

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XI
PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA**



Disusun Oleh :

Liya Khoirunnisa / 2311102124

IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

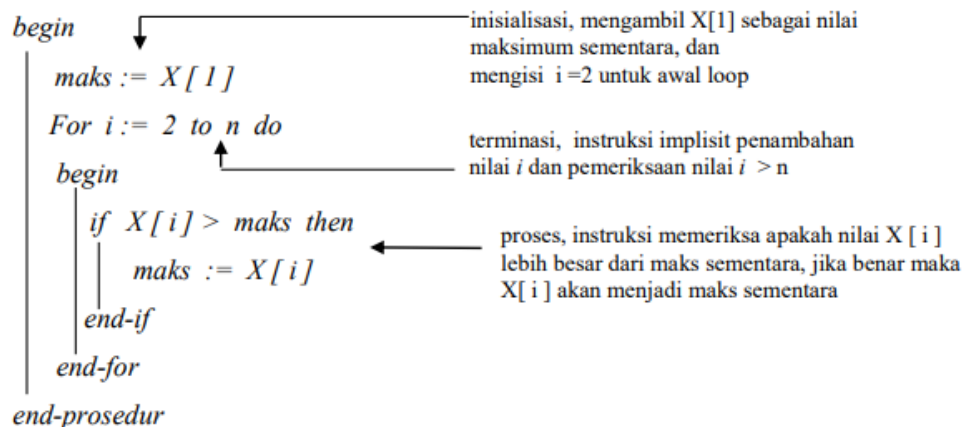
Pencarian merupakan proses yang sering dilakukan di kehidupan sehari-hari. Salah satu algoritma yang dipelajari yaitu metode pencarian nilai ekstrim dengan melakukan pencarian nilai terbesar atau terkecil dalam sekumpulan data. Algoritma ini diproses secara berurutan, nilai atau indeks maksimum pada data akan disimpan untuk dibandingkan dengan data berikutnya. Nilai yang tersisa adalah nilai maksimum yang dicari. Berikut langkah-langkah algoritma ini :

1. Menetapkan data pertama sebagai nilai ekstrim sementara.
2. Periksa nilai ekstrim terhadap data kedua hingga terakhir.
3. Setelah semua data diperiksa, nilai ekstrim yang tersisa adalah valid.

Contoh prosedur dalam bentuk pseudocode :

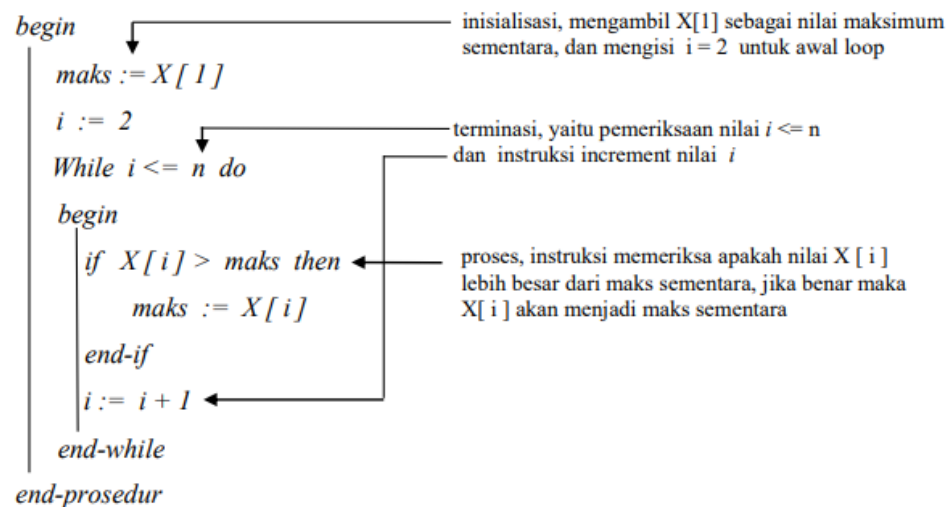
Jika menggunakan loop for

Prosedur NilaiMax(n : jumlah elemen)



Jika menggunakan while-do

Prosedur NilaiMax(n : jumlah elemen)



II. UNGUIDED

1. 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.

