

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XI**

**PENCARIAN NILAI EKSTREM PADA HIMPUNAN DATA**



**Disusun Oleh :**

**Hamzah Ziyad Ibadurrohman / 2311102254**

**IF-11-06**

**Dosen Pengampu :**

**Abednego Dwi Septiadi**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. DASAR TEORI

### Dasar Teori

#### 10.1 Ide Pencarian Nilai Max/Min

Pencarian adalah suatu proses yang lazim dilakukan di dalam kehidupan sehari-hari. Contoh penggunaannya dalam kehidupan nyata sangat beragam, misalnya pencarian file di dalam directory komputer, pencarian suatu teks di dalam sebuah dokumen, pencarian buku pada rak buku, dan contoh lainnya. Pertama pada modul ini akan dipelajari salah satu algoritma pencarian nilai terkecil atau terbesar pada sekumpulan data, atau biasa disebut pencarian nilai ekstrim.

Ide algoritma sederhana sekali. Karena data harus diproses secara sekuensial, maka nilai atau indeks ke nilai maksimum dari data yang telah diproses disimpan untuk dibandingkan dengan data berikutnya. Nilai yang berhasil disimpan sampai algoritma tersebut berakhir adalah nilai maksimum yang dicari. Adapun algoritmanya secara umum adalah sebagai berikut:

1. Jadikan Data pertama menjadi nilai ekstrim
2. Lakukan validasi nilai ekstrim dari data kedua hingga data terakhir. Apabila nilai ekstrim tidak valid, maka update nilai ekstrim tersebut dengan data yang dicek
3. Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid.

Berikut ini adalah notasi dalam psuedocode dan bahasa Go, misalnya untuk pencarian nilai terbesar atau maksimum:

	Notasi Algoritma	Notasi dalam bahasa Go
1	$\text{max} \leftarrow 1$	$\text{max} = 0$
2	$i \leftarrow 2$	$i = 1$
3	while $i \leq n$ do	for $i < n$ {
4	if $a[i] > a[\text{max}]$ then	if $a[i] > a[\text{max}]$ {
5	$\text{max} \leftarrow i$	$\text{max} = i$
6	endif	}
7	$i \leftarrow i + 1$	$i = i + 1$
8	endwhile	}

#### 10.2 Pencarian Nilai Ekstrim pada Array Bertipe Data Dasar

Misalnya terdefinisi sebuah array of integer dengan kapasitas 2023, dan array terisi sejumlah N bilangan bulat, kemudian pencarian nilai terkecil dilakukan pada array tersebut. Perhatikan potongan program dalam bahasa Go berikut ini!

```

5  type arrInt [2023]int
..  ...
15
16 func terkecil_1(tabInt arrInt, n int) int {
17  /* mengembalikan nilai terkecil yang terdapat di dalam tabInt yang berisi n
18  bilangan bulat */
19      var min int = tabInt[0]           // min berisi data pertama
20      var j int = 1                     // pencarian dimulai dari data berikutnya
21      for j < n {
22          if min > tabInt[j] {           // pengecekan apakah nilai minimum valid
23              min = tabInt[j]           // update nilai minimum dengan yang valid
24          }
25          j = j + 1
26      }
27      return min                         // returnkan nilai minimumnya
28  }

```

Potongan program di atas sedikit berbeda dengan sebelumnya karena penggunaan indeks array pada bahasa Go di mulai dari nol atau "0" seperti penjelasan pada modul 9. Selanjutnya, pada penjelasan di awal bab 3 telah disampaikan bahwa pada pencarian yang terpenting adalah posisi atau indeks dari nilai yang dicari dalam kumpulan data atau array. Oleh karena itu modifikasi pada program di atas dapat dilihat pada potongan program berikut ini!

```

..  ...
5  type arrInt [2023]int
..  ...
15
16 func terkecil_2(tabInt arrInt, n int) int {
17  /* mengembalikan indeks nilai terkecil yang terdapat di dalam tabInt yang berisi
18  n bilangan bulat */
19      var idx int = 0                   // idx berisi indeks data pertama
20      var j int = 1                     // pencarian dimulai dari data berikutnya
21      for j < n {
22          if tabInt[idx] > tabInt[j] {   // pengecekan apakah nilai minimum valid
23              idx = j                     // update nilai minimum dengan yang valid
24          }
25          j = j + 1
26      }
27      return idx                         // returnkan indeks nilai minimumnya
28  }

