

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 11
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA



Disusun Oleh:

Muhammad Djulianoor / 2311102253

Kelas

IF-11-06

Dosen Pengampu:

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dasar Teori

Searching

Searching merupakan metode untuk mencari suatu data. Pada pemrograman, *searching* dapat dilakukan untuk mencari data pada *array* tertentu. *Searching* dapat mencari nilai terbesar, nilai terkecil, data tertentu, dan posisi data tersebut disimpan.

Sequential Search

Sequential search ialah metode *searching* yang membandingkan sekumpulan elemen data yang ada dengan memeriksa kumpulan data secara satu-persatu dari awal hingga akhir.

Binary Search

Binary Search merupakan metode *search* yang melakukan pembagian ruang pencarian secara berulang-ulang sampai data ditemukan atau sampai ruang pencarian tidak dapat dibagi lagi. Metode ini memperkecil jumlah operasi perbandingan yang harus dilakukan antara data yang dicari dengan data yang ada di dalam tabel, khususnya untuk jumlah data yang sangat besar ukurannya. Syarat dalam metode ini adalah data sudah dalam keadaan terurut.

II. GUIDED

1. Soal Studi Case

Source code

```
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
```

```

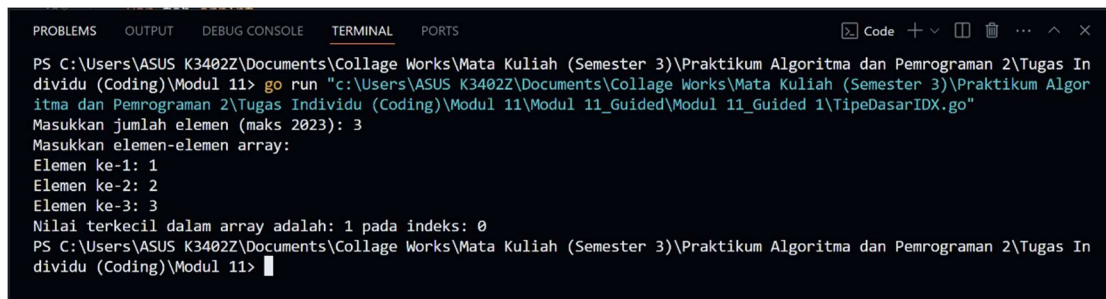
fmt.Println("Masukkan elemen-elemen array:")
for i := 0; i < n; i++ {
    fmt.Print("Elemen ke-", i+1, ": ")
    fmt.Scan(&tab[i])
}

// 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



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11> go run "c:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11\Modul 11_Guided\Modul 11_Guided 1\TipeDasarIDX.go"
Masukkan jumlah elemen (maks 2023): 3
Masukkan elemen-elemen array:
Elemen ke-1: 1
Elemen ke-2: 2
Elemen ke-3: 3
Nilai terkecil dalam array adalah: 1 pada indeks: 0
PS C:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11>

```

Deskripsi Program

Program di atas adalah contoh implementasi dari program yang menggunakan algoritma *searching*. Program tersebut menggunakan *array* untuk menyimpan data yang akan dicari. Data yang akan dicari pada program ini adalah data yang nilainya terkecil. Fungsi dengan nama terkecil bertujuan untuk membuat algoritma *searching* untuk menentukan nilai manakah yang terkecil pada *array*.

Di dalam fungsi terkecil terdapat variabel *idx* yang bertujuan untuk menyimpan indeks elemen terkecil. Lalu, terdapat variabel *j* yang diinisialisasi dengan nilai 1 yang kemudian dibandingkan pada perulangan dengan variabel *n* ($j < n$).

