LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL XI

PENCARIAN NILAI EKSTRIMPADA HIMPUNAN DATA

Disusun Oleh :

TRI MARSELINUS S/ 2311102209

IF 11-06

Dosen Pengampu :

ABEDNEGO DWI SEPTIADI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

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.

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.

II. GUIDED

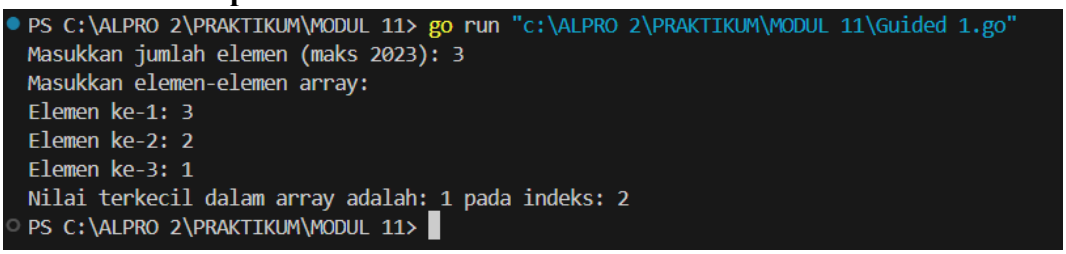
1. Soal Studi Case

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)
}
```

Screenshoot Output



Deskripsi Program

Program diatas akan meminta user untuk mencari nilai terkecil dalam sebuah array bilangan bulat dan mengembalikan indeks dari nilai terkecil tersebut. Program ini menggunakan tipe data baru yang disebut arrInt, yang merupakan array dengan panjang maksimum 2023. Dalam fungsi main, pengguna diminta untuk memasukkan jumlah elemen yang ingin dimasukkan ke dalam array, diikuti dengan elemen-elemen itu sendiri. Setelah input selesai, program memanggil fungsi terkecil, yang bertugas untuk menemukan indeks dari elemen terkecil dalam array dengan membandingkan setiap elemen dengan elemen yang ada di indeks terkecil saat ini. Setelah fungsi terkecil selesai menjalankan logikanya, program kemudian menampilkan output yang terdiri dari nilai terkecil yang ditemukan dan indeksnya dalam array.

2. Soal Studi Case

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
}

// Definisi tipe data array mahasiswa dengan kapasitas
maksimal 2023
type arrMhs [2023]mahasiswa

// Fungsi untuk mencari IPK tertinggi dalam array
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
}

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

    // Mencari dan menampilkan IPK tertinggi
    tertinggi := ipk(dataMhs, n)
    fmt.Printf("\nIPK tertinggi dari %d mahasiswa
adalah: %.2f\n", n, tertinggi)


}
```

Screenshoot Output

```
PS C:\ALPRO 2\PRAKTIKUM\MODUL 11> go run "c:\ALPRO 2\PRAKTIKUM\MODUL 11\Guided 2.go"
Masukkan jumlah mahasiswa (maks 2023): 2

Masukkan data mahasiswa ke-1
Nama: Marsel
NIM: 209
Kelas: 06
Jurusan: Informatika
IPK: 3.8

Masukkan data mahasiswa ke-2
Nama: Djoko
NIM: 212
Kelas: 05
Jurusan: Informatika
IPK: 3.9

IPK tertinggi dari 2 mahasiswa adalah: 3.90
PS C:\ALPRO 2\PRAKTIKUM\MODUL 11>
```

Deskripsi Program

Program diatas digunakan untuk mengumpulkan data mahasiswa dan mencari Indeks Prestasi Kumulatif (IPK) tertinggi di antara mereka. Program ini mendefinisikan sebuah struct bernama mahasiswa yang berisi atribut seperti nama, NIM, kelas, jurusan, dan IPK. Selain itu, program menggunakan tipe data array arrMhs untuk menampung data mahasiswa dengan kapasitas maksimum 2023. Dalam fungsi main, pengguna diminta untuk memasukkan jumlah mahasiswa yang ingin diinput, diikuti dengan data masing-masing mahasiswa. Setelah semua data dimasukkan, program memanggil fungsi ipk untuk menghitung dan menemukan IPK tertinggi dari array yang telah diisi.

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 riil berikutnya adalah berat dari anak kelinci yang akan dijual.