```



## II. GUIDED

### 1. Soal Studi Case

Suatu lingkaran didefinisikan dengan koordinat titik pusat (cx, cy) dengan radius r. Apabila diberikan dua buah lingkaran, maka tentukan posisi sebuah titik sembarang (x, y) berdasarkan dua lingkaran tersebut. Gunakan tipe bentukan titik untuk menyimpan koordinat, dan tipe bentukan lingkaran untuk menyimpan titik pusat lingkaran dan radiusnya.

Masukan terdiri dari beberapa tiga baris. Baris pertama dan kedua adalah koordinat titik pusat dan radius dari lingkaran 1 dan lingkaran 2, sedangkan baris ketiga adalah koordinat titik sembarang. Asumsi sumbu x dan y dari semua titik dan juga radius direpresentasikan dengan bilangan bulat.

Keluaran berupa string yang menyatakan posisi titik "Titik di dalam lingkaran 1 dan 2", "Titik di dalam lingkaran 1", "Titik di dalam lingkaran 2", atau "Titik di luar lingkaran 1 dan 2".

#### 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
        }

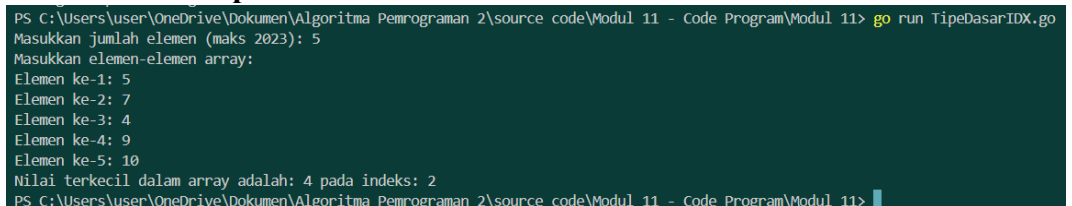
        // Memasukkan elemen-elemen array
        fmt.Println("Masukkan elemen-elemen array:")
        for i := 0; i < n; i++ {
            fmt.Print("Elemen ke-", i+1, ": ")
            fmt.Scan(&tab[i])
        }

        // 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)
    }

```

## Screenshot Output



```

PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11> go run TipeBesarIDX.go
Masukkan jumlah elemen (maks 2023): 5
Masukkan elemen-elemen array:
Elemen ke-1: 5
Elemen ke-2: 7
Elemen ke-3: 4
Elemen ke-4: 9
Elemen ke-5: 10
Nilai terkecil dalam array adalah: 4 pada indeks: 2
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11>

```

## Deskripsi Program

Ini adalah Program yang mencari nilai ekstrem terkecil . Program ini memakai array yang berkapasitas 2023 integer. Cara kerjanya program meminta jumlah data yang akan dimasukkan ke array. Jumlah data tersebut harus tidak kurang dari 1 dan tidak lebih dari 2023(kapasitas array). Kemudian program memakai for looping untuk di setiap iterasi, program akan meminta input elemen. Lalu setelah itu program akan memanggil fungsi terkecil. Fungsi terkecil digunakan untuk menemukan indeks elemen terkecil dalam array. Fungsi ini menerima input berupa array dan jumlah elemen (n), lalu menggunakan loop untuk membandingkan setiap elemen. Indeks elemen terkecil disimpan dalam variabel idx, yang diperbarui jika ditemukan elemen dengan nilai lebih kecil dari elemen sebelumnya. Fungsi

mengembalikan indeks dari elemen terkecil tersebut. Lalu setelah itu Program kemudian menampilkan nilai terkecil beserta indeksnya.

## 2. Soal Studi Case

```
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)
}

```

## Screenshot Output



```

PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11> go run TipeStrukturIPK_IDX.go
Masukkan jumlah mahasiswa (maks 2023): 2

Masukkan data mahasiswa ke-1
Nama: hamzah
NIM: 254
Kelas: if1
Jurusan: informatika
IPK: 3.6

