LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL XI

PENCARIAN NILAI EXTRIM PADA HIMPUNAN DATA

Disusun Oleh :

Arjun Ahmad Santoso / 2311102211

S1IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

4.1 Ide Pencarian

Pencarian adalah proses yang sangat lazim dalam kehidupan sehari-hari. Contoh proses pencarian adalah pencarian barang dalam suatu box atau tempat lainnya. Dalam pemrograman, terdapat banyak algoritma pencarian, yang umum digunakan diantaranya adalah Sequential Search dan Binary Search. Pada modul ini digunakan algoritma Sequential Search yang mana setiap nilai pada sekumpulan nilai dibandingkan satu persatu sampai seluruh nilai telah dibandingkan.

4.2 Ide Pencarian Nilai Minimum dan Nilai Maksimum

Pencarian nilai Minimum atau Maksimum dapat dilakukan dengan membandingkan satu persatu nilai dalam sekumpulan nilai dengan nilai minimum atau maksimum terakhir. Adapun algoritma untuk mencari nilai ekstrim(nilai minimum/maksimum/yang lainnya) dari sekumpulan elemen secara umum sebagai berikut:

1. Jadikan elemen pertama sebagai nilai ekstrim dan elemen “saat ini”.
2. Bandingkan elemen “saat ini” dengan elemen berikutnya. Jadikan elemen berikutnya tersebut sebagai elemen “saat ini” dan sebagai nilai ekstrim jika elemen berikutnya tersebut memenuhi persyaratan.
3. Ulangi langkah 2 hingga semua elemen telah dicek.

II. GUIDED

1.

Soal Guided

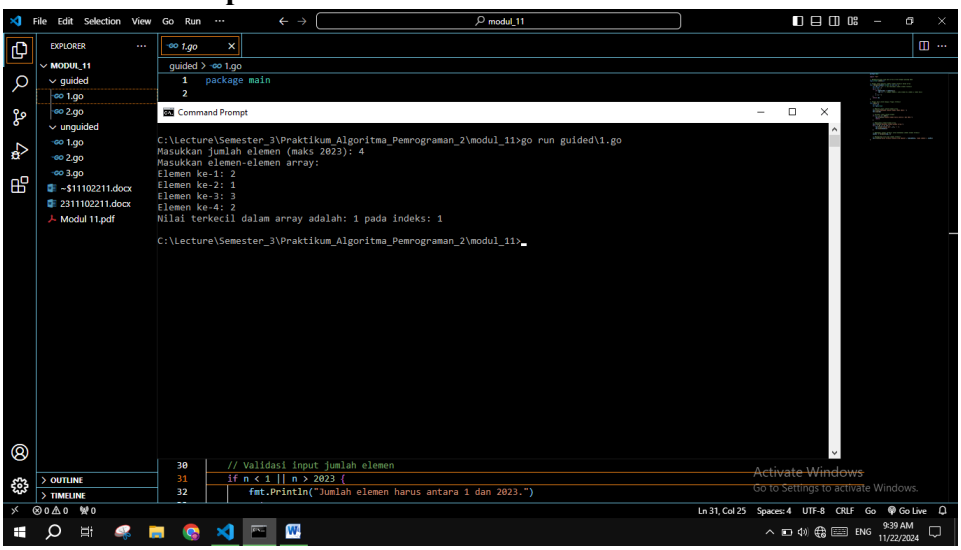
Soal Guided No. 1

Sourcecode

```
package main

import "fmt"

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

// Fungsi untuk mencari indeks elemen terkecil dalam array
func terkecil(tabInt arrInt, n int) int {
    var idx int = 0  // idx menyimpan indeks elemen terkecil
    var j int = 1
    for j < n {
        if tabInt[idx] > tabInt[j] {
            idx = j  // Simpan indeks j jika elemen di indeks
j lebih kecil
        }
        j = j + 1
    }
    return idx
}

// Fungsi main untuk menguji fungsi terkecil
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 indeks elemen
terkecil
    idxMin := terkecil(tab, n)

    // Menampilkan nilai dan indeks terkecil
    fmt.Println("Nilai terkecil dalam array adalah:",
tab[idxMin], "pada indeks:", idxMin)
}
```

Screenshoot Output



Deskripsi Program

Program di atas memuat sebuah fungsi yang dapat digunakan untuk mencari elemen dengan nilai terkecil dari suatu array.

