

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 11**  
**PENCARIAN NILAI EKSTREM PADA HIMPUNAN DATA**



**Disusun Oleh :**  
**Muhammad Djoko Susilo / 2311102212**

**Dosen Pengampu :**  
**Abednego Dwi Septiadi, S.Kom., M.Kom**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024**

## I. DASAR TEORI

### Pengertian Nilai Ekstrem

Nilai ekstrem adalah nilai yang menunjukkan elemen terbesar (maksimum) atau terkecil (minimum) dalam suatu kumpulan data. Nilai ini sering digunakan untuk menganalisis data karena memberikan informasi penting tentang batasan data. Dalam konteks array atau kumpulan elemen, nilai ekstrem mencakup elemen dengan nilai tertinggi (maksimum) dan nilai terendah (minimum). Contohnya, jika terdapat array berisi {5, 2, 9, 3}, nilai minimum adalah 2, sedangkan nilai maksimum adalah 9.

### Pencarian Nilai Ekstrem pada Array Bertipe Data Dasar

Pada array dengan tipe data dasar seperti int, float, atau char, pencarian nilai ekstrem dilakukan dengan iterasi sederhana. Prosesnya dimulai dengan menginisialisasi variabel untuk menyimpan nilai minimum dan maksimum dengan elemen pertama array. Kemudian, array diiterasi dari elemen kedua hingga akhir. Jika elemen array lebih kecil dari nilai minimum, nilai tersebut diperbarui menjadi nilai minimum baru. Sebaliknya, jika elemen array lebih besar dari nilai maksimum, nilai tersebut diperbarui menjadi nilai maksimum baru. Pendekatan ini efisien dengan kompleksitas waktu  $O(n)$ .

### Pencarian Nilai Ekstrem pada Array Bertipe Data Struktur

Pada array dengan tipe data struktur, elemen-elemen dalam array terdiri dari berbagai atribut (fields). Untuk mencari nilai ekstrem, perlu ditentukan atribut mana yang menjadi dasar perbandingan, misalnya berat badan, usia, atau skor. Prosesnya mirip dengan array tipe dasar, tetapi pembandingnya adalah atribut tertentu dari elemen struktur. Misalnya, dalam array balita dengan atribut berat, iterasi dilakukan dengan membandingkan nilai atribut berat di setiap elemen untuk menemukan nilai minimum dan maksimum.

## II. GUIDED

### 1. Soal Studi Case

Buatlah sebuah program untuk mencari nilai terkecil dalam sebuah array yang berisi elemen-elemen integer. Program harus meminta input jumlah elemen array (maksimum 2023 elemen), kemudian meminta input nilai untuk setiap elemen. Program kemudian menghitung dan menampilkan nilai terkecil serta indeks dari elemen terkecil tersebut dalam array.

