

**LAPORAN PRAKTIKUM ALGORITMA DAN  
PEMROGRAMAN 2**

**MODUL XI**

**PENCARIAN NILAI EXTRIM PADA HIMPUNAN DATA**



**Disusun Oleh :**

**Haposan Felix Marcel Siregar/ 2311102210**

**Dosen Pengampu :**

**Abednego Dwi Septiadi, S.Kom., M.Kom**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. DASAR TEORI

### Dasar Teori Pencarian Nilai Ekstrim

Pencarian nilai ekstrim, yaitu nilai maksimum dan minimum, merupakan salah satu operasi dasar dalam pengolahan data. Dalam konteks pemrograman, algoritma pencarian nilai ekstrim sering digunakan untuk menganalisis data dalam array atau struktur data lainnya. Proses ini penting dalam berbagai aplikasi, seperti pengolahan data statistik, analisis performa, dan pengelolaan data.

#### 1. Definisi Nilai Ekstrim

- **Nilai Maksimum:** Nilai tertinggi dalam sekumpulan data.
- **Nilai Minimum:** Nilai terendah dalam sekumpulan data.

#### 2. Algoritma Pencarian Nilai Ekstrim

Algoritma pencarian nilai ekstrim dapat diimplementasikan dengan menggunakan struktur kontrol seperti loop dan kondisi. Berikut adalah langkah-langkah umum dalam algoritma pencarian nilai maksimum:

- **Inisialisasi:** Tentukan nilai awal untuk maksimum (max) dengan elemen pertama dari array.
- **Iterasi:** Lakukan iterasi melalui elemen-elemen array mulai dari elemen kedua hingga terakhir.
- **Perbandingan:** Bandingkan setiap elemen dengan nilai maksimum saat ini. Jika elemen yang diperiksa lebih besar, perbarui nilai maksimum.
- **Output:** Setelah iterasi selesai, nilai maksimum yang ditemukan dapat ditampilkan.

Contoh pseudocode untuk pencarian nilai maksimum:

```
go
max = a[0]
for i = 1 to n-1 do
    if a[i] > max then
        max = a[i]
end for
```

#### 3. Implementasi dalam Bahasa Go

Dalam bahasa Go, pencarian nilai ekstrim dapat dilakukan dengan mendefinisikan array dan menggunakan fungsi untuk menghitung nilai maksimum dan minimum. Berikut adalah contoh implementasi untuk mencari nilai maksimum dan minimum dalam array:

```
go
```

```
type arrInt [2023]int

func cariMax(arr arrInt, n int) int {
    max := arr[0]
    for i := 1; i < n; i++ {
        if arr[i] > max {
            max = arr[i]
        }
    }
    return max
}

func cariMin(arr arrInt, n int) int {
    min := arr[0]
    for i := 1; i < n; i++ {
        if arr[i] < min {
            min = arr[i]
        }
    }
    return min
}
```

## II. GUIDED

### 1. Soal Studi Case

Mencari Elemen Terkecil dalam Array

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
}

func identitas () {
    fmt.Println("=====")
    fmt.Println("Nama : Haposan Felix Marcel Siregar")
    fmt.Println("NIM : 2311102210")
    fmt.Println("=====")
}

// Fungsi main untuk menguji fungsi terkecil
func main() {
    identitas()
    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
}