2.

Soal Guided

Soal Guided No. 2

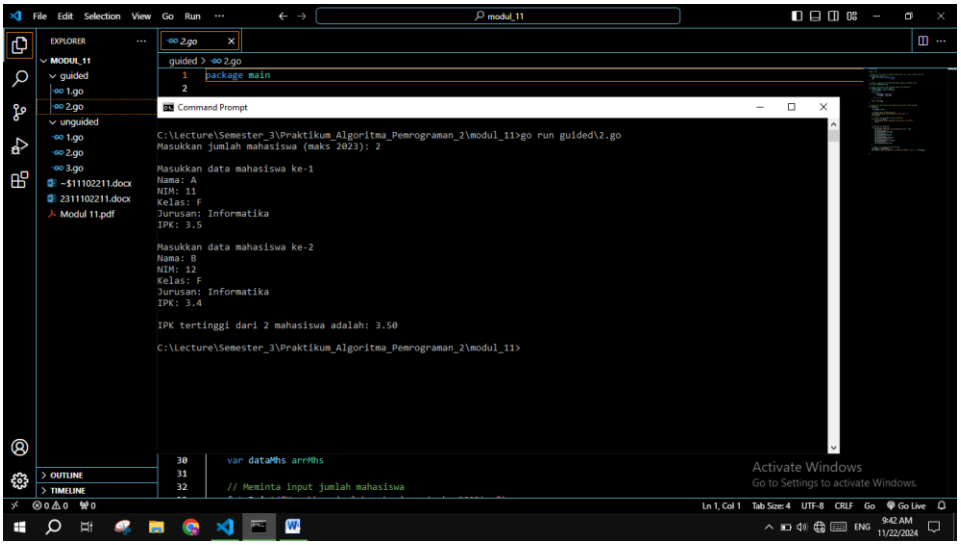
Sourcecode

```
package main

import "fmt"
```

```go
// 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



Deskripsi Program

Program di atas dapat digunakan untuk mencari nilai IPK tertinggi dari data mahasiswa.

III. UNGUIDED

1.

Soal Latihan Modul 11

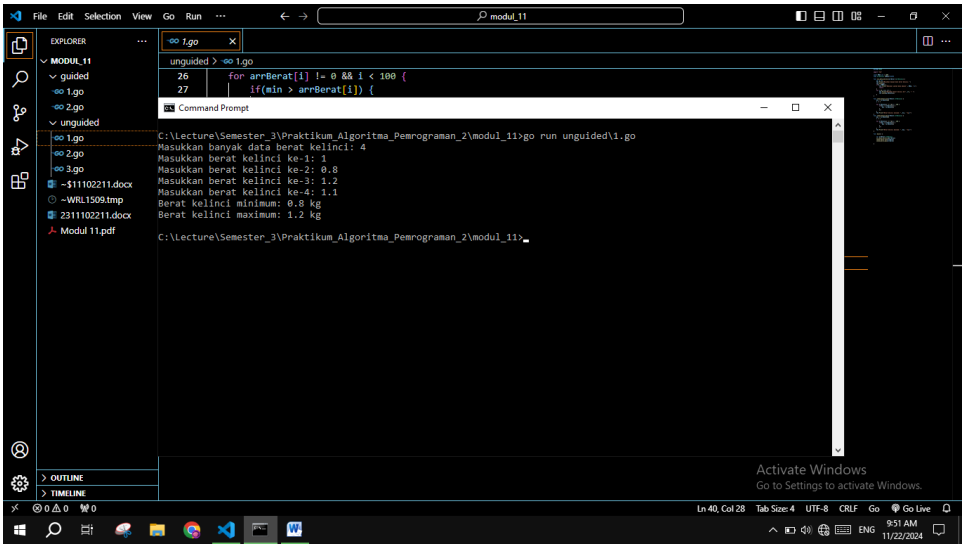
Soal Latihan Modul 11 No. 1

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 banyak data berat kelinci: ")
    fmt.Scan(&n)
    if(n > NMAX) {
        fmt.Print("Maksimal jumlah data adalah ", 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



Deskripsi Program

Program di atas memuat sebuah fungsi yang dapat digunakan untuk mencari berat kelinci minimum dan berat kelinci maksimum dari array yang memuat data berat kelinci.

2.

Soal Latihan Modul 11

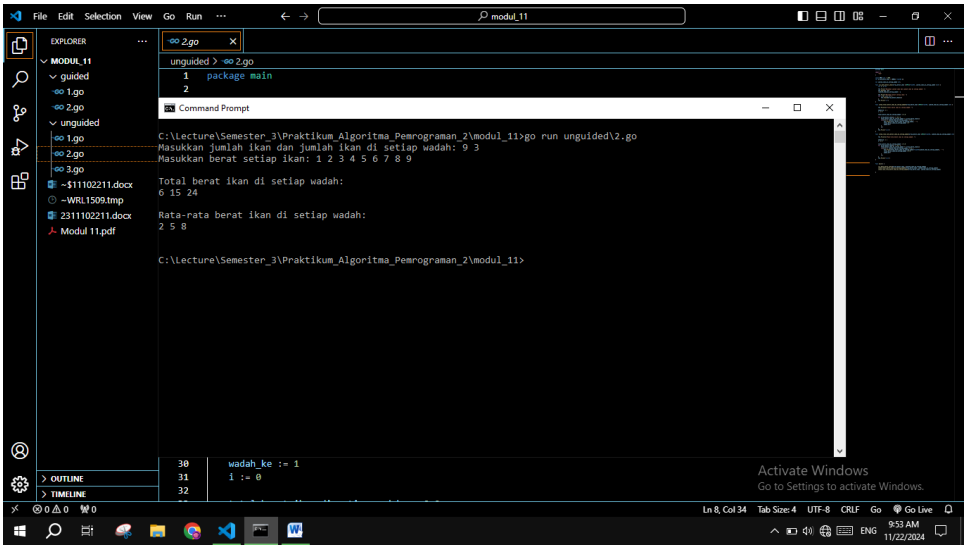
Soal Latihan Modul 11 No.2

Sourcecode

