# LAPORAN PRAKTIKUM
# ALGORITMA DAN PEMROGRAMAN 2

## MODUL XI

## PENCARIAN NILAI EKSTIM PADA HIMPUNAN DATA


**Disusun Oleh :**

**Deshan Rafif Alfarisi / 2311102326**

**S1-IF-11-06**

**Dosen Pengampu :**

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

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**


**2024**

# I. DASAR TEORI

## 1. DASAR Ide Pencarian Nllal 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 sebagai nilai ekstrim
2) Lakukan validasi nilai ekstrim dari data kedua hingga data terakhir.
   - Apabila nilai ekstrim tidak valid, maka update nilai ekstrims tersebut dengan data yangdicek.
3) Apabila semua data telah dicek, maka nilai ekstrim yang dimiliki adalah valid.



Berikut ini adalah notasi dalam pseudocode dan bahasa Go, misalnya untuk pencarian nilai terbesar ataumaksimum:

| | Notasl A1gorltma | Notas1 datam bahasa Go |
|---|---|---|
| 1 | max U | max 0 |
| 2 | i -2 | i 1 |
| 3 | while i•= n do | for i • n { |
| 4 | if a[i] » a[max] then | if a[i] » a[max] {max i |
| 5 | max + i | } |
| 6 | endif | i i+1 |
| 7 | i + i+1 | } |
| 8 | endwhile | |

## 2. Pencarian Nllal Ekstrlm pada Array Bertlpe 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 terkeci1_1 (tablnt arrlnt, n int) int (
17
18    /+ inengeinba11kan n11al  terkeciJ yang terdapat d1 da1aiii tabznt yang ber1s1 n
19        var min int = tablnt[0]          // in1n bert st data pertaina
20
21        var j int    1
22                                         // pengecekan apakah n11a1 mlnlnum valld
23        lo r j < n {
24
25            j   j + 1
26
27        }
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 te rkeci1_2(tablnt arrlnt, n int) int (
17
18    /+ nengeinba11kan Indeks n11a1 terkeclJ yang terdapat d1 dalam tabznt yang ber1s1
19        var idx int = 0                  // 1dx bert st Indeks data pertana
20        var j int   7
21        fo r j « n {
22                                         { // pengecekan apakah n1fat n1n1nuin va1:id
23            if  tablnt [ idx] » tablnt [ j ]
24                idx    j
25            j    j + 1
26
27        }
28    }
```

### 3. PencartanNfalERsztmpadaArrayBerttpeDataTerstruRtur

Pada kasus yang lebih kompleks pencarian ekstrim dapat juga dilakukan, misalnya mencari data mahasiswa dengan nilai terbesar, mencari lagu dengan durasi terlama, mencari pembalap yang memiliki catatan waktu balap tercepat, dan sebagainya. Sebagai contoh misalnya terdapat array yang digunakan untuk menyimpan data mahasiswa, kemudian terdapat fungsi IPK yang digunakan untuk mencari data mahasiswa dengan IPK tertinggi.

```
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk float64

type arrMhs [2023]mahasiswa


func IPK_1(T arrMhs, n int) float64 {
/* mengembalikan Ipk terkecil yang dImIlzkI mahaszsra pada array 7 yang berlsi
n mahaszs a */
    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
}
```

Apabila diperhatikan potongan program di atas, maka kita akan memperoleh ipk tertinggi, tetapi kitatidak memperoleh identitas mahasiswa dengan ipk tertinggi tersebut. Maka seperti penjelasan yang sudah diberikan sebelumnya, maka pencarian yang dilakukan bisa mengembalikan indeks mahasiswa dengan ipk tertinggi tersebut. Berikut ini adalah modifikasinya!