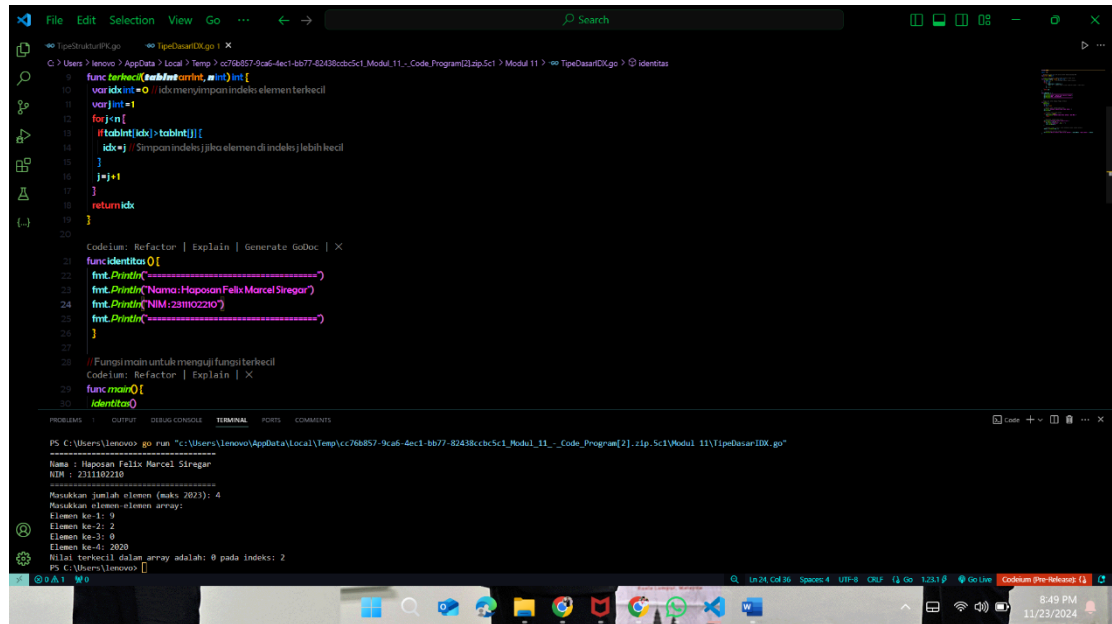
// Memasukkan elemen-elemen array
fmt.Println("Masukkan elemen-elemen array:")
for i := 0; i < n; i++ {
    fmt.Print("Elemen ke-", i+1, ": ")
    fmt.Scan(&tab[i])
}

// 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)
}

```

## Screenshot Output



The screenshot shows a Go IDE with the following code and output:

```
func terkecil(arr []int, n int) int {
    var idx int = 0 // idx menyimpan indeks elemen terkecil
    var j int = 1
    for j < n {
        if arr[idx] > arr[j] {
            idx = j // Simpan indeks jika elemen di indeks j lebih kecil
        }
        j++
    }
    return idx
}

func identitas() {
    fmt.Println("=====")
    fmt.Println("Nama: Haposan Felix Marcel Siregar")
    fmt.Println("NIM: 231102210")
    fmt.Println("=====")
}

// Fungsi main untuk menguji fungsi terkecil
func main() {
    identitas()
}
```

Terminal Output:

```
PS C:\Users\lenovo> go run "c:\Users\lenovo\AppData\Local\Temp\cc768857-9ca6-4ec1-bb77-82438c8c5c1_Modul_11_Code_Program[2].zip_Sc1Modul_11\TipeDasarIDXX.go"
Nama : Haposan Felix Marcel Siregar
NIM : 231102210
Masukkan jumlah elemen (maks. 2023): 4
Masukkan elemen elemen array:
Elemen ke-1: 9
Elemen ke-2: 2
Elemen ke-3: 0
Elemen ke-4: 2020
Nilai terkecil dalam array adalah: 0 pada indeks: 2
PS C:\Users\lenovo>
```

## Deskripsi Program

### Fungsi Terkecil

- Fungsi ini bertujuan untuk mencari indeks elemen terkecil dalam array. Fungsi ini menggunakan algoritma iteratif untuk membandingkan setiap elemen dalam array dan menyimpan indeks elemen terkecil.

### Fungsi Utama

- Input Jumlah Elemen: Program meminta pengguna untuk memasukkan jumlah elemen array yang akan diolah, dengan batas maksimum 2023 elemen.
- Validasi Input: Program memeriksa apakah jumlah elemen yang dimasukkan valid (antara 1 dan 2023). Jika tidak valid, program akan berhenti.
- Input Elemen Array: Pengguna diminta untuk memasukkan nilai-nilai elemen array satu per satu.
- Mencari Elemen Terkecil: Fungsi terkecil dipanggil untuk mencari indeks elemen terkecil dalam array.

## 2. Studi Case

Mencari IPK Tertinggi dalam Data Mahasiswa

### Source Code

```
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
}
```

```
func identitas () {
    fmt.Println("=====")
    fmt.Println("Nama : Haposan Felix Marcel Siregar")
    fmt.Println("NIM : 2311102210")
    fmt.Println("=====")
}
```

```
// Fungsi main untuk mengisi data mahasiswa dan mencari IPK tertinggi
```

```
func main() {
    identitas()
    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
}