**Sourcecode**

```

package main

import "fmt"

type arrInt [2023]int

func terkecil(tabInt arrInt, n int) int {
    var idx int = 0
    var j int = 1
    for j < n {
        if tabInt[idx] > tabInt[j] {
            idx = j
        }
        j = j + 1
    }
    return idx
}

func main() {
    var n int
    var tab arrInt

    fmt.Print("Masukkan jumlah elemen (maks 2023):")
    fmt.Scan(&n)

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

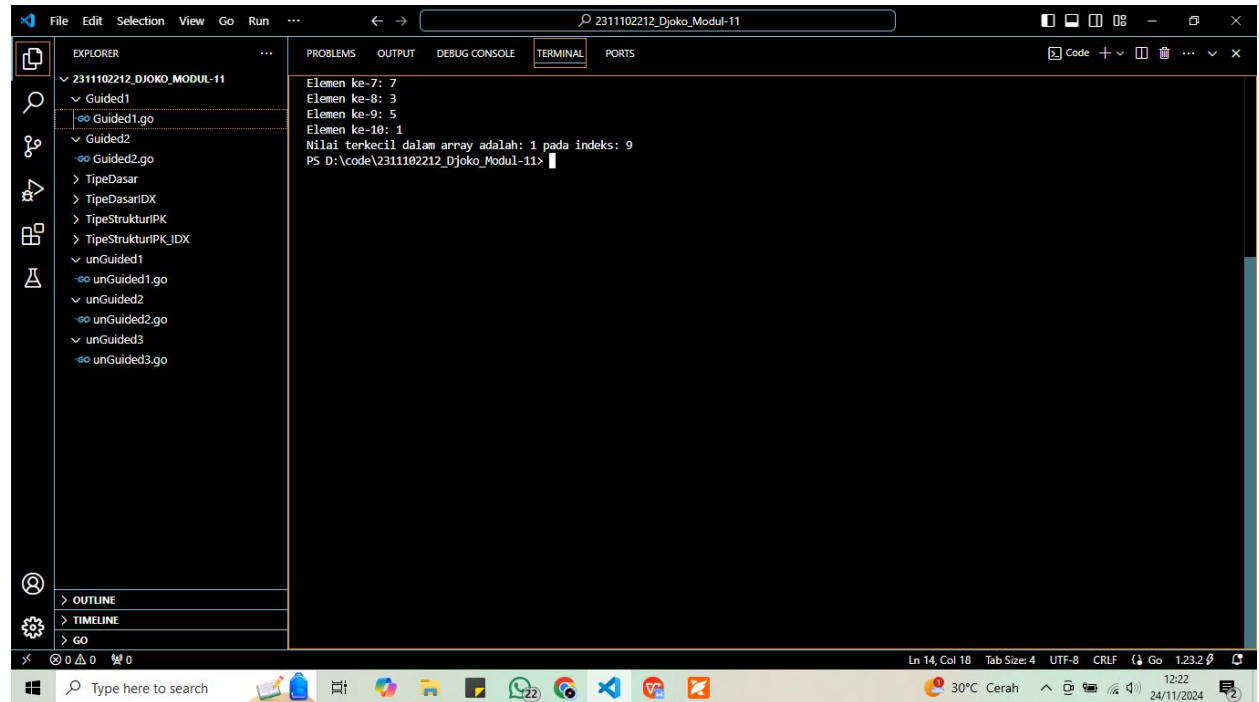
    fmt.Println("Masukkan elemen-elemen array:")
    for i := 0; i < n; i++ {
        fmt.Print("Elemen ke-", i+1, ": ")
        fmt.Scan(&tab[i])
    }

    idxMin := terkecil(tab, n)

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

```

## Screenshot Output

A screenshot of a Go IDE interface. The Explorer panel on the left shows a project named '2311102212\_DJOKO\_MODUL-11' with several files including 'Guided1.go', 'Guided2.go', 'unGuided1.go', 'unGuided2.go', and 'unGuided3.go'. The Terminal panel on the right displays the output of a program execution. The output text is: 'Elemen ke-7: 7', 'Elemen ke-8: 3', 'Elemen ke-9: 5', 'Elemen ke-10: 1', 'Nilai terkecil dalam array adalah: 1 pada indeks: 9', and 'PS D:\code\2311102212\_Djoko\_Modul-11>'. The status bar at the bottom indicates 'Ln 14, Col 18', 'Tab Size: 4', 'UTF-8', 'CRLF', and 'Go 1.23.2'.

```
Elemen ke-7: 7
Elemen ke-8: 3
Elemen ke-9: 5
Elemen ke-10: 1
Nilai terkecil dalam array adalah: 1 pada indeks: 9
PS D:\code\2311102212_Djoko_Modul-11>
```

## Deskripsi Program

Program ini adalah program yang dirancang untuk mencari nilai terkecil dalam sebuah array dan menampilkan indeks serta nilai terkecil tersebut. Cara kerja program ini adalah pertama-tama program akan meminta pengguna untuk memasukkan jumlah elemen array yang ingin diproses, dengan batas maksimal 2023 elemen. Setelah itu, program akan meminta pengguna untuk memasukkan nilai elemen-elemen array satu per satu. Program kemudian mencari nilai terkecil dalam array dengan membandingkan setiap elemen. Di akhir, program akan menampilkan nilai terkecil beserta indeks elemen yang memiliki nilai tersebut.

## 2. Soal Studi Case

Buatlah program untuk mendata mahasiswa dengan atribut nama, NIM, kelas, jurusan, dan IPK. Program harus dapat menginput data mahasiswa (maksimal 2023 mahasiswa) dan menghitung IPK tertinggi dari data yang dimasukkan. Setelah selesai, program menampilkan IPK tertinggi yang ada.