```
type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk float64

type arrMhs [2023]mahasiswa


func IPK_2(T arrl4hs, n int) int (
/+ inengenba11kan 1ndeks nahas1swa yang nen111k1 1pk tert1ngg1 pada array T yang
berlsl n inahas1 swa +/
    var idx int = 0
    var j :fnt = 1
    fo r j « n (
        1f T[ 1dx] .1pk < T[ j ] . ipk (
            idx     j
        }
        j    j + 1
    }
    return idx
}
```

Sehingga melalui algoritma di atas, identitas mahasiswa dapat diperoleh, misalnya T[ idx] .nama,

`T[idx].nim, T[idx].kelas, hingga T[idx].jurusan.`

## II. GUIDED

1. **Tipe Data**

**Sourcecode**

```
package main

import "fmt"

// Mendeklarasikan tipe data array arrInt dengan panjang
2023
type arrInt [2023]int

// Fungsi untuk mencari elemen terkecil dalam array
func terkecil(tabInt arrInt, n int) int {
    var min int = tabInt[0]
    var j int = 1
    for j < n {
        if min > tabInt[j] {
            min = tabInt[j]
        }
        j = j + 1
    }
    return min
}

// Fungsi main
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 nilai
terkecil
    minVal := terkecil(tab, n)

    // Menampilkan nilai terkecil
    fmt.Println("Nilai terkecil dalam array adalah:",
minVal)
}
```

**Screenshoot Output**

```
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11> go run
Masukkan jumlah elemen (maks 2023): 4
Masukkan elemen-elemen array:
Elemen ke-1: 5
Elemen ke-2: 7
Elemen ke-3: 8
Elemen ke-4: 9
Nilai terkecil dalam array adalah: 5
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11>
```

**Deskripsi Program**

Program ini mencari nilai terkecil dari sebuah array bilangan bulat. Program ini memulai dengan mendefinisikan tipe data array arrInt dengan kapasitas maksimal 2023. Kemudian, program meminta pengguna untuk memasukkan jumlah elemen array dan nilai-nilai elemen tersebut. Setelah itu, program memanggil fungsi terkecil untuk mencari nilai terkecil dalam array. Fungsi terkecil akan membandingkan setiap elemen dalam array dengan nilai minimum yang sudah ditemukan sejauh ini, dan jika ditemukan nilai yang lebih kecil, maka nilai minimum akan diupdate. Akhirnya, program akan menampilkan nilai terkecil yang ditemukan.

2. **Tipe Struktur IPK**

**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
}
```

```go
// 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\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11> go run
Masukkan jumlah mahasiswa (maks 2023): 2

Masukkan data mahasiswa ke-1
Nama: deshan
NIM: 326
Kelas: 06
Jurusan: IF
IPK: 4

Masukkan data mahasiswa ke-2
Nama: rafif
NIM: 326
Kelas: 06
Jurusan: IF
IPK: 3.9

Mahasiswa dengan IPK tertinggi:
Nama: deshan
NIM: 326
Kelas: 06
Jurusan: IF
IPK: 4.00
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11>
```

## Deskripsi Program

Program ini untuk mencari mahasiswa dengan IPK tertinggi dari suatu kumpulan data mahasiswa. Program ini memulai dengan mendefinisikan struktur data mahasiswa untuk menyimpan informasi seperti nama, NIM, kelas, jurusan, dan IPK. Kemudian, program membuat sebuah array mahasiswa dengan kapasitas maksimal 2023. Selanjutnya, program meminta pengguna untuk memasukkan jumlah mahasiswa dan data masing-masing mahasiswa. Setelah data terisi, program akan mencari indeks mahasiswa dengan IPK tertinggi menggunakan fungsi indeksIPKTertinggi dan kemudian menampilkan informasi lengkap dari mahasiswa tersebut.

# III. UNGUIDED

## 1. Soal Studi Case

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

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 djual.

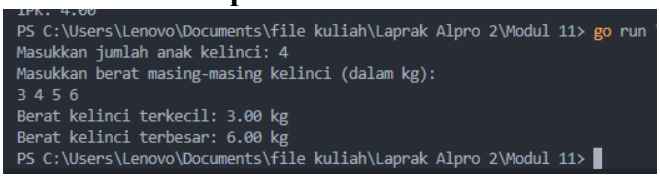
Keluaran terdiri dari dua buah bilangan riil yang menyatakan berat kelinci terkecil dan terbesar.