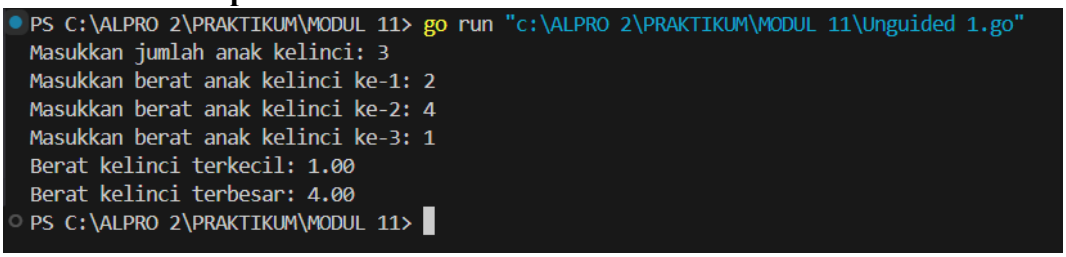
Keluaran terdiri dari dua buah bilangan riil yang menyatakan berat kelinci terkecil dan terbesar.

Sourcecode

```
package main

import "fmt"

func main() {
    var N int

    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&N)

    if N < 1 || N > 1000 {
        fmt.Println("Jumlah kelinci tidak valid.")
        return
    }

    beratKelinci := make([]float64, N)

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

    beratTerkecil := beratKelinci[0]
    beratTerbesar := beratKelinci[0]

    for _, berat := range beratKelinci {
        if berat < beratTerkecil {
            beratTerkecil = berat
        }
        if berat > beratTerbesar {
            beratTerbesar = berat
        }
    }
```

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

Screenshoot Output



Deskripsi Program

Program diatas adalah sebuah program untuk mengumpulkan dan menganalisis berat anak kelinci. Pengguna diminta untuk memasukkan jumlah kelinci yang ingin diinput, dengan batasan antara 1 hingga 1000. Setelah jumlah kelinci divalidasi, program kemudian meminta pengguna untuk memasukkan berat setiap kelinci satu per satu dan menyimpannya dalam sebuah slice bertipe **float64**. Setelah semua data berat dimasukkan, program melakukan iterasi untuk menemukan berat terkecil dan terbesar dengan membandingkan setiap nilai berat yang ada dalam slice. Akhirnya, program menampilkan hasil berupa berat kelinci terkecil dan terbesar.

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 dimasukan ke dalam wadah. Baris kedua terdiri dari sejumlah x bilangan riil yang menyatakan banyaknya ikan yang akan dijual. Informatics lab

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.

Sourcecode

```
package main

import "fmt"

func main() {
    var x, y int

    fmt.Print("Masukkan jumlah ikan yang akan dijual dan
banyaknya ikan dalam wadah (x y): ")
    fmt.Scan(&x, &y)

    if x < 1 || x > 1000 || y < 1 {
        fmt.Println("Jumlah ikan harus antara 1 dan
1000, dan banyaknya ikan dalam wadah harus lebih dari
0.")
        return
    }

    beratIkan := make([]float64, x)

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

    var totalBerat []float64
    var totalIkanDiWadah []int

    for i := 0; i < x; i++ {
        if i%y == 0 {
```

```
            totalBerat = append(totalBerat, 0)
            totalIkanDiWadah = append(totalIkanDiWadah,
0)
        }
        totalBerat[len(totalBerat)-1] += beratIkan[i]
        totalIkanDiWadah[len(totalIkanDiWadah)-1]++
    }

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

    fmt.Println("Berat rata-rata ikan di setiap wadah:")
    for i := 0; i < len(totalBerat); i++ {
        if totalIkanDiWadah[i] > 0 {
            rataRata := totalBerat[i] /
float64(totalIkanDiWadah[i])
            fmt.Printf("%.2f ", rataRata)
        } else {
            fmt.Printf("0 ")
        }
    }
    fmt.Println()
}
```

Screenshoot Output

```
PS C:\ALPRO 2\PRAKTIKUM\MODUL 11> go run "c:\ALPRO 2\PRAKTIKUM\MODUL 11\Unguided 2.go"
Masukkan jumlah ikan yang akan dijual dan banyaknya ikan dalam wadah (x y): 3 3
Masukkan berat ikan:
Berat ikan ke-1: 2
Berat ikan ke-2: 4
Berat ikan ke-3: 1
Total berat ikan di setiap wadah:
7.00
Berat rata-rata ikan di setiap wadah:
2.33
PS C:\ALPRO 2\PRAKTIKUM\MODUL 11>
```

Deskripsi Program

Program diatas adalah program yang menghitung total berat dan rata-rata berat ikan yang dijual berdasarkan jumlah ikan dan kapasitas wadah. Pengguna diminta untuk memasukkan dua nilai: jumlah ikan yang akan dijual (x) dan banyaknya ikan dalam setiap wadah (y). Program kemudian memvalidasi input untuk memastikan bahwa jumlah ikan berada dalam rentang yang benar dan bahwa jumlah ikan dalam wadah lebih dari nol. Setelah itu, pengguna diminta untuk memasukkan berat setiap ikan, yang disimpan dalam sebuah slice bertipe **float64**.

