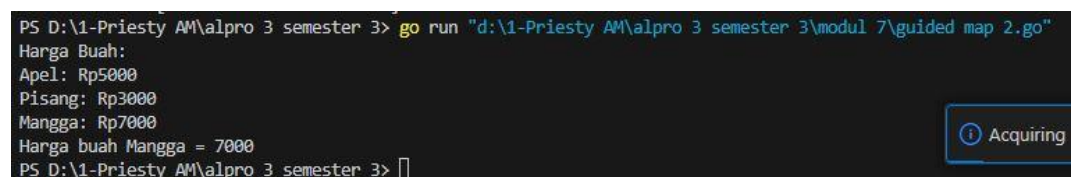
Sourcecode



```
guided slice 2.go 1  guided map 2.go 1 X  unguided array 1.go 1  unguided array 4.go 1

modul 7 > guided map 2.go > ...
1 package main
2 import (
3     "fmt"
4 )
5
6 func main() {
7     // Membuat map dengan nama buah sebagai kunci dan harga sebagai nilai
8     hargaBuah := map[string]int{
9         "Apel": 5000,
10        "Pisang": 3000,
11        "Mangga": 7000,
12    }
13
14    // Menampilkan harga dari setiap buah
15    fmt.Println("Harga Buah:")
16    for buah, harga := range hargaBuah {
17        fmt.Printf("%s: Rp%d\n", buah, harga)
18    }
19
20    fmt.Print("Harga buah Mangga = ", hargaBuah["Mangga"])
21 }
```

Screenshoot Output



```
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 7\guided map 2.go"
Harga Buah:
Apel: Rp5000
Pisang: Rp3000
Mangga: Rp7000
Harga buah Mangga = 7000
PS D:\1-Priesty AM\alpro 3 semester 3>
```

Deskripsi Program

Program mengelola daftar harga buah

- Data : menyimpan nama dan harga buah dalam map
- Proses : menampilkan semua harga buah dengan perulangan
- Output : menampilkan daftar lengkap dan harga spesifik buah mangga

Data harga buah :

- Apel = Rp5.000
- Pisang = Rp3.000

- Mangga = Rp7.000

GUIDED

Soal Studi Case

3.

Sourcecode

```
modul 7 > go guided slice 2.go > main
1  package main
2
3  import (
4      "fmt"
5  )
6
7  // Fungsi untuk mengecek apakah nama sudah ada di dalam slice
8  func sudahAda(daftarTeman []string, nama string) bool {
9      for _, teman := range daftarTeman {
10         if teman == nama {
11             return true
12         }
13     }
14     return false
15 }
16
17 func main() {
18     // Slice awal untuk daftar teman dengan beberapa data
19     daftarTeman := []string{"Andi", "Budi", "Cici"}
20
21     // Nama-nama baru yang ingin ditambahkan
22     namaBaru := []string{"Dewi", "Budi", "Eka"}
23
24     // Menambahkan nama baru hanya jika belum ada di daftar
25     for _, nama := range namaBaru {
26         if !sudahAda(daftarTeman, nama) {
27             daftarTeman = append(daftarTeman, nama)
28         } else {
29             fmt.Println("Nama", nama, "sudah ada dalam daftar.")
30         }
31     }
32
33     // Menampilkan daftar teman akhir
34     fmt.Println("Daftar Teman:", daftarTeman)
35 }
36
```

Screenshoot Output

```
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 7\guided slice 2.go"
Nama Budi sudah ada dalam daftar.
Daftar Teman: [Andi Budi Cici Dewi Eka]
PS D:\1-Priesty AM\alpro 3 semester 3> 
```

Deskripsi Program

Program mengelola daftar nama

- Input : daftar nama awal (andi, budi, cici) dan nama baru (dewi, budi, eka)
- Proses : mengecek setiap nama baru, hanya menambahkan yang belum ada
- Output : pesan jika nama duplikasi dan daftar final tanpa duplkasi

Cara kerja : program memeriksa setiap nama baru, menambahkannya jika unik, dan memberikan peingatan jika nama sudah ada dalam daftar

Contoh : budi tidak ditambahkan karena sudah ada, dewi dan eka ditambahkan karena baru

3.) UNGUIDED

Modul 10

Soal Studi Case

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.

Sourcecode

```
modul 8 > -go unguided max.min 1.go > main
1 package main
2
3 import (
4     "fmt"
5 )
6
7 func main() {
8     var N int
9     var weights [1000]float64
10
11     fmt.Print("Masukkan jumlah anak kelinci: ")
12     fmt.Scan(&N)
13
14     for i := 0; i < N; i++ {
15         fmt.Printf("Masukkan berat anak kelinci ke-%d: ", i+1)
16         fmt.Scan(&weights[i])
17     }
18
19     minWeight := weights[0]
20     maxWeight := weights[0]
21
22     for i := 1; i < N; i++ {
23         if weights[i] < minWeight {
24             minWeight = weights[i]
25         }
26         if weights[i] > maxWeight {
27             maxWeight = weights[i]
28         }
29     }
30
31     fmt.Printf("Berat kelinci terkecil: %.2f\n", minWeight)
32     fmt.Printf("Berat kelinci terbesar: %.2f\n", maxWeight)
33 }
34
```

Screenshoot Output

```
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 8\unguided max.min 1.go"
Masukkan jumlah anak kelinci: 3
Masukkan berat anak kelinci ke-1: 2.5
Masukkan berat anak kelinci ke-2: 3.8
Masukkan berat anak kelinci ke-3: 1.9
Berat kelinci terkecil: 1.90
Berat kelinci terbesar: 3.80
```

Deskripsi Program

Program ini merupakan aplikasi sederhana untuk mencari berat kelinci terkecil dan terbesar. cara kerjanya dimulai dengan memasukkan jumlah kelinci dan berat masing-masing kelinci. program menggunakan algoritma pencarian nilai minimum dan maksimum yang membandingkan setiap data berat untuk menemukan nilai terkecil dan terbesar. Misalnya, jika

dimasukkan tiga data kelinci dengan berat 2.5kg,3.8kg,dan 1.9kg, maka program akan menampilkan hasil terkecil 1.90kg dan terbesar 3.80 kg.