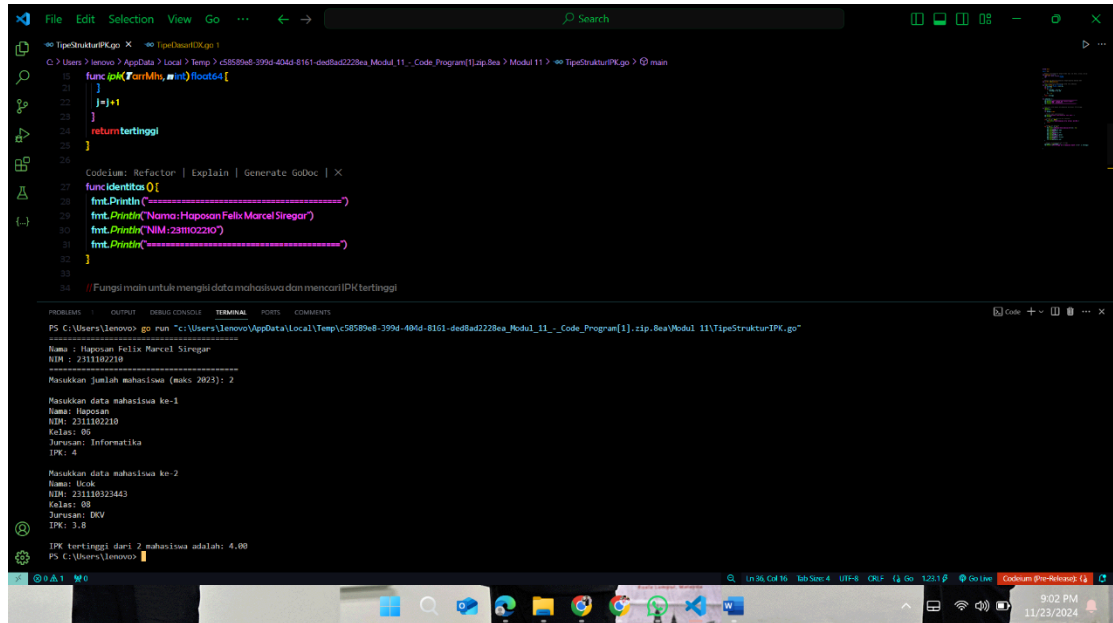
// Mengisi data mahasiswa
for i := 0; i < n; i++ {
    fmt.Printf("\nMasukkan data mahasiswa ke-%d\n", i+1)
    fmt.Print("Nama: ")
    fmt.Scan(&dataMhs[i].nama)
    fmt.Print("NIM: ")
    fmt.Scan(&dataMhs[i].nim)
    fmt.Print("Kelas: ")
    fmt.Scan(&dataMhs[i].kelas)
    fmt.Print("Jurusan: ")
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

```

Screenshot Program





```
File Edit Selection View Go ... Search
TipeStrukturIPK.go
10 func ipk(TarMhs, n int) float64 {
11     j := 0
12     j = j + 1
13 }
14 return tertinggi
15 }
16
17 Codeium: Refactor | Explain | Generate Guboc | X
18 func identitas 0 {
19     fmt.Println("=====")
20     fmt.Println("Nama: Hapusan Felix Marcel Siregar")
21     fmt.Println("NIM: 231102210")
22     fmt.Println("=====")
23 }
24
25 // Fungsi main untuk mengisi data mahasiswa dan mencari IPK tertinggi
26
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
PS C:\Users\lenovo> go run "C:\Users\lenovo\AppData\Local\Temp\c58589e8-399d-404d-8161-ded8d2228ea_Modul_11_-_Code_Program[1].zip.Bea\Modul_11\TipeStrukturIPK.go"
=====
Nama : Hapusan Felix Marcel Siregar
NIM : 231102210
=====
Masukkan jumlah mahasiswa (maks. 2023): 2
Masukkan data mahasiswa ke-1
Nama: Hapusan
NIM: 231102210
Kelas: 05
Jurusan: Informatika
IPK: 4
Masukkan data mahasiswa ke-2
Nama: Ulok
NIM: 23110323443
Kelas: 06
Jurusan: DKV
IPK: 3.8
IPK tertinggi dari 2 mahasiswa adalah: 4.00
PS C:\Users\lenovo>
```

## Deskripsi Program

- Input Jumlah Mahasiswa: Meminta pengguna untuk memasukkan jumlah mahasiswa yang akan diolah, dengan batas maksimum 2023.
- Validasi Input: Memeriksa apakah jumlah mahasiswa yang dimasukkan valid (antara 1 dan 2023). Jika tidak valid, program akan berhenti.
- Input Data Mahasiswa: Meminta pengguna untuk memasukkan data mahasiswa satu per satu (nama, NIM, kelas, jurusan, dan IPK).
- Mencari IPK Tertinggi: Memanggil fungsi ipk untuk mencari IPK tertinggi dalam array mahasiswa.

### III. UNGUIDED

#### 1. Soal Studi Case

Mendata berat anak kelinci, mencari berat terkecil dan terbesar

#### Source code