## 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
}

type arrMhs [2023]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 main() {
    var n int
    var dataMhs arrMhs

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

    if n < 1 || n > 2023 {
        fmt.Println("Jumlah mahasiswa harus antara 1
dan 2023.")
        return
    }

    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)
    }
}

```

```

    }

    tertinggi := ipk(dataMhs, n)
    fmt.Printf("¥nIPK tertinggi dari %d mahasiswa
    adalah: %.2f¥n", n, tertinggi)

}

```

## Screenshot Output

```

PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\Guided2\Guided2.go"
Masukkan jumlah mahasiswa (maks 2023): 3

Masukkan data mahasiswa ke-1
Nama: Amat
NIM: 2311101
Kelas: 01
Jurusan: Informatika
IPK: 3.4

Masukkan data mahasiswa ke-2
Nama: Ikan
NIM: 2311102
Kelas: 01
Jurusan: Informatika
IPK: 3.7

Masukkan data mahasiswa ke-3
Nama: Joko
NIM: 2311103
Kelas: 01
Jurusan: Informatika
IPK: 3.8

IPK tertinggi dari 3 mahasiswa adalah: 3.80
PS D:\code\2311102212_Djoko_Modul-11>

```

## Deskripsi Program

Program ini adalah program yang dirancang untuk mendata informasi mahasiswa dan menghitung IPK tertinggi di antara mahasiswa yang ada. Cara kerja program ini adalah pertama-tama program akan meminta pengguna untuk memasukkan jumlah mahasiswa yang akan didata (dengan batas maksimal 2023 mahasiswa). Setelah itu, program akan meminta pengguna untuk memasukkan data setiap mahasiswa, yang meliputi nama, NIM, kelas, jurusan, dan IPK. Semua data mahasiswa tersebut akan disimpan dalam sebuah array of struct.

Setelah semua data mahasiswa dimasukkan, program akan memproses data tersebut untuk mencari IPK tertinggi menggunakan fungsi `ipk`. Fungsi ini akan membandingkan setiap IPK yang ada dalam array dan mencari nilai tertinggi. Nilai tertinggi tersebut kemudian akan ditampilkan kepada pengguna sebagai hasil akhir.

Di akhir program, program akan menampilkan IPK tertinggi dari sekian banyak mahasiswa yang telah didata. Program ini sangat berguna untuk mengetahui IPK terbaik di suatu kelompok mahasiswa dalam suatu jurusan atau kelas.

### III. UNGUIDED

#### 1. Soal Studi Case

Buatlah sebuah program yang bisa untuk mendata berat dari anak kelinci yang akan dijual ke pasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat anak kelinci yang akan dijual.

##### Sourcecode

```
package main

import (
    "fmt"
)

type Rabbit struct {
    weight int
}

type arrRabbit [1000]Rabbit

func main() {
    var n2311102212 int
    var rabbits arrRabbit

    fmt.Print("Banyaknya anak kelinci: ")
    fmt.Scan(&n2311102212)

    for i := 0; i < n2311102212; i++ {
        i+1)    fmt.Printf("Anak kelinci ke-%d, berat: ",
                fmt.Scan(&rabbits[i].weight)
    }

    maks := beratTerbesar(rabbits, n2311102212)
    min := beratTerkecil(rabbits, n2311102212)

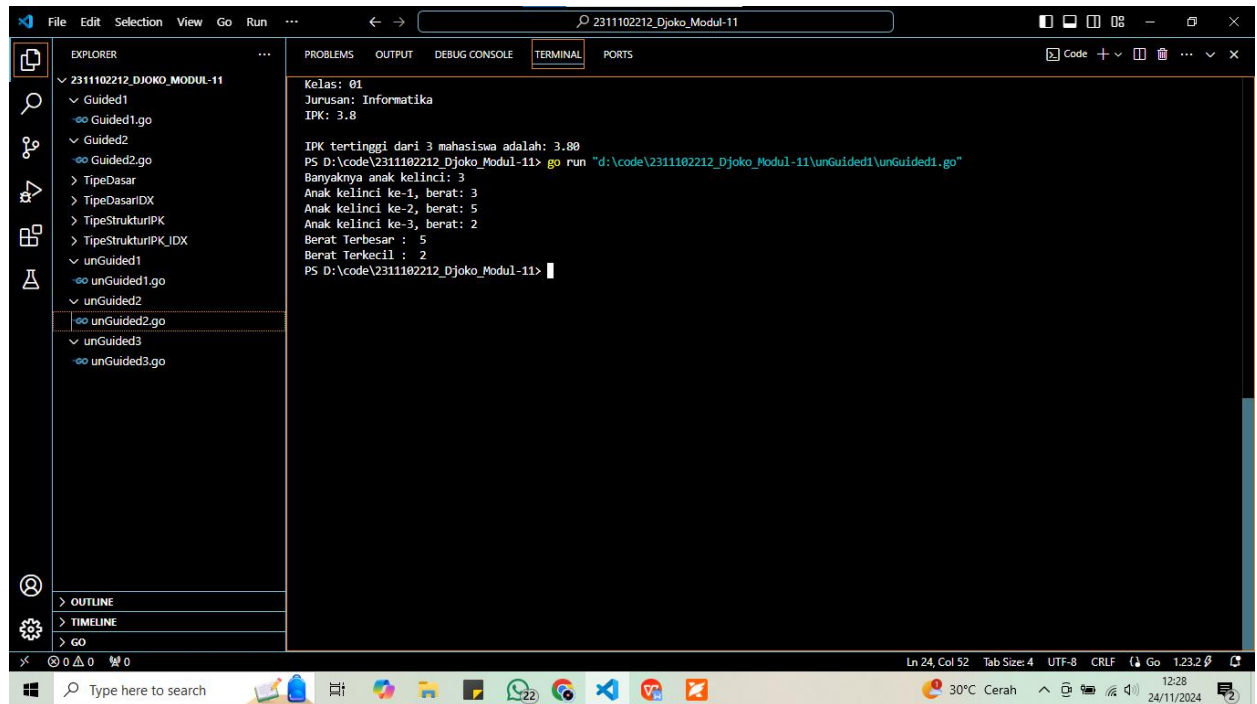
    fmt.Println("Berat Terbesar : ", maks)
    fmt.Println("Berat Terkecil : ", min)
}
```