Source code

```
/* Liya Khoirunnisa - 2311102124 */
package main

import "fmt"

// Fungsi untuk menghitung berat minimum dan maksimum
func hitungBeratKelinci_124(berat []float64, jumlahData
int, beratMin, beratMax *float64) {
    *beratMin = berat[0] // Inisialisasi berat minimum
    *beratMax = berat[0] // Inisialisasi berat maksimum

    // Perulangan untuk mencari berat kelinci yang minimum
    dan maksimum
    for i := 1; i < jumlahData; i++ {
        if berat[i] < *beratMin { // Jika berat kelinci
        lebih kecil dari berat minimum
            *beratMin = berat[i] // Update berat minimum
        }
        if berat[i] > *beratMax { // Jika berat kelinci
        lebih besar dari berat maksimum
            *beratMax = berat[i] // Update berat maksimum
        }
    }
}

func main() {
    // Deklarasi variabel
    var jumlahKelinci int

    // Menggunakan slice untuk data berat kelinci
    var dataKelinci [1000]float64 // Array dengan
    kapasitas 1000

    // Deklarasi variabel
    var beratMin, beratMax float64

    // Meminta input jumlah data berat kelinci
    fmt.Print("Masukkan jumlah data berat kelinci: ")
    fmt.Scan(&jumlahKelinci)

    // Validasi jumlah data kelinci
```

```

        if jumlahKelinci < 1 || jumlahKelinci > 1000 {
            fmt.Println("Jumlah data harus antara 1 dan
1000.")
            return
        }

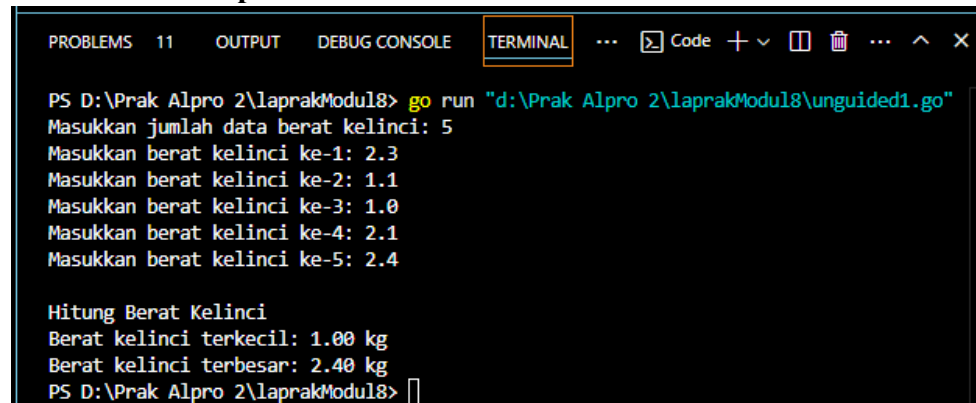
        // Input berat kelinci
        for i := 0; i < jumlahKelinci; i++ {
            fmt.Printf("Masukkan berat kelinci ke-%d: ", i+1)
            fmt.Scan(&dataKelinci[i])
        }

        // Menampilkan hasil perhitungan berat minimum dan
maksimum
        hitungBeratKelinci_124(dataKelinci[:], jumlahKelinci,
&beratMin, &beratMax)

        // Menampilkan hasil
        fmt.Print("\nHitung Berat Kelinci\n")
        fmt.Printf("Berat kelinci terkecil: %.2f kg\n",
beratMin)
        fmt.Printf("Berat kelinci terbesar: %.2f kg\n",
beratMax)
    }

```

Screenshot Output



```

PS D:\Prak Alpro 2\laprakModul8> go run "d:\Prak Alpro 2\laprakModul8\unguided1.go"
Masukkan jumlah data berat kelinci: 5
Masukkan berat kelinci ke-1: 2.3
Masukkan berat kelinci ke-2: 1.1
Masukkan berat kelinci ke-3: 1.0
Masukkan berat kelinci ke-4: 2.1
Masukkan berat kelinci ke-5: 2.4

Hitung Berat Kelinci
Berat kelinci terkecil: 1.00 kg
Berat kelinci terbesar: 2.40 kg
PS D:\Prak Alpro 2\laprakModul8>