2. Soal Studi Case

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 indeks IPK tertinggi dalam array mahasiswa
func indeksIPKTertinggi(T arrMhs, n int) int {
    var idx int = 0 // Inisialisasi indeks IPK tertinggi pada indeks
    pertama
    for j := 1; j < n; j++ {
        if T[idx].ipk < T[j].ipk {
            idx = j // Update indeks jika ditemukan IPK yang lebih
            tinggi
        }
    }
    return idx
}

// Fungsi main untuk mengisi data mahasiswa dan mencari IPK tertinggi
// beserta indeksnya
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
}

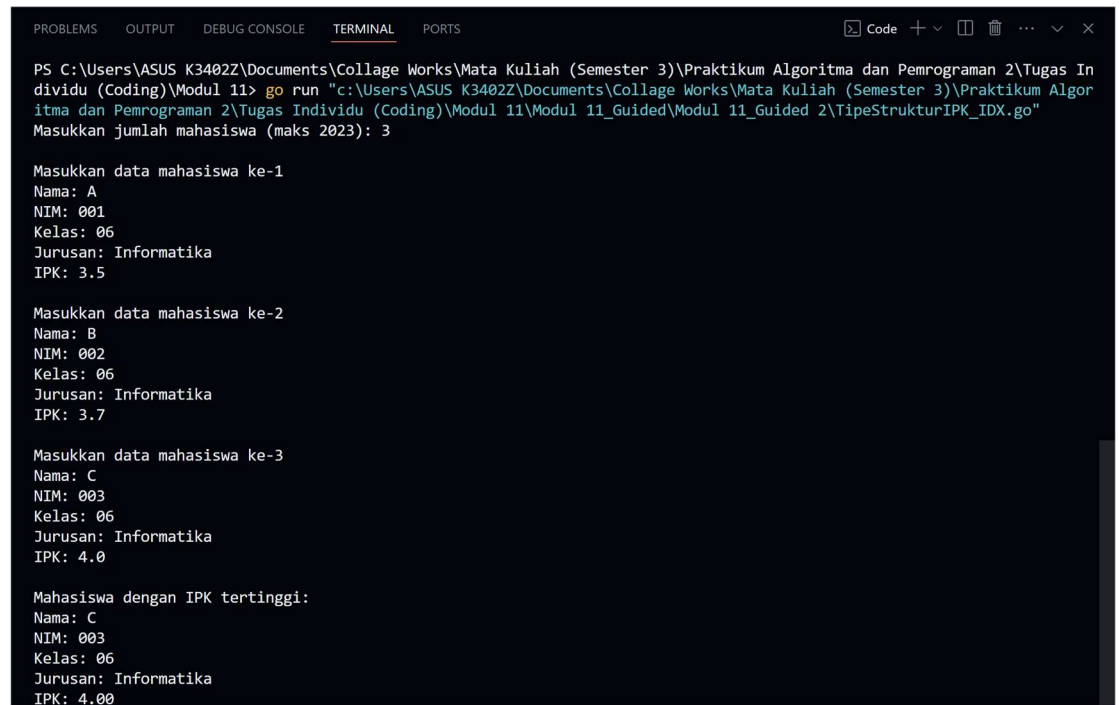
// Mengisi data mahasiswa
for i := 0; i < n; i++ {
    fmt.Printf("\nMasukkan data mahasiswa ke-%d\n", i+1)
    fmt.Print("Nama: ")
    fmt.Scan(&dataMhs[i].nama)
    fmt.Print("NIM: ")
    fmt.Scan(&dataMhs[i].nim)
    fmt.Print("Kelas: ")
    fmt.Scan(&dataMhs[i].kelas)
    fmt.Print("Jurusan: ")
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

// Mendapatkan indeks IPK tertinggi
idxTertinggi := indeksIPKTertinggi(dataMhs, n)

// Menampilkan data mahasiswa dengan IPK tertinggi
fmt.Printf("\nMahasiswa dengan IPK tertinggi:\n")
fmt.Printf("Nama: %s\n", dataMhs[idxTertinggi].nama)
fmt.Printf("NIM: %s\n", dataMhs[idxTertinggi].nim)
fmt.Printf("Kelas: %s\n", dataMhs[idxTertinggi].kelas)
fmt.Printf("Jurusan: %s\n", dataMhs[idxTertinggi].jurusan)
fmt.Printf("IPK: %.2f\n", dataMhs[idxTertinggi].ipk)
}

```

Screenshoot Output



```
PS C:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11> go run "c:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11\Modul 11_Guided\Modul 11_Guided 2\TipeStrukturIPK_IDX.go"
Masukkan jumlah mahasiswa (maks 2023): 3

Masukkan data mahasiswa ke-1
Nama: A
NIM: 001
Kelas: 06
Jurusan: Informatika
IPK: 3.5