```go
package main

import (
    "fmt"
)

const NMAX int = 1000
var array_berat_ikan = [NMAX] float64 {}

var jumlah_ikan_di_setiap_wadah int;

func isi_data_berat_ikan(array_berat_ikan *[NMAX]float64,
jumlah_ikan_di_setiap_wadah *int) {
    var x, y int

    fmt.Print("Masukkan jumlah ikan dan jumlah ikan di setiap
wadah: ")
    fmt.Scan(&x, &y)
    *jumlah_ikan_di_setiap_wadah = y

    fmt.Print("Masukkan berat setiap ikan: ")
    for i:=0; i<x; i++ {
        fmt.Scan(&array_berat_ikan[i])
    }
    fmt.Print("\n")
}

func cetak_total_berat_ikan_di_setiap_wadah(array_berat_ikan
[NMAX]float64, jumlah_ikan_di_setiap_wadah int) {

    fmt.Println("Total berat ikan di setiap wadah: ")

    wadah_ke := 1
    i := 0

    total_berat_ikan_di_setiap_wadah := 0.0

    for array_berat_ikan[i] != 0.0 {
        total_berat_ikan_di_setiap_wadah +=
array_berat_ikan[i]
        if((i+1) % jumlah_ikan_di_setiap_wadah == 0) {
            fmt.Print(total_berat_ikan_di_setiap_wadah, " ")
            total_berat_ikan_di_setiap_wadah = 0
            wadah_ke++
        }
```

```
        i++
    }
    fmt.Print("\n\n")
}

func
cetak_rata_rata_berat_ikan_di_setiap_wadah(array_berat_ikan
[NMAX]float64, jumlah_ikan_di_setiap_wadah int) {

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

    wadah_ke := 1
    i := 0

    total_berat_ikan_di_setiap_wadah := 0.0
    for array_berat_ikan[i] != 0.0 {
        total_berat_ikan_di_setiap_wadah +=
array_berat_ikan[i]
        if((i+1) % jumlah_ikan_di_setiap_wadah == 0) {
            fmt.Print(total_berat_ikan_di_setiap_wadah/float64
(jumlah_ikan_di_setiap_wadah), " ")
            total_berat_ikan_di_setiap_wadah = 0.0
            wadah_ke++
        }
        i++
    }
    fmt.Print("\n\n")
}


func main() {

    isi_data_berat_ikan(&array_berat_ikan,
&jumlah_ikan_di_setiap_wadah)
    cetak_total_berat_ikan_di_setiap_wadah(array_berat_ikan,
jumlah_ikan_di_setiap_wadah)
    cetak_rata_rata_berat_ikan_di_setiap_wadah(array_berat_ika
n, jumlah_ikan_di_setiap_wadah)

