

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XI
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Disusun Oleh :

M. Haidar Akhbiyani / 2311102276

S1-IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Praktikum kali ini mencakup beberapa konsep penting dalam pemrograman, khususnya algoritma untuk pencarian nilai ekstrem dan pengolahan data dalam array. Array digunakan sebagai struktur data statis untuk menyimpan data dalam jumlah tetap dengan tipe homogen, seperti mencatat berat balita, berat ikan, atau atribut mahasiswa. Pencarian nilai ekstrem, seperti nilai minimum atau maksimum, dilakukan dengan iterasi yang membandingkan setiap elemen array dengan nilai ekstrem yang ditemukan sebelumnya. Selain itu, penghitungan rata-rata dilakukan dengan menjumlahkan semua elemen dalam array dan membaginya dengan jumlah elemen untuk analisis data numerik. Validasi input juga digunakan untuk memastikan bahwa data yang dimasukkan sesuai dengan batas yang diperbolehkan, seperti membatasi jumlah elemen array atau nilai input yang sesuai dengan logika program.

Program ini juga menerapkan prinsip pemrograman modular dengan memecahnya menjadi fungsi-fungsi spesifik, seperti `hitungMinMax` dan `rerata`, untuk memudahkan pembacaan dan pemeliharaan kode. Penggunaan tipe data kompleks, seperti `struct`, memungkinkan pengelolaan data yang memiliki atribut majemuk, seperti data mahasiswa yang terdiri dari nama, NIM, kelas, jurusan, dan IPK. Secara keseluruhan, pemrograman berbasis iterasi dan penggunaan array atau wadah untuk menyimpan dan mengolah data adalah teknik-teknik utama yang digunakan untuk menyelesaikan masalah-masalah yang ada dalam program ini.

II. GUIDED

1. Guided 1

Sourcecode

```
package main

import "fmt"

// Mendeklarasikan tipe data array arrInt dengan panjang 2023
type arrInt [2023]int

// Fungsi untuk mencari elemen terkecil dalam array
func terkecil(tabInt arrInt, n int) int {
    var min int = tabInt[0]
    var j int = 1
    for j < n {
        if min > tabInt[j] {
            min = tabInt[j]
        }
        j = j + 1
    }
    return min
}

// Fungsi main
func main() {
    var n int
    var tab arrInt

    // Meminta input jumlah elemen array
    fmt.Print("Masukkan jumlah elemen (maks 2023): ")
    fmt.Scan(&n)

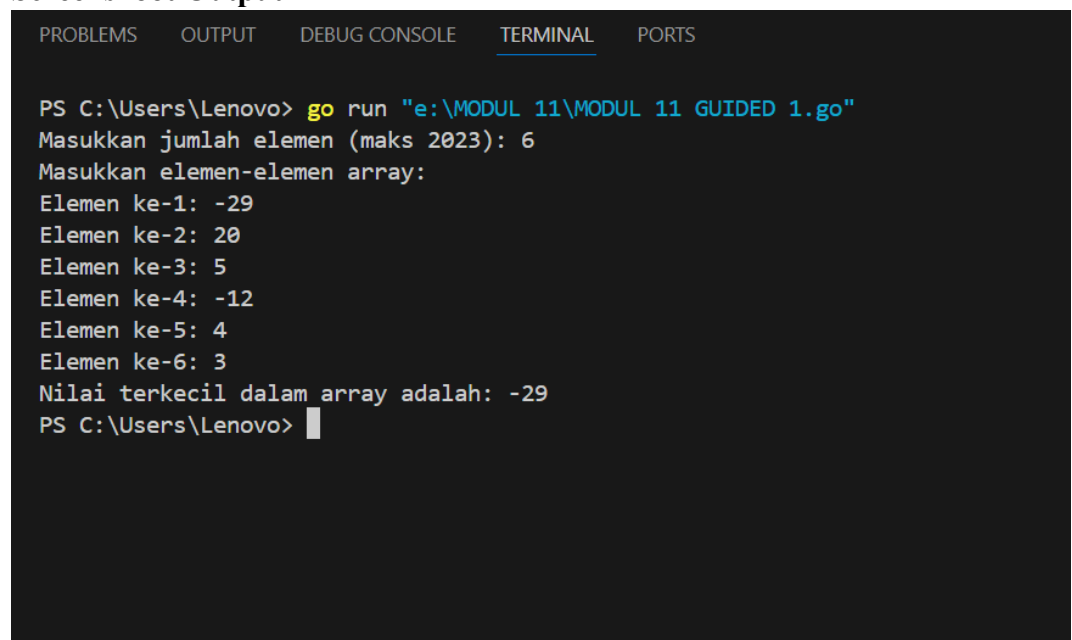
    // Validasi input jumlah elemen
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah elemen harus antara 1 dan 2023.")
        return
    }

    // Memasukkan elemen-elemen array
    fmt.Println("Masukkan elemen-elemen array:")
    for i := 0; i < n; i++ {
        fmt.Print("Elemen ke-", i+1, ": ")
        fmt.Scan(&tab[i])
    }
}
```

```
// Memanggil fungsi terkecil untuk menemukan nilai terkecil
minVal := terkecil(tab, n)

// Menampilkan nilai terkecil
fmt.Println("Nilai terkecil dalam array adalah:", minVal)
}
```