Masukkan data mahasiswa ke-2
Nama: B
NIM: 002
Kelas: 06
Jurusan: Informatika
IPK: 3.7

Masukkan data mahasiswa ke-3
Nama: C
NIM: 003
Kelas: 06
Jurusan: Informatika
IPK: 4.0

Mahasiswa dengan IPK tertinggi:
Nama: C
NIM: 003
Kelas: 06
Jurusan: Informatika
IPK: 4.00
```

Deskripsi Program

Program di atas adalah program untuk mencari mahasiswa mana yang memiliki IPK tertinggi dibandingkan mahasiswa lainnya. Program ini mengimplementasikan algoritma *searching* dan juga *struct*. Ketika program dijalankan, program meminta masukan dari pengguna, berapa banyak mahasiswa yang akan dimasukkan ke dalam program sebelum program membandingkan IPK mahasiswa. Ketika sudah dimasukkan, program akan mencari IPK mana yang tertinggi dibandingkan mahasiswa lainnya.

Untuk mencari tahu IPK mana yang lebih tinggi, pada kode program digunakan *syntax*

```
for j := 1; j < n; j++ {
    if T[idx].ipk < T[j].ipk {
        idx = j
    }
}
```

Perulangan tersebut merupakan algoritma *searching, statement* pada percabangan *if* bertujuan untuk memperbarui jika ada IPK yang lebih tinggi dibandingkan mahasiswa lainnya.

III. UNGUIDED

1. Soal Studi Case

Sebuah program digunakan untuk mendata berat anak kelinci yang akan dijual ke pasar. Program ini menggunakan *array* dengan kapasitas 1000 untuk menampung data berat anak kelinci yang akan dijual.

Masukan terdiri dari sekumpulan bilangan, yang mana bilangan pertama adalah bilangan bulat N yang menyatakan banyaknya anak kelinci yang akan ditimbang beratnya. Selanjutnya N bilangan rill berikutnya adalah berat dari anak kelinci yang akan dijual.

Keluaran terdiri dari dua buah bilangan rill yang menyatakan berat kelinci terkecil dan terbesar.

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var n int
    fmt.Print("Masukkan jumlah anak kelinci (N): ")
    fmt.Scan(&n)

    if n <= 0 || n > 1000 {
        fmt.Println("Jumlah anak kelinci harus di antara 1 hingga 1000.")
        return
    }

    beratKelinci := make([]float64, n)
    fmt.Println("Masukkan berat masing-masing anak kelinci:")

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

```

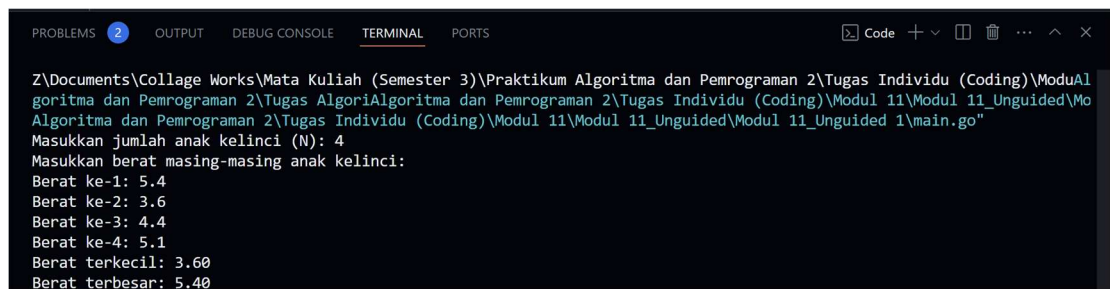
beratTerkecil := math.MaxFloat64
BeratMaksimal := -math.MaxFloat64

for _, weight := range beratKelinci {
    if weight < beratTerkecil {
        beratTerkecil = weight
    }
    if weight > BeratMaksimal {
        BeratMaksimal = weight
    }
}