Masukkan data mahasiswa ke-2
Nama: ziyad
NIM: 137
Kelas: if1
Jurusan: informatika
IPK: 3.9

Mahasiswa dengan IPK tertinggi:
Nama: ziyad
NIM: 137
Kelas: if1
Jurusan: informatika
IPK: 3.90
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11> 

```

### Deskripsi Program

Ini adalah program yang dibuat untuk mengelola data mahasiswa yang mencakup nama, NIM, kelas, jurusan, dan IPK, serta mencari mahasiswa dengan IPK tertinggi. Data setiap mahasiswa disimpan dalam sebuah struct bernama mahasiswa, yang terdiri dari atribut nama, nim, kelas, jurusan bertipe string, dan ipk bertipe float64. Array dengan tipe data arrMhs digunakan untuk menyimpan maksimal 2023 data mahasiswa. Program ini memiliki fungsi utama indeksIPKTertinggi, yang menerima array mahasiswa dan jumlah elemen sebagai input, kemudian menggunakan pencarian linear untuk menemukan indeks mahasiswa dengan IPK tertinggi. Indeks ini diperbarui jika ditemukan IPK yang lebih tinggi selama proses iterasi, dan hasil akhirnya adalah indeks mahasiswa dengan IPK tertinggi.

Fungsi main berfungsi untuk mengelola input dan output program. Pertama, program meminta jumlah mahasiswa (n) dari pengguna dan memvalidasinya agar berada dalam rentang 1 hingga 2023. Jika valid, program melanjutkan dengan meminta pengguna mengisi data setiap mahasiswa melalui iterasi, yang mencakup nama, NIM, kelas, jurusan, dan IPK. Setelah semua data dimasukkan, fungsi indeksIPKTertinggi dipanggil untuk menentukan mahasiswa dengan IPK tertinggi. Program kemudian menampilkan data mahasiswa tersebut, termasuk nama, NIM, kelas, jurusan, dan IPK.

### III. UNGUIDED

1. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

### 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 riil berikutnya adalah berat dari anak kelinci yang akan dijual.

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

### Sourcecode

```
package main

import "fmt"

func carinilaiminmaks(data []float64, n int) (float64, float64) {

    min := data[0]
    maks := data[0]

    for i := 1; i < n; i++ {
        if data[i] < min {
            min = data[i]
        }
        if data[i] > maks {
            maks = data[i]
        }
    }
    return min, maks
}

func main() {
    var n int
    var berat [1000]float64

    fmt.Print("Jumlah anak kelinci berapa?: ")
    fmt.Scan(&n)

    if n < 1 || n > 1000 {
        fmt.Println("jumlah harusnya 1 sampai 1000.")
        return
    }
    fmt.Println("Berat kelinci masukkan : ")
    for i := 0; i < n; i++ {
```

```

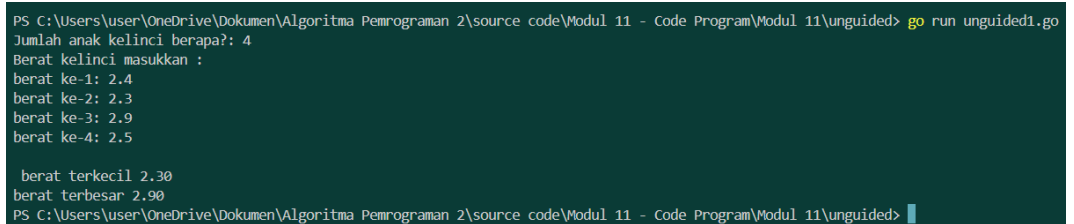
        fmt.Printf("berat ke-%d: ", i+1)
        fmt.Scan(&berat[i])
    }

    min, maks := carinilaiminmaks(berat[:n], n)

    fmt.Printf("\n berat terkecil %.2f\n", min)
    fmt.Printf("berat terbesar %.2f\n", maks)
}

```

## Screenshoot Output