Screenshot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Lenovo> go run "e:\MODUL 11\MODUL 11 GUIDED 1.go"
Masukkan jumlah elemen (maks 2023): 6
Masukkan elemen-elemen array:
Elemen ke-1: -29
Elemen ke-2: 20
Elemen ke-3: 5
Elemen ke-4: -12
Elemen ke-5: 4
Elemen ke-6: 3
Nilai terkecil dalam array adalah: -29
PS C:\Users\Lenovo> █
```

Deskripsi Program

Kode diatas merupakan hasil dari implementasi progrm bahasa GO yang memliki tujuan untuk mencari atau menentukan suatu elemen terkecil yang ada pada array. Program ini mendefinisikan tipe data arrInt sebagai array dengan panjang maksimum 2023, lalu menggunakan fungsi terkecil untuk menentukan elemen terkecil dalam array berdasarkan panjang yang ditentukan pengguna.

2. Guided 2

Sourcecode

```
package main

import "fmt"

// Definisi struct mahasiswa dengan atribut nama, nim, kelas, jurusan,
dan ipk
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk                        float64
}

// Definisi tipe data array mahasiswa dengan kapasitas maksimal 2023
type arrMhs [2023]mahasiswa

// Fungsi untuk mencari IPK tertinggi dalam array mahasiswa
func ipk(T arrMhs, n int) float64 {
    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
}

// Fungsi main untuk mengisi data mahasiswa dan mencari IPK tertinggi
func main() {
    var n int
    var dataMhs arrMhs

    // Meminta input jumlah mahasiswa
    fmt.Print("Masukkan jumlah mahasiswa (maks 2023): ")
    fmt.Scan(&n)

    // Validasi jumlah mahasiswa yang dimasukkan
    if n < 1 || n > 2023 {
        fmt.Println("Jumlah mahasiswa harus antara 1 dan 2023.")
        return
    }
}
```

```
// Mengisi data mahasiswa
for i := 0; i < n; i++ {
    fmt.Printf("\nMasukkan data mahasiswa ke-%d\n", i+1)
    fmt.Print("Nama: ")
    fmt.Scan(&dataMhs[i].nama)
    fmt.Print("NIM: ")
    fmt.Scan(&dataMhs[i].nim)
    fmt.Print("Kelas: ")
    fmt.Scan(&dataMhs[i].kelas)
    fmt.Print("Jurusan: ")
    fmt.Scan(&dataMhs[i].jurusan)
    fmt.Print("IPK: ")
    fmt.Scan(&dataMhs[i].ipk)
}