}
```

Screenshoot Output



Deskripsi Program

Program di atas akan meminta pengguna untuk memasukkan jumlah ikan, jumlah ikan di setiap wadah dan berat ikan, setelah semua data yang diinputkan valid, kemudian program akan menampilkan total dan rata-rata berat ikan disetiap wadah.

3.

Soal Latihan Modul 11

Soal Latihan Modul 11 No.3

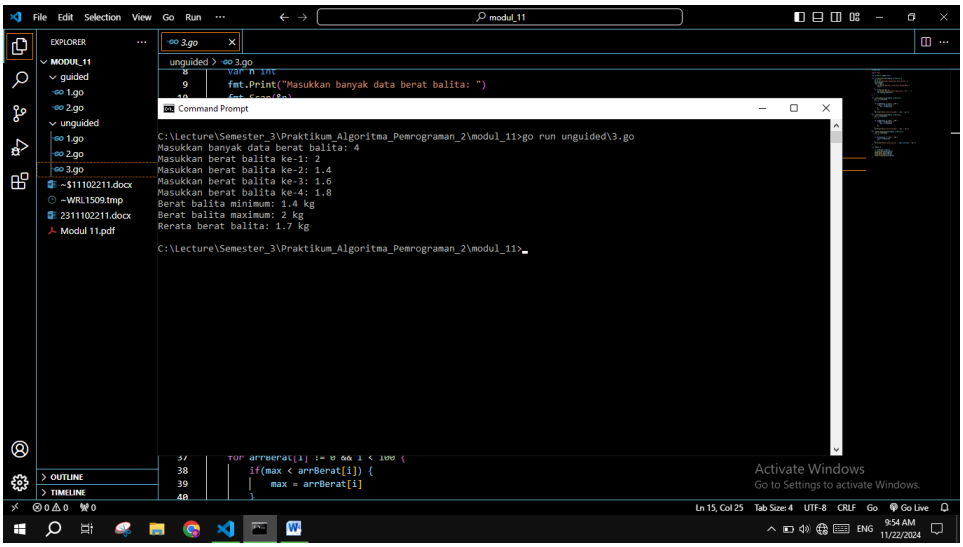
Sourcecode

```go
package main

import "fmt"

type arrBalita [100]float64

func isi_data_balita(arrBerat *arrBalita) {
    var n int
    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)
    if(n > 100) {
        fmt.Print("Maksimal jumlah data adalah 100\n")
        return
    }
    for i:=0; i<n; i++ {
        fmt.Print("Masukkan berat balita ke-", i+1, ": ")
        fmt.Scan(&arrBerat[i])
```

```
    }
}

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

    for arrBerat[i] != 0 && i < 100 {
        if(min > arrBerat[i]) {
            min = arrBerat[i]
        }
        i++
    }
    fmt.Print("Berat balita minimum: ", min, " kg\n")
}
func cetak_berat_max(arrBerat arrBalita) {
    max := arrBerat[0]
    i := 1

    for arrBerat[i] != 0 && i < 100 {
        if(max < arrBerat[i]) {
            max = arrBerat[i]
        }
        i++
    }
    fmt.Print("Berat balita maximum: ", max, " kg\n")
}
func cetak_rerata_berat(arrBerat arrBalita) {
    sum := arrBerat[0]
    i := 1

    for arrBerat[i] != 0 && i < 100 {
        sum += arrBerat[i]
        i++
    }
    fmt.Print("Rerata berat balita: ", sum/float64(i), "
kg\n")
}

func main() {

    var arrBerat arrBalita
    isi_data_balita(&arrBerat)
    cetak_berat_min(arrBerat)
    cetak_berat_max(arrBerat)
```

```
    cetak_rerata_berat(arrBerat)
}
```

Screenshoot Output



Deskripsi Program

Program di atas dapat digunakan untuk mencari berat minimum, berat maksimum, dan rerata berat dari data balita.