# LAPORAN PRAKTIKUM ALGORITMA DAN PEMROGRAMAN 2 MODUL X PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Oleh:

ANISSA FAUZIA ISYANTI

2311102219

S1IF-11-02

**S1 TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**2024**

## 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.

Ide algortima 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 algortima tersebut berakhir adalah nilai maksimum yang dicari. Adapun algoritmanya secara umum adalah sebagai barikut:

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 ekstrim tersebut dengan data yang dicek)
3. Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid.

A white sheet with black text

Description automatically generated

1. 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.

A computer code with black text

Description automatically generated

1. 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.

A screenshot of a computer code

Description automatically generated

Apabila diperhatikan potongan program di atas, maka kita akan memperoleh IPK tertinggi, tetapi kita tidak memperoleh identitias mahasiswa dengan IPK tertinggi tersebut. Maka seperti penjelasan yang sudah diberikan sebelumnya, maka pencarian yang dilakukan bisa mengembalikan indeks makasiswa dengan IPK tertinggi tersebut. Berikut ini adalah modifikasinya!

A screenshot of a computer code

Description automatically generated

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

## GUIDED

### Source Code

|  |
| --- |
| 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 dengan 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: %5.2f\n", min)  fmt.Printf("Berat terbesar: %5.2f\n", max)  } |

Screenshot Program

A screen shot of a computer

Description automatically generated

Program ini digunakan untuk menentukan berat terkecil dan terbesar dari sejumlah anak kelinci. Pengguna diminta untuk memasukkan jumlah anak kelinci dan berat masing-masing anak kelinci. Program akan membaca data berat tersebut, kemudian menghitung nilai minimum dan maksimum dari berat yang diinputkan. Hasilnya ditampilkan dengan dua angka di belakang koma. Program ini menggunakan array untuk menyimpan berat dan perulangan untuk membandingkan setiap nilai berat dengan nilai minimum dan maksimum sementara.

### 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("Masukkan berat tiap ikan: ")  for i := 0; i < x; i++ {  fmt.Scan(&berat[i])  }  jumlahwadah := (x + y - 1) / y //Pembulatan ke atas jika x tidak habis diabgi y  totalBeratWadah := make([]float64, jumlahwadah)  for i := 0; i < x; i++ {  indeksWadah := i / y  totalBeratWadah[indeksWadah] += berat[i]  }  //Output total berat tiap wadah  fmt.Println("Total berat tiap wadah: ")  for \_, total := range totalBeratWadah {  fmt.Printf("%.2f ", total)  }  fmt.Println()  //Output rata-rata berat tiap wadah  fmt.Println("Rata-rata berat tiap wadah: ")  for \_, total := range totalBeratWadah {  rataRata := total / float64(y)  fmt.Printf("%.2f ", rataRata)  }  fmt.Println()  } |

Screenshot Program

A screen shot of a computer

Description automatically generated

Program menghitung total berat dan rata-rata berat ikan yang dimasukkan ke dalam sejumlah wadah dengan kapasitas tertentu. Pengguna diminta memasukkan jumlah ikan, kapasitas wadah, serta berat masing-masing ikan. Program kemudian menghitung berapa banyak wadah yang dibutuhkan dengan pembulatan ke atas jika jumlah ikan tidak habis dibagi kapasitas wadah. Berat tiap ikan dimasukkan ke wadah berdasarkan urutan, dan total berat tiap wadah dihitung. Program menampilkan total berat dan rata-rata berat di setiap wadah dalam format desimal dua angka di belakang koma.

## UNGUIDED

### Source Code

|  |
| --- |
| package main  import (  "fmt"  )  type arrBalita [100]float64  var jml\_balita int  func hitungMinMax(arrBerat arrBalita, bMin, bMax \*float64) {  \*bMin = arrBerat[0]  \*bMax = arrBerat[0]  for i := 0; i < jml\_balita; i++ {  if arrBerat[i] < \*bMin {  \*bMin = arrBerat[i]  }  if arrBerat[i] > \*bMax {  \*bMax = arrBerat[i]  }  }  }  func Rerata(arrBerat arrBalita) float64 {  var jumlah float64  for \_, value := range arrBerat {  jumlah += value  }  return jumlah / float64(jml\_balita)  }  func main() {  var balitas arrBalita  var min, max float64  fmt.Print("Masukan Banyak data berat balita: ")  fmt.Scan(&jml\_balita)  for i := 0; i < jml\_balita; i++ {  fmt.Print("Masukan berat balita ke-", i+1, ": ")  fmt.Scan(&balitas[i])  }  hitungMinMax(balitas, &min, &max)  fmt.Printf("Berat balita minimum: %.2f kg\n", min)  fmt.Printf("Berat balita maksimum: %.2f kg\n", max)  fmt.Printf("Rerata berat balita: %.2f kg\n", Rerata(balitas))  } |

Screenshot Program

A screen shot of a computer

Description automatically generated

Program ini menghitung nilai minimum, maksimum, dan rata-rata dari data yang dimasukkan. Pengguna diminta untuk memasukkan jumlah balita dan berat badan masing-masing balita. Data yang dimasukkan akan diproses menggunakan fungsi hitungMinMax untuk mencari berat minimum dan maksimum, serta fungsi Rerata untuk menghitung rata-rata berat badan. Hasilnya ditampilkan berat balita minimum, maksimum, dan rata-rata dengan dua angka di belakang koma.