// Mencari dan menampilkan IPK tertinggi
tertinggi := ipk(dataMhs, n)
fmt.Printf("\nIPK tertinggi dari %d mahasiswa adalah: %.2f\n", n,
tertinggi)
}
```

Screenshoot Output

```
PS C:\Users\Lenovo> go run "e:\MODUL 11\MODUL 11 GUIDED 2.go"
Masukkan jumlah mahasiswa (maks 2023): 2

Masukkan data mahasiswa ke-1
Nama: Haider
NIM: 2311102276
Kelas: IF-06
Jurusan: Informatika
IPK: 4,00

Masukkan data mahasiswa ke-2
Nama: NIM: Yani
Kelas: IF-06
Jurusan: DKV
IPK: 3,80

IPK tertinggi dari 2 mahasiswa adalah: 4.00
```

Deskripsi Program

Program diatas berguna untuk mendata mahasiswa dan juga untuk mencari IPK yang tertinggi diantara mahasiswa satu sama lain,didalamnya mempunyai atribut seperti nama,nim,kelas,jurusan dan IPK.Didalamnya

ada tipe data array arrMhs yang berguna untuk menyimpan data hingga 2023 mahasiswa.

I. UNGUIDED

1. Unguided 1

Sourcecode

```
package main

import "fmt"

const NMAX int = 1000
type arrKelinci [NMAX]float64

func isi_data_kelinci(arrBerat *arrKelinci) {
    var n int
    fmt.Print("Masukkan berat kelinci: ")
    fmt.Scan(&n)
    if(n > NMAX) {
        fmt.Print("Maksimal jumlah data ", NMAX, "\n")
        return
    }
    for i:=0; i<n; i++ {
        fmt.Print("Masukkan berat kelinci ke-", i+1, ": ")
        fmt.Scan(&arrBerat[i])
    }
}

func cetak_berat_min(arrBerat arrKelinci) {
    min := arrBerat[0]
    i := 1

    for arrBerat[i] != 0 && i < 100 {
        if(min > arrBerat[i]) {
            min = arrBerat[i]
        }
        i++
    }
    fmt.Print("Berat kelinci minimum: ", min, " kg\n")
}

func cetak_berat_max(arrBerat arrKelinci) {
    max := arrBerat[0]
    i := 1

    for arrBerat[i] != 0 && i < 100 {
```

```

        if(max < arrBerat[i]) {
            max = arrBerat[i]
        }
        i++
    }
    fmt.Print("Berat kelinci maximum: ", max, " kg\n")
}

func main() {

    var arrBerat arrKelinci
    isi_data_kelinci(&arrBerat)
    cetak_berat_min(arrBerat)
    cetak_berat_max(arrBerat)

}

```

Screenshoot Output

```

PS C:\Users\Lenovo> go run "e:\MODUL 11\UNGUIDED 1 MODUL 11.go"
Masukkan berat kelinci: 6
Masukkan berat kelinci ke-1: 2
Masukkan berat kelinci ke-2: 8
Masukkan berat kelinci ke-3: 4
Masukkan berat kelinci ke-4: 3
Masukkan berat kelinci ke-5: 5
Masukkan berat kelinci ke-6: 3,5
Berat kelinci minimum: 2 kg
Berat kelinci maximum: 8 kg
PS C:\Users\Lenovo>

```

Deskripsi Program

Program diatas memiliki kegunaan untuk mengelola berat kelinci, dimana didalamnya menggunakan array statis Fungsi isi_data_kelinci mengisi array dengan input pengguna, sementara cetak_berat_min dan cetak_berat_max menghitung serta menampilkan berat minimum dan maksimum. Fungsi-fungsi ini dipanggil dalam main untuk memproses dan menganalisis data yang dimasukkan.

2. Unguided 2

Sourcecode

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var x, y int
    fmt.Print("Inputkan jumlah ikan yang mau dijual (x) dan jumlah ikan per wadah (y): ")
    fmt.Scan(&x, &y)
    fmt.Println("Inputkan berat ikan:")
    berat := make([]float64, x)
    for i := 0; i < x; i++ {
        fmt.Scan(&berat[i])
    }

    wadahHitung := int(math.Ceil(float64(x) / float64(y)))
    wadahTotals := make([]float64, wadahHitung)

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

    wadahRatarata := make([]float64, wadahHitung)
    for i := 0; i < wadahHitung; i++ {
        if (i+1)*y <= x {
            wadahRatarata[i] = wadahTotals[i] / float64(y)
        } else {
            wadahRatarata[i] = wadahTotals[i] / float64(x%y)
        }
    }

    fmt.Println("Total ikan di wadah:")
    for _, total := range wadahTotals {
        fmt.Printf("%.2f ", total)
    }
    fmt.Println()

    fmt.Println("Rata-rata berat ikan di setiap wadah:")
}
```

```
for _, avg := range wadahRatarata {  
    fmt.Printf("%.2f ", avg)  
}  
fmt.Println()  
}
```