```
func beratTerbesar(T arrRabbit, n int) int {
    maks := T[0].weight
    for i := 1; i < n; i++ {
        if T[i].weight > maks {
            maks = T[i].weight
        }
    }
    return maks
}

func beratTerkecil(T arrRabbit, n int) int {
    min := T[0].weight
    for i := 1; i < n; i++ {
        if T[i].weight < min {
            min = T[i].weight
        }
    }
    return min
}
```

### Screenshot Output





```
File Edit Selection View Go Run ...
2311102212_Djoko_Modul-11
EXPLORER
  2311102212_DJOKO_MODUL-11
    Guided1
    Guided1.go
    Guided2
    Guided2.go
    TipeDasar
    TipeDasariDX
    TipeStrukturIPK
    TipeStrukturIPK_JDX
    unGuided1
    unGuided1.go
    unGuided2
    unGuided2.go
    unGuided3
    unGuided3.go
  OUTLINE
  TIMELINE
  GO
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Kelas: 01
Jurusan: Informatika
IPK: 3.8
IPK tertinggi dari 3 mahasiswa adalah: 3.80
PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\unGuided1\unGuided1.go"
Banyaknya anak kelinci: 3
Anak kelinci ke-1, berat: 3
Anak kelinci ke-2, berat: 5
Anak kelinci ke-3, berat: 2
Berat Terbesar : 5
Berat Terkecil : 2
PS D:\code\2311102212_Djoko_Modul-11>
Ln 24, Col 52 Tab Size: 4 UTF-8 CRLF 12:28 24/11/2024
```

## Deskripsi Program

Program ini program yang dirancang untuk mendata berat anak kelinci yang menghasilkan nilai terbesar dan terkecil dari semua kelinci yang ada, Cara kerja dari program ini adalah pertama tama program akan meminta pengguna untuk memasukan jumlah dari anak kelinci yang ada, setelah itu program akan melakukan pengulangan untuk mendata satu persatu berat dari anak kelinci. Diakhir terdapat sebuah data yang menjabarkan berat kelinci terbesar dan terkecil.

## 2. Soal Studi Case

Buatlah program untuk menghitung total berat ikan di setiap wadah dan rata-rata berat ikan per wadah. Program ini bertujuan untuk membantu menghitung distribusi berat ikan berdasarkan jumlah ikan yang dimasukkan dalam beberapa wadah.

### Sourcecode

```
package main

import (
    "fmt"
)

func hitungTotalBerat(weights []float64, x, y int)
[]float64 {
    totalWeights := make([]float64, (x+y-1)/y)
    var idx int
    for i := 0; i < len(totalWeights); i++ {
```

```

        for j := 0; j < y && idx < x; j++ {
            totalWeights[i] += weights[idx]
            idx++
        }
    }
    return totalWeights
}

func hitungRataRata(totalWeights []float64) float64 {
    var totalWeight float64
    for _, w := range totalWeights {
        totalWeight += w
    }
    return totalWeight / float64(len(totalWeights))
}

func main() {
    var x2311102212, y2311102212 int
    fmt.Print("Masukkan jumlah ikan (x) dan jumlah
ikan per wadah (y): ")
    fmt.Scan(&x2311102212, &y2311102212)

    weights := make([]float64, x2311102212)
    fmt.Println("Masukkan berat ikan (pisahkan
dengan spasi):")
    for i := 0; i < x2311102212; i++ {
        fmt.Scan(&weights[i])
    }