```

Deskripsi Program

Program di atas dibuat untuk menentukan berat terkecil dan terbesar dari jumlah anak kelinci yang diinputkan pengguna. Program diawali dengan meminta input jumlah kelinci, dengan rentang jumlah kelinci adalah 1 hingga 1000. Setelah itu, pengguna diminta untuk menginputkan berat masing-masing anak kelinci yang disimpan dalam array. Terdapat fungsi `hitungBeratKelinci_124` untuk menentukan berat terkecil dan terbesar. Setiap iterasi, berat kelinci dibandingkan untuk menemukan berat yang lebih kecil atau lebih besar. Kemudian program akan menampilkan hasil

berupa berat terkecil dan terbesar dalam format desimal dengan dua angka dibelakang koma.

***Note:** Pada terminal terdapat 11 problem dikarenakan package dan func main digunakan juga di file lain

2. 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 dimasukan 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 dimasukan 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.

Source code

```
/* Liya Khoirunnisa - 2311102124 */
package main

import (
    "fmt"
)

// Fungsi untuk menghitung total berat ikan di setiap
wadah
func hitungTotalBerat_124(wadah []float64, ikan []float64,
totalIkan, kapasitasWadah int) {
    for i := 0; i < totalIkan; i += kapasitasWadah {
        var total float64
        for j := i; j < i+kapasitasWadah && j < totalIkan;
j++ {
            total += ikan[j]
        }
        wadah[i/kapasitasWadah] = total // Simpan total
berat ke wadah
    }
}

// Fungsi untuk menghitung berat rata-rata ikan di setiap
wadah
func hitungRataRata(wadah []float64, totalIkan,
kapasitasWadah int) []float64 {
```

```

var rataRata []float64
for i := 0; i < len(wadah); i++ {
    count := 0
    for j := i * kapasitasWadah; j <
(i+1)*kapasitasWadah && j < totalIkan; j++ {
        count++
    }
    if count > 0 {
        rataRata = append(rataRata,
wadah[i]/float64(count))
    } else {
        rataRata = append(rataRata, 0)
    }
}
return rataRata
}

func main() {
    // Deklarasi variabel
    var x, y int

    // Meminta input jumlah ikan dan kapasitas maksimum
dalam wadah
    fmt.Print("Masukkan total ikan dan kapasitas maksimum
dalam wadah: ")
    fmt.Scan(&x, &y)

    // Cek jika x lebih dari kapasitas array
    if x > 1000 {
        fmt.Println("Jumlah ikan tidak boleh lebih dari
1000.")
        return
    }

    // Membuat array untuk menyimpan berat ikan
    ikan := make([]float64, x)

    // Meminta input berat ikan satu per satu
    fmt.Println("Masukkan berat masing-masing ikan:")
    for i := 0; i < x; i++ {
        // Menampilkan prompt input dengan nomor ikan
        fmt.Printf("Berat ikan ke-%d: ", i+1)
        fmt.Scan(&ikan[i])
    }

    // Menghitung jumlah wadah yang diperlukan
    jumlahWadah := (x + y - 1) / y
    totalBeratWadah := make([]float64, jumlahWadah)

    // Memanggil fungsi untuk menghitung total berat ikan
di setiap wadah
    hitungTotalBerat_124(totalBeratWadah, ikan, x, y)

    // Menampilkan total berat ikan di setiap wadah
    fmt.Println("\nTotal berat ikan di setiap wadah:")
    for i, total := range totalBeratWadah {

```

```

        fmt.Printf("Wadah %d: %.2f kg\n", i+1, total)
    }

    // Menghitung rata-rata berat ikan di setiap wadah
    rataRata := hitungRataRata(totalBeratWadah, x, y)

    // Menampilkan berat rata-rata ikan di setiap wadah
    fmt.Println("\nBerat rata-rata ikan di setiap wadah:")
    for i, rata := range rataRata {
        fmt.Printf("Wadah %d: %.2f kg\n", i+1, rata)
    }
}