```
package main

import (
    "fmt"
)

const MAX int = 1000

type LarikFloat [MAX]float64

func isi_data(berat *LarikFloat, n *int) {
    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(n)
    fmt.Println("Masukkan berat anak kelinci:")
    for i := 0; i < *n; i++ {
        fmt.Printf("Berat anak kelinci ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }
}

func extrem(berat *LarikFloat, n int, min *float64, max *float64) {
    *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]
        }
    }
}
```

```

}

func identitas () {
    fmt.Println("=====")
    fmt.Println("Nama : Haposan Felix Marcel Siregar")
    fmt.Println("NIM : 231102210")
    fmt.Println("=====")
}

func main() {
    identitas()
    var berat LarikFloat
    var n int
    var min, max float64

    isi_data(&berat, &n)
    extrem(&berat, n, &min, &max)

    fmt.Printf("Berat terkecil: %.2f\n", min)
    fmt.Printf("Berat terbesar: %.2f\n", max)
}

```

## Screenshot Output

```

PS C:\Users\Lenovo> go run "C:\Users\Lenovo\AppData\Local\Temp\cc76b057-9ca6-4ec1-bb77-82438c9bc5c1_Modal_11_-_Code_Program[2].zip\Sc1\Modal_11\TipeDasarIDK.go"
=====
Nama : Haposan Felix Marcel Siregar
NIM : 231102210
=====
Masukkan jumlah anak kelinci: 4
Masukkan berat anak kelinci:
Berat anak kelinci ke-1: 2.3
Berat anak kelinci ke-2: 43
Berat anak kelinci ke-3: 5
Berat anak kelinci ke-4: 6
Berat terkecil: 2.30
Berat terbesar: 43.00
PS C:\Users\Lenovo>

```

### Deskripsi Program

Program ini mendata berat anak kelinci dan mencari berat terkecil serta terbesar. Pengguna diminta memasukkan jumlah anak kelinci dan berat masing-masing. Fungsi `extrem` digunakan untuk menentukan berat terkecil dan terbesar, kemudian hasilnya ditampilkan.

### 2. Soal Studi Case

Program ini mendata berat ikan dan mencari berat terkecil serta terbesar. Pengguna diminta memasukkan jumlah ikan dan berat masing-masing.

### Sourcecode

```
package main

import (
    "fmt"
)

const MAX int = 1000

// Fungsi untuk mengisi data berat ikan
func isi_data(berat *[MAX]float64, n *int) {
    fmt.Print("Masukkan jumlah ikan: ")
    fmt.Scan(n)
    fmt.Println("Masukkan berat ikan:")
    for i := 0; i < *n; i++ {
        fmt.Printf("Berat ikan ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }
}

// Fungsi untuk mencari berat terkecil dan terbesar
func extrem(berat *[MAX]float64, n int, min *float64, max *float64) {
    *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]
    }
}

}

func identitas () {
    fmt.Println("=====")
    fmt.Println("Nama : Haposan Felix Marcel Siregar")
    fmt.Println("NIM : 2311102210")
    fmt.Println("=====")
}