**Sourcecode**

```go
package main

import "fmt"

const maxKelinci = 1000

func main() {
    var n int
    var berat [maxKelinci]float64
    var terkecil, terbesar float64

    // Meminta input jumlah kelinci
    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&n)

    // Validasi input
    if n <= 0 || n > maxKelinci {
        fmt.Println("Jumlah kelinci harus antara 1 dan",
maxKelinci)
        return
    }

    // Memasukkan data berat kelinci
    fmt.Println("Masukkan berat masing-masing kelinci (dalam
kg):")
    for i := 0; i < n; i++ {
        fmt.Scan(&berat[i])
    }
```

```
    // Inisialisasi nilai terkecil dan terbesar dengan berat
kelinci pertama
    terkecil = berat[0]
    terbesar = berat[0]

    // Mencari nilai terkecil dan terbesar
    for i := 1; i < n; i++ {
        if berat[i] < terkecil {
            terkecil = berat[i]
        }
        if berat[i] > terbesar {
            terbesar = berat[i]
        }
    }

    // Menampilkan hasil
    fmt.Printf("Berat kelinci terkecil: %.2f kg\n",
terkecil)
    fmt.Printf("Berat kelinci terbesar: %.2f kg\n",
terbesar)
}
```

## Screenshoot Output




## Deskripsi Program

Program ini mencari berat kelinci terkecil dan terbesar dari sekumpulan data berat kelinci. Program ini meminta pengguna untuk memasukkan jumlah kelinci dan berat masing-masing kelinci. Selanjutnya, program akan mencari nilai terkecil dan terbesar dari data berat kelinci yang telah dimasukkan, kemudian menampilkan hasil pencarian tersebut.

## 2. Soal Studi Case

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

**Masukan** terdiri dari dua baris, yang mana baris pertama terdiri dari dua bilangan bulat x dan y. Bilangan x menyatakan banyaknya ikan yang akan djual, sedangkan y adalah banyaknya ikanyang akan dimasukan ke dalam wadah. Baris kedua terdiri dari sejumlah x bilangan riil yang menyatakan banyaknya ikan yang akan djual.

Keluaran terdiri dari dua baris. Baris pertama adalah kumpulan bilangan riil yang menyatakantotal berat ikan di setiap wadah §umIah 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.

**Sourcecode**

```go
package main

import (
    "fmt"
)

const maxIkan = 1000

func main() {
    var x, y int
    var beratIkan [maxIkan]float64
    var totalBeratWadah []float64

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

    // Validasi input
    if x <= 0 || y <= 0 || x > maxIkan {
        fmt.Println("Input tidak valid.")
        return
    }

    // Memasukkan data berat ikan
```

```
    fmt.Println("Masukkan berat masing-masing ikan (dalam
kg):")
    for i := 0; i < x; i++ {
        fmt.Scan(&beratIkan[i])
    }

    // Menghitung total berat ikan di setiap wadah
    var totalBerat float64
    for i := 0; i < x; i++ {
        totalBerat += beratIkan[i]
        if (i+1)%y == 0 || i == x-1 {
            totalBeratWadah = append(totalBeratWadah,
totalBerat)
            totalBerat = 0
        }
    }

    // Menampilkan total berat ikan di setiap wadah
    fmt.Println("Total berat ikan di setiap wadah:")
    for _, berat := range totalBeratWadah {
        fmt.Printf("%.2f kg ", berat)
    }
    fmt.Println()

    // Menghitung berat rata-rata ikan di setiap wadah
    var totalSemuaWadah float64
    for _, berat := range totalBeratWadah {
        totalSemuaWadah += berat
    }
    rataRata := totalSemuaWadah /
float64(len(totalBeratWadah))
    fmt.Printf("Berat rata-rata ikan di setiap wadah: %.2f
kg\n", rataRata)
}
```

## Screenshoot Output

```
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11> go run "
Masukkan jumlah ikan: 3
Masukkan kapasitas wadah: 2
Masukkan berat masing-masing ikan (dalam kg):
1 2 1
Total berat ikan di setiap wadah:
3.00 kg 1.00 kg
Berat rata-rata ikan di setiap wadah: 2.00 kg
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Modul 11>
```