```

Screenshoot Output

```

PS D:\Prak Alpro 2\laprakModul8> go run "d:\Prak Alpro 2\laprakModul8\unguided2.go"
Masukkan total ikan dan kapasitas maksimum dalam wadah: 5 2
Masukkan berat masing-masing ikan:
Berat ikan ke-1: 2.3
Berat ikan ke-2: 1.1
Berat ikan ke-3: 1.0
Berat ikan ke-4: 2.1
Berat ikan ke-5: 2.4

Total berat ikan di setiap wadah:
Wadah 1: 3.40 kg
Wadah 2: 3.10 kg
Wadah 3: 2.40 kg

Berat rata-rata ikan di setiap wadah:
Wadah 1: 1.70 kg
Wadah 2: 1.55 kg
Wadah 3: 2.40 kg
PS D:\Prak Alpro 2\laprakModul8>

```

Deskripsi Program

Program di atas dibuat untuk menghitung total berat dan berat rata-rata ikan dalam wadah. Terdapat 2 fungsi yang digunakan yaitu `hitungTotalBerat_124` dan `hitung rata-rata`. Program diawali dengan meminta input jumlah ikan dan kapasitas maksimum ikan dalam wadah dengan batas maksimal 1000 data. Jika input tidak valid, program akan menampilkan pesan kesalahan dan berhenti. Setelah itu, pengguna diminta untuk menginputkan berat masing-masing ikan. Untuk menghitung total berat ikan dalam setiap wadah memanggil fungsi `hitungTotalBerat_124`. Hasil perhitungan disimpan dalam array. Untuk menghitung rata-rata berat ikan pada setiap wadah memanggil fungsi `hitungRataRata` dengan menjumlahkan semua data dan membagi dengan banyak data. Program akan menampilkan total berat dan rata-rata berat dalam setiap wadah dalam format desimal dengan dua angka dibelakang koma.

***Note:** Pada terminal terdapat 2 problem dikarenakan package dan func main digunakan juga di file lain

3. 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 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

```
/* Liya Khoirunnisa - 2311102124 */
package main

import "fmt"

// Fungsi untuk menghitung berat minimum dan maksimum
func hitungBeratBalita_124(berat []float64, jumlahData
int, beratMin, beratMax *float64) {
    *beratMin = berat[0] // Inisialisasi berat minimum
    *beratMax = berat[0] // Inisialisasi berat maksimum

    // Perulangan untuk mencari berat balita yang minimum
    dan maksimum
    for i := 1; i < jumlahData; i++ {
        if berat[i] < *beratMin { // Jika berat balita
lebih kecil dari berat minimum
            *beratMin = berat[i] // Update berat minimum
        }
        if berat[i] > *beratMax { // Jika berat balita
lebih besar dari berat maksimum
            *beratMax = berat[i] // Update berat maksimum
        }
    }
    // Menampilkan hasil berat minimum dan maksimum
```



```

        fmt.Printf("Berat balita minimum: %.2f kg\n",
*beratMin)
        fmt.Printf("Berat balita maksimum: %.2f kg\n",
*beratMax)
    }

    // Fungsi untuk menghitung rata-rata berat balita
    func hitungRataRata(berat []float64, jumlahData int)
float64 {
        totalBerat := 0.0 // Variabel untuk menyimpan total
berat balita

        // Perulangan untuk menghitung total berat dari semua
balita
        for i := 0; i < jumlahData; i++ {
            totalBerat += berat[i]
        }

        // Menghitung rata-rata berat balita
        rataRata := totalBerat / float64(jumlahData)

        // Menampilkan hasil rata-rata berat
        fmt.Printf("Rata-rata berat balita: %.2f kg\n",
rataRata)
        return rataRata // Mengembalikan rata-rata berat
    }

    func main() {
        // Deklarasi variabel
        var jumlahBalita int

        // Menggunakan slice untuk data berat balita
        var dataBalita []float64

        // Deklarasi variabel
        var beratMin, beratMax float64

        // Meminta input jumlah data berat balita
        fmt.Print("Masukkan jumlah data berat balita: ")
        fmt.Scan(&jumlahBalita)

        // Validasi jumlah data balita
        if jumlahBalita > 100 {
            fmt.Println("Jumlah data tidak boleh lebih dari
100.")
            return
        }

        // slice dengan panjang sesuai jumlah data
        dataBalita = make([]float64, jumlahBalita)

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