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

```

Screenshoot Output



```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Z:\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11\Modul 11_Unguided\Modul 11_Unguided 1\main.go
Masukkan jumlah anak kelinci (N): 4
Masukkan berat masing-masing anak kelinci:
Berat ke-1: 5.4
Berat ke-2: 3.6
Berat ke-3: 4.4
Berat ke-4: 5.1
Berat terkecil: 3.60
Berat terbesar: 5.40

```

Deskripsi Program

Program di atas adalah mencari berat badan kelinci yang terkecil dan juga terbesar. Program akan meminta pengguna untuk memasukkan jumlah anak kelinci terlebih dahulu. Lalu, pengguna diminta memasukkan berat badan kelinci. Setelahnya, program akan mengetahui berat badan terkecil dan juga berat badan terbesar dari kelinci.

2. Soal Studi Case

Sebuah program digunakan untuk menentukan tarif ikan yang akan dijual ke pasar. Program ini menggunakan array dengan kapasitas 1000 untuk menampung data berat ikan yang akan dijual.

Masukan terdiri dari dua baris, yang mana baris pertama terdiri dari dua bilangan bulat x dan y. Bilangan x menyatakan banyaknya ikan yang akan dijual, sedangkan y adalah banyaknya ikan yang akan dimasukkan ke dalam wadah. Baris kedua terdiri dari sejumlah x bilangan rill yang menyatakan banyaknya ikan yang akan dijual.

Keluaran terdiri dari dua baris. Baris pertama adalah kumpulan bilangan rill yang menyatakan total berat ikan di setiap wadah (jumlah wadah tergantung pada nilai x dan y, urutan ikan yang dimasukkan ke dalam wadah sesuai urutan pada masukan baris ke-2). Baris kedua adalah sebuah bilangan rill yang menyatakan berat rata-rata ikan di setiap wadah.

Source Code

```
package main

import (
    "fmt"
)

func bacaBeratIkan(x int) []float64 {
    berat := make([]float64, x)
    fmt.Println("Masukkan berat ikan (dipisahkan dengan spasi):")
    for i := 0; i < x; i++ {
        fmt.Scan(&berat[i])
    }
    return berat
}

func kelompokkanIkan(berat []float64, x, y int) ([][]float64, []float64) {
    var containers [][]float64
    var totalBerat []float64

    for i := 0; i < x; i++ {
        if i%y == 0 {
            containers = append(containers, []float64{})
        }
    }
}
```

```

    }
    lastIndex := len(containers) - 1
    containers[lastIndex] = append(containers[lastIndex], berat[i])
}

var sum_berat float64
for _, container := range containers {
    var BeratTotal float64
    for _, weight := range container {
        BeratTotal += weight
    }
    totalBerat = append(totalBerat, BeratTotal)
    sum_berat += BeratTotal
}

return containers, totalBerat
}

func hitungRataRata(totalBerat []float64) float64 {
    var sum float64
    for _, weight := range totalBerat {
        sum += weight
    }
    return sum / float64(len(totalBerat))
}

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

    if x <= 0 || y <= 0 || x > 1000 {
        fmt.Println("Nilai x harus antara 1 hingga 1000, dan y harus lebih dari 0.")
        return
    }

    berat := bacaBeratIkan(x)

    _, totalBerat := kelompokkanIkan(berat, x, y)

```

```

    fmt.Println("Total berat ikan di setiap wadah:")
    for _, BeratTotal := range totalBerat {
        fmt.Printf("%.2f ", BeratTotal)
    }
    fmt.Println()

    averageWeight := hitungRataRata(totalBerat)
    fmt.Printf("Berat rata-rata ikan di setiap wadah: %.2f\n",
averageWeight)
}

```

Screenshoot Output



```

PS C:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11> go run "c:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11\Modul 11_Unguided\Modul 11_Unguided 2\main2.go"
Masukkan jumlah ikan (x) dan jumlah ikan per wadah (y): 5 2
Masukkan berat ikan (dipisahkan dengan spasi):
2.3 3.1 1.2 2.3 4.1
Total berat ikan di setiap wadah:
5.40 3.50 4.10
Berat rata-rata ikan di setiap wadah: 4.33

```

Deskripsi Program

Program di atas merupakan program untuk mencari berat rata-rata dari ikan yang berada di suatu wadah. Program akan meminta pengguna untuk memasukkan berat ikan, kemudian program akan mencari rata-ratanya.