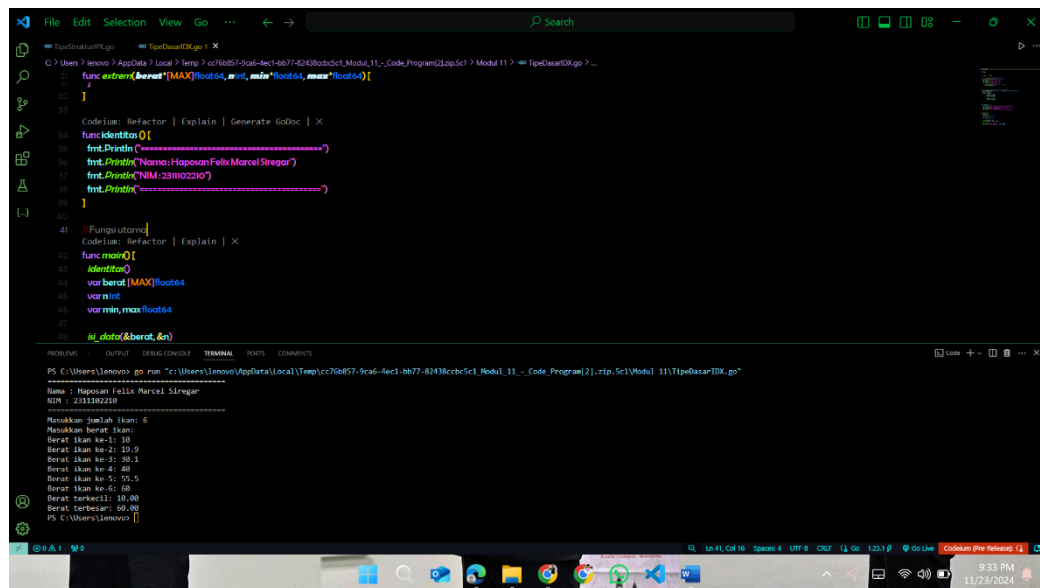
// Fungsi utama
func main() {
    identitas()
    var berat [MAX]float64
    var n int
    var min, max float64

    isi_data(&berat, &n)
    extrem(&berat, n, &min, &max)

    fmt.Printf("Berat terkecil: %.2f\n", min)
    fmt.Printf("Berat terbesar: %.2f\n", max)
}

```

## Screenshot Output



The screenshot shows a Go IDE with a dark theme. The editor displays a Go program with functions `extrem()`, `isi_data()`, and `main()`. The `main()` function calls `isi_data()` and `extrem()`. The terminal at the bottom shows the command `go run .` and the output of the program, which includes the name, NIM, and a list of fish weights.

```
1 func extrem(berat [MAX]float64, min, max float64) {
2     // ...
3 }
4
5 func isi_data() {
6     fmt.Println("=====")
7     fmt.Println("Nama: Haposen Felix Marcel Siregar")
8     fmt.Println("NIM: 231102210")
9     fmt.Println("=====")
10 }
11
12 // Fungsi utama
13 func main() {
14     isi_data()
15     var berat [MAX]float64
16     var min float64
17     var max float64
18     isi_data(&berat, &min, &max)
19 }
```

PS C:\Users\lenovo> go run .

=====

Nama : Haposen Felix Marcel Siregar

NIM : 231102210

=====

Masukkan jumlah ikan: 6

Masukkan berat ikan:

Berat ikan ke-1: 10

Berat ikan ke-2: 10.9

Berat ikan ke-3: 20.1

Berat ikan ke-4: 40

Berat ikan ke-5: 55.5

Berat ikan ke-6: 60

Berat terkecil: 10.00

Berat terbesar: 60.00

PS C:\Users\lenovo>

## Deskripsi Program

fungsi `isi_data()` untuk memasukkan jumlah dan berat ikan yang akan diproses, serta fungsi `extrem()` yang berfungsi untuk mencari nilai berat ikan terkecil dan terbesar dari data yang telah dimasukkan. Dalam program ini, saya menggunakan array untuk menyimpan data berat ikan dengan kapasitas maksimal 1000 data

### 3. Soal Studi Case

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.

## Sourcecode

```
package main

import (
    "fmt"
)

const MAX int = 100

type arrBalita [MAX]float64

// Fungsi untuk mengisi data berat balita
```

```

func isi_data(berat*arrBalita, n*int) {
    fmt.Print("Masukkan jumlah balita: ")
    fmt.Scan(n)
    fmt.Println("Masukkan berat balita (dalam kg):")
    for i := 0; i < n; i++ {
        fmt.Printf("Berat balita ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }
}

// Fungsi untuk mencari berat terkecil dan terbesar
func hitungMinMax(berat*arrBalita, n int, bMin*float64, bMax*float64) {
    *bMin = berat[0]
    *bMax = berat[0]
    for i := 1; i < n; i++ {
        if berat[i] < *bMin {
            *bMin = berat[i]
        }
        if berat[i] > *bMax {
            *bMax = berat[i]
        }
    }
}

// Fungsi untuk menghitung rata-rata berat balita
func rerata(berat*arrBalita, n int) float64 {
    var total float64
    for i := 0; i < n; i++ {
        total += berat[i]
    }
    return total / float64(n)
}

func identitas () {
    fmt.Println("=====")
    fmt.Println("Nama : Haposan Felix Marcel Siregar")
}

```