```

```

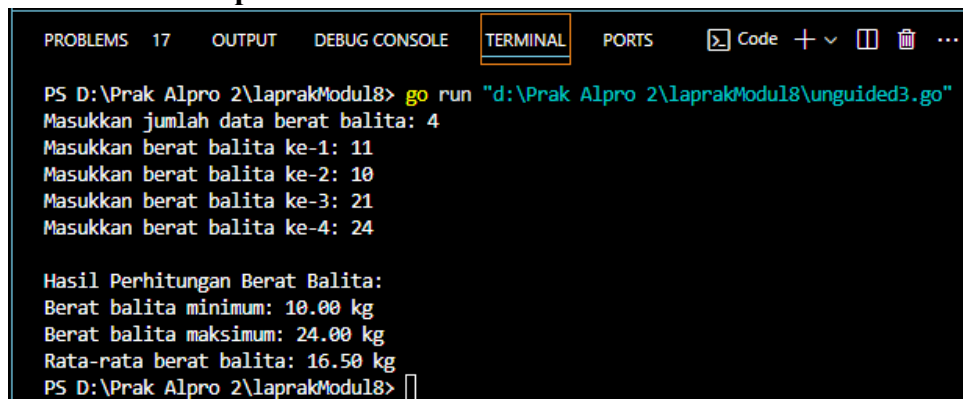
        // Menampilkan hasil perhitungan berat minimum,
maksimum, dan rata-rata
        fmt.Println("\nHasil Perhitungan Berat Balita: ")

        // Panggil fungsi untuk menghitung berat minimum dan
maksimum
        hitungBeratBalita_124(dataBalita, jumlahBalita,
&beratMin, &beratMax)

        // Panggil fungsi untuk menghitung dan menampilkan
rata-rata berat
        hitungRataRata(dataBalita, jumlahBalita)
    }

```

Screenshoot Output



```

PS D:\Prak Alpro 2\laprakModul8> go run "d:\Prak Alpro 2\laprakModul8\unguided3.go"
Masukkan jumlah data berat balita: 4
Masukkan berat balita ke-1: 11
Masukkan berat balita ke-2: 10
Masukkan berat balita ke-3: 21
Masukkan berat balita ke-4: 24

Hasil Perhitungan Berat Balita:
Berat balita minimum: 10.00 kg
Berat balita maksimum: 24.00 kg
Rata-rata berat balita: 16.50 kg
PS D:\Prak Alpro 2\laprakModul8>

```

Deskripsi Program

Program di atas dibuat untuk menganalisis data berat balita. Terdapat 3 fungsi yaitu `hitungBeratBalita` dan `hitungRataRata`. Program diawali dengan meminta input jumlah balita dengan batas maksimal 100 data. Jika input tidak valid, program akan menampilkan pesan kesalahan dan berhenti. Setelah itu, pengguna diminta untuk menginputkan berat masing-masing balita yang disimpan dalam slice. Untuk menghitung berat minimum dan maksimum memanggil fungsi `hitungBeratBalita` dengan membandingkan setiap elemen dengan nilai minimum dan maksimum dan memperbarui nilai jika ditemukan nilai kecil atau lebih besar. Untuk menghitung rata-rata berat memanggil fungsi `hitungRataRata` dengan menjumlahkan semua data dan membagi dengan banyak data. Program akan menampilkan hasil berupa berat terkecil dan terbesar dan rata-rata dalam format desimal dengan dua angka dibelakang koma.

***Note:** Pada terminal terdapat 17 problem dikarenakan package dan func main digunakan juga di file lain.

DAFTAR PUSTAKA

- [1] Universitas Esa Unggul. (2019). *Introduction to Computer Systems*. Diakses pada 20 November 2024, dari https://lms-paralel.esaunggul.ac.id/pluginfile.php?file=/186856/mod_resource/content/1/037228_ccs120_032019_pdf.pdf