```

func hitungRataRata(totalBerat []float64) float64 {

    var sum float64

    for _, weight := range totalBerat {

        sum += weight

    }

    return sum / float64(len(totalBerat))

}

```

Syntax tersebut merupakan bagaimana program bisa mencari rata-rata berat ikan.

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 rata-ratanya.

Buatlah program dengan spesifikasi subprogram sebagai berikut:

```
type arrBalita [100]float64
```

```
func hitungMinMax(arrBerat arrBalita; bMin, bMax *float64) {  
    /* I.S. Terdefinisi array dinamis arrBerat
```

```
    Proses: Menghitung berat minimum dan maksimum dalam array  
    F.S. Menampilkan berat minimum dan maksimum balita */
```

```
    ...  
}
```

```
function rerata (arrBerat arrBalita) real {  
    /* menghitung dan mengembalikan rerata berat balita dalam array */  
    ...  
}
```

Source Code

```
package main  
  
import (  
    "fmt"  
    "math"  
)  
  
type arrBalita [100]float64  
  
func hitungMinMax(arrBerat arrBalita, n int, bMin, bMax *float64) {  
    *bMin = math.MaxFloat64  
    *bMax = -math.MaxFloat64  
  
    for i := 0; i < n; i++ {  
        if arrBerat[i] < *bMin {  
            *bMin = arrBerat[i]
```

```

    }
    if arrBerat[i] > *bMax {
        *bMax = arrBerat[i]
    }
}
}

func rata_rata(arrBerat arrBalita, n int) float64 {
    var total float64
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }
    return total / float64(n)
}

func main() {
    var n int
    var berat arrBalita

    fmt.Println("Masukkan banyak data berat balita: ")
    fmt.Scan(&n)

    if n <= 0 || n > 100 {
        fmt.Println("Jumlah data berat balita harus antara 1 hingga 100.")
        return
    }

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

    var bMin, bMax float64
    hitungMinMax(berat, n, &bMin, &bMax)
    rata := rata_rata(berat, n)

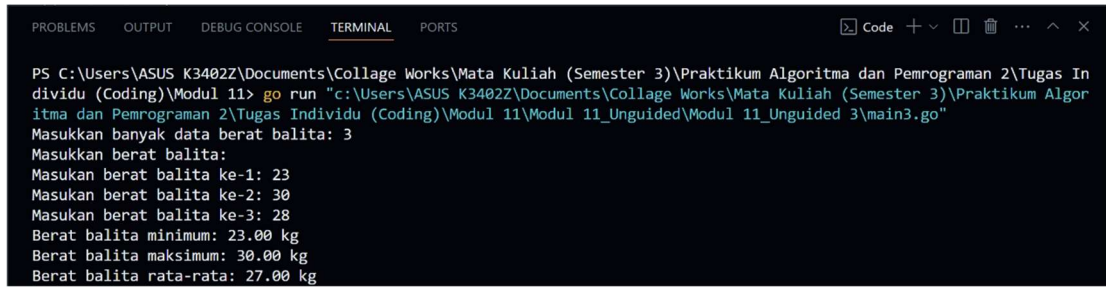
    fmt.Printf("Berat balita minimum: %.2f kg\n", bMin)
    fmt.Printf("Berat balita maksimum: %.2f kg\n", bMax)
    fmt.Printf("Berat balita rata-rata: %.2f kg\n", rata)
}

```



```
}
```

Screenshoot Output



```
PS C:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11> go run "c:\Users\ASUS K3402Z\Documents\Collage Works\Mata Kuliah (Semester 3)\Praktikum Algoritma dan Pemrograman 2\Tugas Individu (Coding)\Modul 11\Modul 11_Unguided\Modul 11_Unguided 3\main3.go"
Masukkan banyak data berat balita: 3
Masukkan berat balita:
Masukan berat balita ke-1: 23
Masukan berat balita ke-2: 30
Masukan berat balita ke-3: 28
Berat balita minimum: 23.00 kg
Berat balita maksimum: 30.00 kg
Berat balita rata-rata: 27.00 kg
```

Deskripsi Program

Program di atas adalah program untuk mencari berat badan minimum, maksimum, dan rata-rata. Program tersebut akan meminta pengguna untuk memasukkan jumlah balita, dan juga berat badan balita. Program ini menggunakan prosedur dan juga fungsi untuk mencari berat minimum, maksimum, dan rata-rata balita berdasarkan masukan pengguna.