UNGUIDED

Soal Studi Case

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.

Sourcecode

```
modul 8 > -go unguided max.min 2.go > main
1 package main
2
3 import (
4     "fmt"
5 )
6
7 func main() {
8     var x, y int
9     fmt.Scan(&x, &y)
10
11     weights := make([]float64, x)
12     for i := 0; i < x; i++ {
13         fmt.Scan(&weights[i])
14     }
15
16     totalWeights := make([]float64, (x+y-1)/y)
17     for i := 0; i < x; i++ {
18         totalWeights[i/y] += weights[i]
19     }
20
21     for _, total := range totalWeights {
22         fmt.Printf("%.2f ", total)
23     }
24     fmt.Println()
25
26     averageWeights := make([]float64, len(totalWeights))
27     for i, total := range totalWeights {
28         averageWeights[i] = total / float64(y)
29         if i == len(totalWeights)-1 && x%y != 0 {
30             averageWeights[i] = total / float64(x%y)
31         }
32     }
33
34     for _, avg := range averageWeights {
35         fmt.Printf("%.2f ", avg)
36     }
37     fmt.Println()
38 }
39
```

Screenshoot Output

```
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 8\unguided max.min 2.go"
10 3
1.1 2.2 3.3. 4.4 5.5. 6.6 7.7 8.8 9.9 10.10
6.60 16.50 26.40 10.10
2.20 5.50 8.80 10.10
PS D:\1-Priesty AM\alpro 3 semester 3>
```

Deskripsi Program

Program ini bekerja dengan menerima input berupa dua angka : x dan y. X mewakili jumlah data yang ada, sedangkan y mewakili ukuran setiap kelompok. program kemudian menerima x buah nilai berat yang disimpan dalam slice yang disebut weights. Program membuat slice totalWeights untuk menyimpan total berat setiap kelompok. program menghitung total

berat untuk setiap kelompok dengan ukuran y dan menampilkan hasil total setiap kelompok dalam format 2 angka desimal. Program juga membuat slice averageweights untuk menyimpan rata-rata berat setiap kelompok. rata-rata dihitung dengan membagi total berat kelompok dengan ukuran kelompok (y). Untuk kelompok terakhir, jika jumlah data tidak habis dibagi y, rata-rata dihitung dengan membagi total dengan sisa pembagian (x%y). hasil rata-rata setiap kelompok ditampilkan dalam format 2 angka desimal contoh pengguna input

Input :10 3 1.1 2.2 3.3 4.4 5.5 6.6 7.7 8.8 9.9 10.10

Output :6.60 16.50 26.40 10.10 2.20 5.50 8.80 10.10

UNGUIDED

Soal Studi Case

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 */

    ...
}
```

Sourcecode

```
modul 8 > go unguided max.min 3.go > main
1  package main
2
3  import (
4      "fmt"
5  )
6
7  func main() {
8      var n int
9      fmt.Print("Masukan banyak data berat balita: ")
10     fmt.Scan(&n)
11
12     beratBalita := make([]float64, n)
13
14     for i := 0; i < n; i++ {
15         fmt.Printf("Masukan berat balita ke-%d: ", i+1)
16         fmt.Scan(&beratBalita[i])
17     }
18
19     min := beratBalita[0]
20     max := beratBalita[0]
21     total := 0.0
22
23     for _, berat := range beratBalita {
24         if berat < min {
25             min = berat
26         }
27         if berat > max {
28             max = berat
29         }
30         total += berat
31     }
32
33     average := total / float64(n)
34
35     fmt.Printf("Berat balita minimum: %.2f kg\n", min)
36     fmt.Printf("Berat balita maksimum: %.2f kg\n", max)
37     fmt.Printf("Rerata berat balita: %.2f kg\n", average)
38 }
```

Screenshoot Output

```
PS D:\1-Priesty AM\alpro 3 semester 3> go run "d:\1-Priesty AM\alpro 3 semester 3\modul 8\unguided max.min 3.go"
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
PS D:\1-Priesty AM\alpro 3 semester 3> |
```

Deskripsi Program

Program ini digunakan untuk menganalisis data berat badan balita program meminta pengguna memasukkan jumlah data dan nilai berat badan balita. Setelah data dimasukkan, program akan melakukan perhitungan untuk mendapatkan berat terendah (minimum), tertinggi (maksimum), dan rata-rata. Hasil perhitungan ditampilkan dalam format dua angka desimal dengan satuan kilogram. contohnya, dari 4 data berat yang dimasukkan (5.3 kg, 6.2 kg, 4.1 kg, 9.9 kg), program akan menampilkan berat minimum 4.10 kg, maksimum 9.90 kg, dan rata-rata 6.38 kg