```

PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided> go run unguided1.go
Jumlah anak kelinci berapa?: 4
Berat kelinci masukkan :
berat ke-1: 2.4
berat ke-2: 2.3
berat ke-3: 2.9
berat ke-4: 2.5

    berat terkecil 2.30
    berat terbesar 2.90
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided>

```

## Deskripsi Program

Ini adalah program yang dibuat untuk mencari berat anak kelinci terkecil dan terbesar dari sekumpulan data yang diberikan oleh pengguna. Fungsi yang digunakan untuk mencari nilai adalah fungsi `carinilaiminmaks`. Fungsi ini menginisialisasi nilai minimum (`min`) dan maksimum dengan elemen pertama array, kemudian menggunakan iterasi untuk memeriksa setiap elemen berikutnya. Jika ditemukan nilai lebih kecil dari `min`, maka `min` diperbarui; begitu pula jika ditemukan nilai lebih besar dari `max`, maka `max` diperbarui. Setelah seluruh elemen diperiksa, fungsi mengembalikan nilai minimum dan maksimum.

Di fungsi `main`, program meminta pengguna untuk memasukkan jumlah anak kelinci (`n`) yang ingin dihitung beratnya, dengan validasi untuk memastikan jumlah tersebut berada dalam rentang 1 hingga 1000. Selanjutnya, program meminta pengguna memasukkan data berat untuk setiap anak kelinci. Data ini disimpan dalam array `berat`. Setelah data diinput, fungsi `carinilaiminmaks` dipanggil untuk menghitung berat terkecil dan terbesar, kemudian hasilnya ditampilkan dengan format dua desimal.

### 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 dijual, sedangkan y adalah banyaknya ikan yang akan yang dimasukkan ke dalam wadah. baris kedua terdiri dari sejumlah x bilangan riil yang menyatakan banyaknya ikan yang akan dijual. keluaran terdiri dari dua baris. baris pertama adalah kumpulan bilangan riil 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 riil yang menyatakan berat rata-rata ikan di setiap wadah..

### Sourcecode

```
package main

import "fmt"

func main(){

    var x, y int

    var beratikan [1000] float64

    fmt.Print("Jumlah ikan dijual berapa?: (x) dan jumlah ikan
per wadah ?(y) : ")

    fmt.Scan(&x, &y)

    if x <1 || x > 1000 || y < 1{

        fmt.Println("jumlah nya harus dari 1 hingga 1000!.
jumlah minimum wadah itu 1.")

    }

    fmt.Println("berat ikan: ")

    for i:= 1; i < x; i++){

        fmt.Printf("berat ikan %d: ", i+1)

        fmt.Scan(&beratikan[i])
```

```

    }

    var totalberat []float64
    var total float64
    for i:= 1; i < x; i++){
        total +=beratikan[i]
        if (i+1)%y == 0 || i == x-1 {
            totalberat = append(totalberat,total)
            total = 0
        }
    }

    var rata float64
    for _, berat := range totalberat {
        rata += berat
    }
    rata /= float64(len(totalberat))

    fmt.Println("total berat setiap wadah: ")
    for i, berat := range totalberat {
        fmt.Printf("wadah %d: %.2f\n", i+1, berat)
    }

    fmt.Printf("berat rata-rata ikan setiap wadah: %.2f\n",
rata)

```

```
}
```

## Screenshot output

```
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided> go run unguided2.go
Jumlah ikan dijual berapa?: (x) dan jumlah ikan per wadah?(y) : 5 2
berat ikan:
berat ikan 1: 3.2
berat ikan 2: 2.6
berat ikan 3: 1.9
berat ikan 4: 3.0
berat ikan 5: 4
total berat setiap wadah:
wadah 1: 5.80
wadah 2: 4.90
wadah 3: 4.00
berat rata-rata ikan setiap wadah: 4.90
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided>
```

## Deskripsi Output

Ini adalah Program golang yang dibuat untuk menghitung total berat ikan di setiap wadah dan rata-rata berat ikan per wadah, berdasarkan jumlah ikan yang dijual dan kapasitas ikan per wadah. Ketika Program dimulai, program meminta input dua bilangan bulat, yaitu x (jumlah ikan yang dijual) dan y (jumlah ikan per wadah). Validasi dilakukan untuk memastikan bahwa x berada dalam rentang 1 hingga 1000 dan y minimal bernilai 1, agar data dapat diproses dengan benar. Setelah itu, program meminta input berat setiap ikan sebanyak x elemen, yang disimpan dalam array. Selanjutnya, program menghitung total berat ikan per wadah menggunakan sebuah loop. Setiap berat ikan ditambahkan ke variabel total hingga jumlah ikan dalam wadah mencapai kapasitas y atau hingga ikan terakhir diproses. Ketika wadah penuh atau ikan terakhir telah diproses, total berat untuk wadah tersebut disimpan dalam slice totalBerat, dan variabel total direset ke nol untuk menghitung berat wadah berikutnya. Setelah semua ikan diproses, program menghitung rata-rata berat ikan per wadah dengan menjumlahkan total berat dari semua wadah dan membaginya dengan jumlah wadah (len(totalBerat)). Hasil akhir berupa dua keluaran: total berat ikan di setiap wadah dan berat rata-rata ikan per wadah, yang ditampilkan dengan format dua desimal

### 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.

Buatlah program golang dengan spesifikasi subprogram seperti yang di gambar berikut.

### Sourcecode