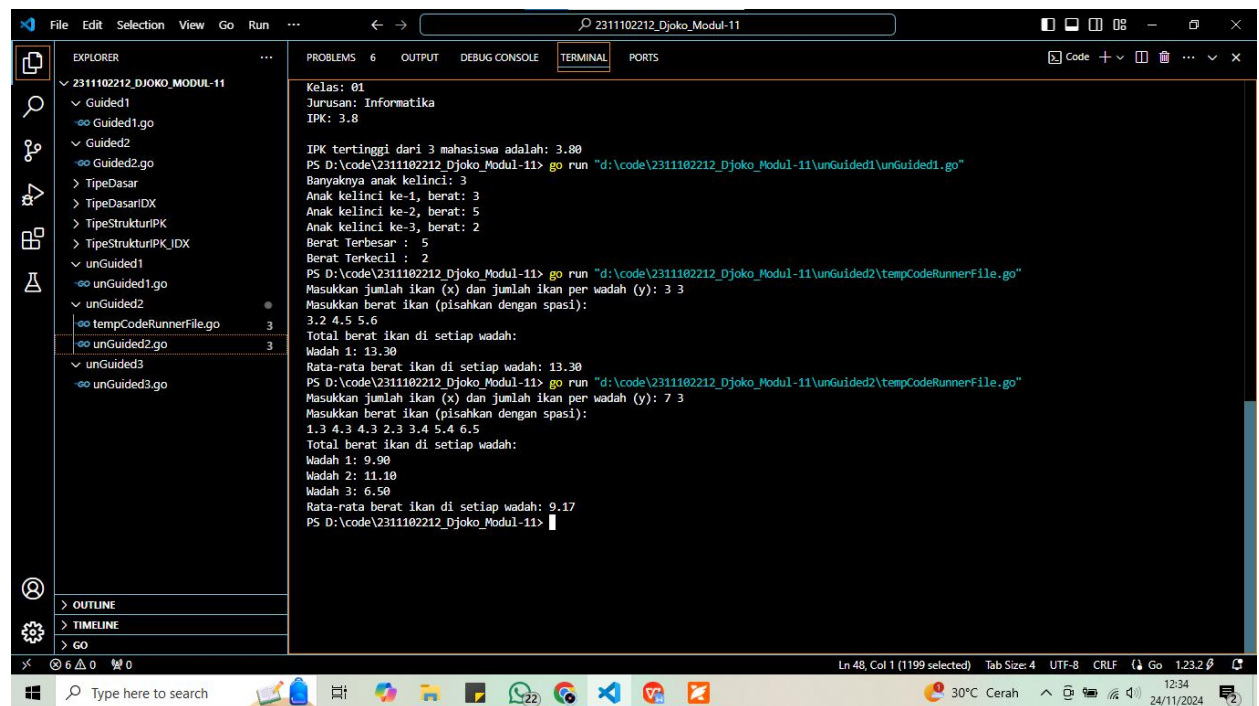
    totalWeights := hitungTotalBerat(weights,
x2311102212, y2311102212)

    fmt.Println("Total berat ikan di setiap wadah:")
    for i, weight := range totalWeights {
        fmt.Printf("Wadah %d: %.2f¥n", i+1, weight)
    }

    averageWeight := hitungRataRata(totalWeights)
    fmt.Printf("Rata-rata berat ikan di setiap
wadah: %.2f¥n", averageWeight)
}

```

## Screenshot Output



The screenshot shows a Go IDE with the following content in the terminal:

```
Kelas: 01
Jurusan: Informatika
IPK: 3.8

IPK tertinggi dari 3 mahasiswa adalah: 3.80
PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\unGuided1\unGuided1.go"
Banyaknya anak kelinci: 3
Anak kelinci ke-1, berat: 3
Anak kelinci ke-2, berat: 5
Anak kelinci ke-3, berat: 2
Berat Terbesar : 5
Berat Terkecil : 2
PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\unGuided2\tempCodeRunnerFile.go"
Masukkan jumlah ikan (x) dan jumlah ikan per wadah (y): 3 3
Masukkan berat ikan (pisahkan dengan spasi):
3 2 4 5 5 6
Total berat ikan di setiap wadah:
Wadah 1: 13.30
Rata-rata berat ikan di setiap wadah: 13.30
PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\unGuided2\tempCodeRunnerFile.go"
Masukkan jumlah ikan (x) dan jumlah ikan per wadah (y): 7 3
Masukkan berat ikan (pisahkan dengan spasi):
1 3 4 3 4 3 2 3 3 4 5 4 6 5
Total berat ikan di setiap wadah:
Wadah 1: 9.90
Wadah 2: 11.10
Wadah 3: 6.50
Rata-rata berat ikan di setiap wadah: 9.17
PS D:\code\2311102212_Djoko_Modul-11>
```

## Deskripsi Program

Program ini adalah program yang dirancang untuk mendata berat ikan yang akan dikelompokkan dalam wadah dengan kapasitas tertentu dan menghitung total serta rata-rata berat ikan di setiap wadah. Cara kerja program ini adalah pertama-tama program akan meminta pengguna untuk memasukkan jumlah ikan yang ada dan jumlah ikan per wadah. Setelah itu, program akan meminta pengguna untuk memasukkan berat ikan satu per satu. Berat ikan tersebut kemudian dikelompokkan ke dalam wadah sesuai dengan kapasitas yang ditentukan. Setelah semua data dimasukkan, program menghitung total berat ikan di setiap wadah dan rata-rata berat ikan per wadah. Di akhir, program menampilkan total berat ikan di setiap wadah dan rata-rata berat ikan per wadah.

### 3. Soal Studi Case

Buatlah program untuk sebuah Posyandu yang dapat mencatat data berat badan balita dalam kilogram (kg). Data ini akan dianalisis untuk mendapatkan informasi mengenai berat minimum, berat maksimum, dan rata-rata berat balita.

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

func main() {
    var n2311102212 int
    var arr arrBalita
    var bMin, bMax float64

    fmt.Print("Masukkan banyak data berat balita: ")
    fmt.Scan(&n2311102212)

    for i := 0; i < n2311102212; i++ {
        fmt.Printf("Masukkan berat balita ke-%d: ",
i+1)
        fmt.Scan(&arr[i])
    }
}
```

```

        hitungMinMax(arr, n2311102212, &bMin, &bMax)
        avg := rerata(arr, n2311102212)

        fmt.Printf("¥nBerat balita minimum: %.2f kg¥n",
bMin)
        fmt.Printf("Berat balita maksimum: %.2f kg¥n",
bMax)
        fmt.Printf("Rerata berat balita: %.2f kg¥n", avg)
    }

```

## Screenshot Program

```

PS D:\code\2311102212_Djoko_Modul-11> go run "d:\code\2311102212_Djoko_Modul-11\unGuided3\unGuided3.go"
Masukkan banyak data berat balita: 3
Masukkan berat balita ke-1: 5.4
Masukkan berat balita ke-2: 4.7
Masukkan berat balita ke-3: 5.4

Berat balita minimum: 4.70 kg
Berat balita maksimum: 5.40 kg
Rerata berat balita: 5.17 kg
PS D:\code\2311102212_Djoko_Modul-11>

```

## Deskripsi Program

Program ini adalah program yang dirancang untuk mendata berat balita yang akan dihitung nilai ekstrem (terkecil dan terbesar) serta rerata beratnya. Cara kerja program ini adalah pertama-tama program akan meminta pengguna untuk memasukkan jumlah data berat balita yang ada. Setelah itu, program akan meminta pengguna untuk memasukkan berat balita satu per satu. Berat balita tersebut kemudian disimpan dalam array untuk diproses lebih lanjut.

Setelah semua data berat balita dimasukkan, program akan menghitung nilai ekstrem (berat balita terkecil dan terbesar) menggunakan fungsi `hitungMinMax`. Fungsi ini memeriksa setiap elemen dalam array untuk menemukan nilai minimum dan maksimum. Fungsi ini akan mengembalikan nilai-nilai ekstrem tersebut yang kemudian ditampilkan ke pengguna.

Selain menghitung nilai ekstrem, program juga menghitung rata-rata berat balita dengan menggunakan fungsi `rerata`. Fungsi ini akan menjumlahkan semua berat balita dalam array dan membaginya dengan jumlah balita untuk mendapatkan rata-rata berat.

Di akhir, program akan menampilkan hasil berupa berat balita terkecil, berat balita terbesar, dan rata-rata berat balita yang telah dihitung. Program ini sangat berguna untuk analisis data berat balita dalam suatu kelompok.