```

fmt.Println("NIM : 231102210")
fmt.Println("=====")
}

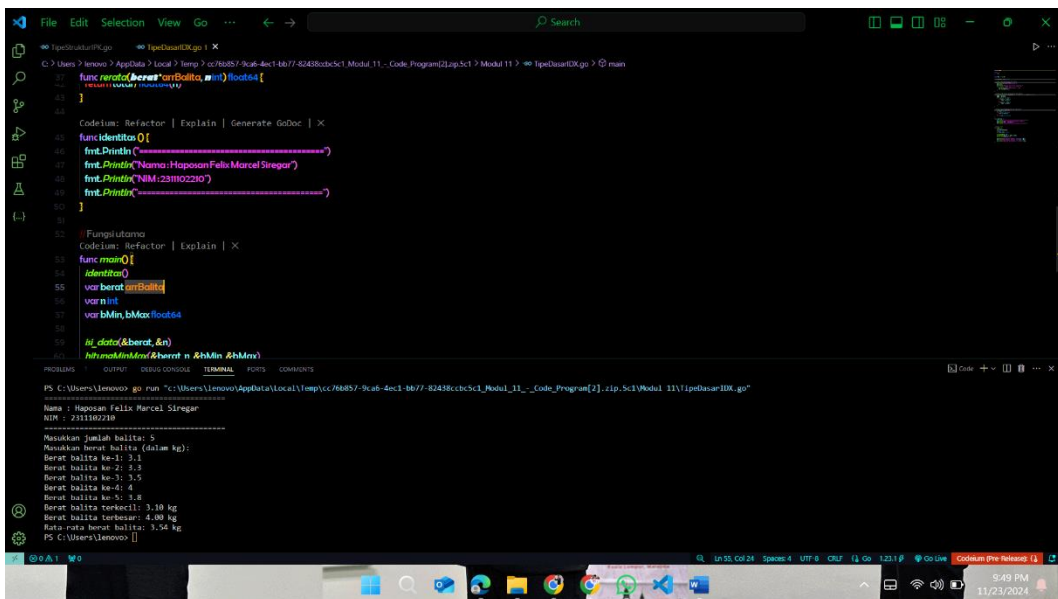
// Fungsi utama
func main() {
    identitas()
    var berat arrBalita
    var n int
    var bMin, bMax float64

    isi_data(&berat, &n)
    hitungMinMax(&berat, n, &bMin, &bMax)
    rata := rerata(&berat, n)

    fmt.Printf("Berat balita terkecil: %.2f kg\n", bMin)
    fmt.Printf("Berat balita terbesar: %.2f kg\n", bMax)
    fmt.Printf("Rata-rata berat balita: %.2f kg\n", rata)
}

```

## Screenshoot Program



```

C:\Users\lenovo> AppData\Local\Temp\cc7885f-9cab-4ec1-b677-82438c8c5c1_Modul_11\ipbdasar10X.go
func rerata(berat *arrBalita, n int) float64 {
    // ...
}

func identitas() {
    fmt.Println("=====")
    fmt.Println("Nama: Hapasan Felix Marcel Stregar")
    fmt.Println("NIM: 231102210")
    fmt.Println("=====")
}

// Fungsi utama
func main() {
    identitas()
    var berat arrBalita
    var n int
    var bMin, bMax float64

    isi_data(&berat, &n)
    hitungMinMax(&berat, n, &bMin, &bMax)
}

PS C:\Users\lenovo> go run ".\ipbdasar10X.go"
=====
Nama : Hapasan Felix Marcel Stregar
NIM : 231102210
=====
Masukkan jumlah balita: 5
Masukkan berat balita (dalam kg):
Berat balita ke-1: 3.1
Berat balita ke-2: 3.3
Berat balita ke-3: 3.5
Berat balita ke-4: 4
Berat balita ke-5: 3.8
Berat balita terkecil: 3.39 kg
Berat balita terbesar: 4.00 kg
Rata-rata berat balita: 3.54 kg
PS C:\Users\lenovo>

```



**Deskripsi Program**

fungsi `isi_data()` untuk input data berat balita, fungsi `hitungMinMax()` untuk mencari berat terendah dan tertinggi, serta fungsi `rerata()` untuk menghitung rata-rata berat dari seluruh balita yang didata. Output dari program akan menampilkan berat balita terkecil, terbesar, dan rata-rata dengan format dua angka desimal dalam satuan kilogram.