Screenshoot Output

```
Inputkan jumlah ikan yang mau dijual (x) dan jumlah ikan per wadah (y): 8 7  
Inputkan berat ikan:  
2  
3  
2,5  
4  
1,5  
1,8  
Total ikan di wadah:  
22.00 1.00  
Rata-rata berat ikan di setiap wadah:  
3.14 1.00  
PS C:\Users\Lenovo> █
```

Deskripsi Program

Program ini menghitung total dan rata-rata berat ikan yang dibagi ke dalam sejumlah wadah. Pengguna diminta memasukkan jumlah ikan (x), kapasitas ikan per wadah (y), serta berat masing-masing ikan. Program menghitung jumlah wadah yang diperlukan menggunakan fungsi `math.Ceil`, lalu mendistribusikan berat ikan ke setiap wadah. Total berat ikan per wadah disimpan dalam array `wadahTotals`, dan rata-rata berat ikan per wadah dihitung dalam array `wadahRatarata`. Hasil akhir berupa total berat dan rata-rata berat ikan di setiap wadah ditampilkan dengan format dua desimal.

3. Unguided 3

Sourcecode

```
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 {
    total := 0.0
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}

func main() {
    var arrBerat arrBalita
    var n int
    var bMin, bMax float64

    fmt.Print("Masukkan jumlah balita (maks 100): ")
    fmt.Scan(&n)

    if n > 100 {
        fmt.Println("Jumlah balita tidak boleh lebih dari
100.")
        return
    }

    fmt.Println("Masukkan berat balita (kg):")
```

```

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

    hitungMinMax(arrBerat, n, &bMin, &bMax)

    rerataBerat := rerata(arrBerat, n)

    fmt.Printf("Berat minimum: %.2f kg\n", bMin)
    fmt.Printf("Berat maksimum: %.2f kg\n", bMax)
    fmt.Printf("Rata-rata berat: %.2f kg\n", rerataBerat)
}

```

Screenshoot Output

```

PS C:\Users\Lenovo> go run "e:\MODUL 11\UNGUIDED 3 MODUL 11.go"
Masukkan jumlah balita (maks 100): 6
Masukkan berat balita (kg):
Berat balita ke-1: 3
Berat balita ke-2: 4
Berat balita ke-3: 4,5
Berat balita ke-4: Berat balita ke-5: 3,8
Berat balita ke-6: Berat minimum: 3.00 kg
Berat maksimum: 8.00 kg
Rata-rata berat: 4.50 kg
PS C:\Users\Lenovo> 

```

Deskripsi Program

Program dalam soal unguided ini bertujuan untuk mencatat berat badan balita dalam array, lalu akan menghitung berat minimum, berat maksimum dan rata rata dari berat badan balita yang sudah di inputkan. Fungsi `hitungMinMax` berguna untuk menentukan berat balita terkecil sampai terbesar, sementara fungsi `rerata` menghitung rata-rata berat dari data yang dimasukkan.