```
package main

import "fmt"

type arrbalita [100]float64

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 hitungmindanmaks(arrberat arrbalita, n int, bmin,
bmaks *float64) {
    *bmin, *bmaks = arrberat[0], arrberat[0]
    for i := 1; i < n; i++ {
        if arrberat[i] < *bmin {
            *bmin = arrberat[i]
        }
        if arrberat[i] > *bmaks {
```

```

        *bmaks = arrberat[i]

    }

}

func main() {

    var databerat arrbalita

    var n int

    var bmin, bmaks float64

    fmt.Print("Input data banyak berat balita: ")

    fmt.Scan(&n)

    if n < 1 || n > 100 {

        fmt.Println("data harus dari 1 hingga 100.
        kalau kelebihan kapasitas kepenuhan.")

        return

    }

    for i := 0; i < n; i++ {

        fmt.Printf("berat balita ke-%d: ", i+1)

        fmt.Scan(&databerat[i])

    }

    hitungmindanmaks(databerat, n, &bmin, &bmaks)

    rata := rerata(databerat, n)

    fmt.Printf("berat balita minimum: %.2f\n", bmin)

    fmt.Printf("berat balita maksimum: %.2f\n",
    bmaks)

```



```
fmt.Printf("rata-rata berat: %.2f\n", rata)

}
```

## Screenshot Output

```
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided> go run unguided3.go
Input data banyak berat balita: 5
berat balita ke-1: 6.0
berat balita ke-2: 5.7
berat balita ke-3: 7.5
berat balita ke-4: 7.7
berat balita ke-5: 8.0
berat balita minimum: 5.70
berat balita maksimum: 8.00
rata-rata berat: 6.98
PS C:\Users\user\OneDrive\Dokumen\Algoritma Pemrograman 2\source code\Modul 11 - Code Program\Modul 11\unguided> █
```

## Deskripsi Program

Ini adalah program yang dibuat untuk mencatat berat balita dan menentukan berat terkecil, terbesar, serta rata-rata berat balita menggunakan array. tipe data `arrBalita` dideklarasikan sebagai array yang mampu menampung hingga 100 elemen bertipe `float64`. Program memiliki dua fungsi utama: `hitungMinMax` dan `hitungRerata`. Fungsi `hitungMinMax` bertugas mencari nilai berat minimum dan maksimum dalam array, dengan cara menginisialisasi nilai awal minimum (`bmin`) dan maksimum (`bmaks`) menggunakan elemen pertama array, lalu membandingkannya dengan elemen-elemen lainnya. fungsi `Rerata` menghitung rata-rata berat balita dengan menjumlahkan seluruh elemen array dan membaginya dengan jumlah elemen (`n`). Kemudian di bagian utama program, Program meminta pengguna untuk memasukkan jumlah data balita (`n`) dan memvalidasi bahwa nilai `n` berada dalam rentang 1 hingga 100, sesuai kapasitas array. Kemudian, program meminta input berat balita satu per satu dan menyimpannya dalam array `dataBerat`. Setelah semua data dimasukkan, fungsi `hitungMinMax` dipanggil untuk menentukan berat minimum dan maksimum, sementara fungsi `hitungRerata` menghitung rata-rata berat balita. Hasilnya ditampilkan dengan format dua angka desimal