**Deskripsi Program**

Program ini menghitung total berat ikan dalam setiap wadah dan berat rata-rata ikan per wadah. Program ini menerima input berupa jumlah ikan, kapasitas maksimal setiap wadah, dan berat masing-masing ikan. Kemudian, program akan membagi ikan ke dalam wadah sesuai dengan kapasitas yang ditentukan, menghitung total berat ikan di setiap wadah, dan akhirnya menghitung rata-rata berat ikan per wadah.

## 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 dengan spesifikasi subprogram sebagai berikut:

```
type arrBalita [700)float64

func hitungMinMax(arrBerat arrBalita; bMin, bMax *float64) {
/* Z.S.  Terdefznîsl  array  d1nan1s  arrBerat
   Proses :  iengh1tung  be ra t  n1n1inuln  dan  naks1nun  da1ain  a rray
   F.S. Nenampll/ran öerat  mynt/rum  dan  naks1nun  ball ta  */


function rerata (arrBerat arrBalita) real {
/+ nengh1tung  dan  inengeinba11kan  rerata  berat  ball ta  da1an  a rray  +/
```

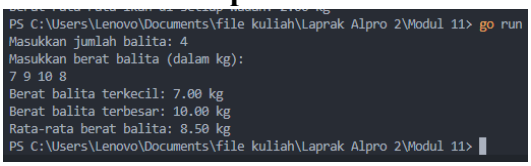
**Sourcecode**

```go
package main

import "fmt"

const maxBalita = 100

// Tipe data array untuk menyimpan berat balita
type arrBalita [maxBalita]float64

// Fungsi untuk menghitung berat minimum dan maksimum
func hitungMinMax(arrBerat arrBalita, n int) (float64,
float64) {
    var min, max float64
```

```
    // Inisialisasi nilai awal min dan max dengan elemen
pertama
    min = arrBerat[0]
    max = arrBerat[0]

    // Iterasi untuk mencari nilai min dan max
    for i := 1; i < n; i++ {
        if arrBerat[i] < min {
            min = arrBerat[i]
        }
        if arrBerat[i] > max {
            max = arrBerat[i]
        }
    }

    return min, max
}

// Fungsi untuk menghitung rata-rata berat
func rerata(arrBerat arrBalita, n int) float64 {
    var total float64

    // Hitung total berat semua balita
    for i := 0; i < n; i++ {
        total += arrBerat[i]
    }

    // Hitung rata-rata
    return total / float64(n)
}

func main() {
    var n int
    var beratBalita arrBalita

    // Input jumlah balita
    fmt.Print("Masukkan jumlah balita: ")
    fmt.Scan(&n)

    // Validasi input
    if n <= 0 || n > maxBalita {
        fmt.Println("Jumlah balita harus antara 1 dan",
maxBalita)
        return
```

```
    }

    // Input berat balita
    fmt.Println("Masukkan berat balita (dalam kg):")
    for i := 0; i < n; i++ {
        fmt.Scan(&beratBalita[i])
    }

    // Panggil fungsi untuk menghitung min, max, dan rata-
rata
    min, max := hitungMinMax(beratBalita, n)
    rata := rerata(beratBalita, n)

    // Tampilkan hasil
    fmt.Printf("Berat balita terkecil: %.2f kg\n", min)
    fmt.Printf("Berat balita terbesar: %.2f kg\n", max)
    fmt.Printf("Rata-rata berat balita: %.2f kg\n", rata)
}
```

## Screenshoot Output




## Deskripsi Program

Program ini untuk menghitung statistik berat badan balita di sebuah posyandu. Program ini akan meminta pengguna memasukkan jumlah balita dan berat badan masing-masing balita dalam satuan kilogram. Setelah data dimasukkan, program akan menghitung dan menampilkan berat badan balita terkecil, terbesar, dan rata-rata dari seluruh data yang telah dimasukkan.