Setelah semua data berat ikan dimasukkan, program menghitung total berat ikan untuk setiap wadah dengan menggunakan loop, di mana setiap wadah akan menampung ikan sesuai dengan kapasitas yang ditentukan. Total berat dan jumlah ikan dalam setiap wadah disimpan dalam slice terpisah. Program kemudian menampilkan total berat ikan di setiap wadah dan menghitung rata-rata berat ikan per wadah dengan membagi total berat dengan jumlah ikan di wadah tersebut. Hasil akhir berupa total berat dan rata-rata berat ikan untuk setiap wadah ditampilkan

3. Soal Studi Case

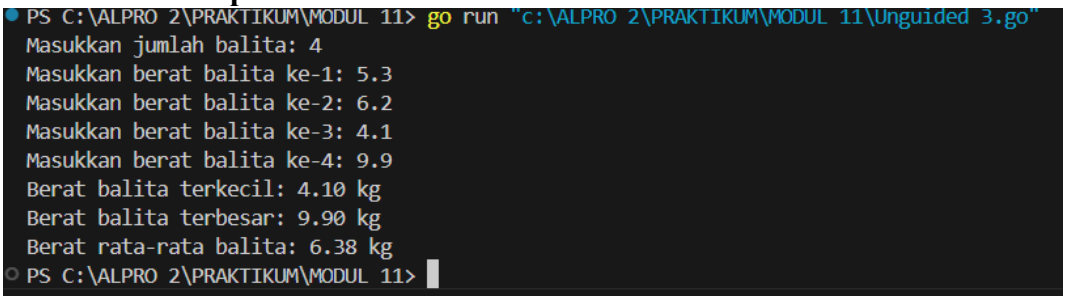
Buatlah sebuah program Pos Pelayanan Terpadu (posyandu) sebagai tempat pelayanan kesehatan perlu mencatat data berat balita (dalam kg). Petugas akan memasukkan data tersebut ke dalam array. Dari data yang diperoleh akan dicari berat balita terkecil, terbesar, dan reratanya.

Sourcecode

```
package main

import  "fmt"

type arrBalita [100]float64

func hitungMinMax(arrBerat arrBalita, n int, bMin, bMax
*float64) {
    if n == 0 {
        return
    }
    *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 {
    if n == 0 {
        return 0.0
    }
    var total float64
    for i := 0; i < n; i++ {
        total += arrBerat[i]
```

```
    }
    return total / float64(n)
}

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

    if n > 100 {
        fmt.Println("Jumlah balita tidak boleh lebih
dari 100.")
        return
    }

    var beratBalita arrBalita
    for i := 0; i < n; i++ {
        fmt.Print("Masukkan berat balita ke-",i+1,": ")
        fmt.Scan(&beratBalita[i])
    }



    var minBerat, maxBerat float64
    hitungMinMax(beratBalita, n, &minBerat, &maxBerat)
    rataBerat := rerata(beratBalita, n)

    fmt.Printf("Berat balita terkecil: %.2f kg\n",
minBerat)
    fmt.Printf("Berat balita terbesar: %.2f kg\n",
maxBerat)
    fmt.Printf("Berat rata-rata balita: %.2f kg\n",
rataBerat)
}
```

Screenshoot Output



Deskripsi Program

Program di atas adalah program yang meminta pengguna untuk mengumpulkan dan menganalisis berat balita. Pengguna diminta untuk memasukkan jumlah balita yang ingin diinput (maksimal 100), dan setelah validasi, program meminta pengguna untuk memasukkan berat setiap balita satu per satu. Berat balita disimpan dalam sebuah array bertipe **arrBalita**.

Setelah semua data berat dimasukkan, program memanggil fungsi **hitungMinMax** untuk menentukan berat terkecil dan terbesar dengan menggunakan pointer untuk memperbarui nilai minimum dan maksimum. Selain itu, fungsi **rerata** digunakan untuk menghitung berat rata-rata balita dengan menjumlahkan semua berat dan membaginya dengan jumlah balita yang ada. Hasil akhir berupa berat balita terkecil, terbesar, dan rata-rata ditampilkan