LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 12 & 13

PENGURUTAN DATA

Disusun Oleh :

Annasya Maulafidatu Zahra / 2311102246

S1-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

Pengurutan data adalah cara untuk menata elemen dalam urutan tertentu, baik dari yang terkecil ke yang terbesar maupun sebaliknya. Dua algoritma dasar yang umum digunakan untuk pengurutan adalah Selection Sort dan Insertion Sort.

Selection Sort berfungsi dengan mencari elemen terkecil dalam array dan menukarnya dengan elemen yang pertama. Kemudian, algoritma ini akan terus mencari elemen terkecil berikutnya dan menukarnya dengan elemen kedua, dan seterusnya hingga seluruh array teratur.

Sementara itu, Insertion Sort mirip dengan cara kita mengatur kartu. Algoritma ini menangani setiap elemen satu per satu, menyisipkannya pada posisi yang tepat di antara elemen-elemen yang sudah terurut. Dengan metode ini, elemen baru akan diletakkan di tempat yang sesuai tanpa harus memindahkan seluruh array.

Kedua algoritma ini dapat diterapkan dengan mudah dalam bahasa Go, dan pilihan antara keduanya bergantung pada kebutuhan spesifik dari data yang ingin diurutkan. Insertion Sort sering dipilih untuk dataset kecil atau yang hampir terurut, sedangkan Selection Sort lebih gampang untuk dipahami.

I. GUIDED

1. Guided 1

Soal Studi Case

Hercules, preman terkenal seantero ibukota, memiliki kerabat di banyak daerah. Tentunya Hercules sangat suka mengunjungi semua kerabatnya itu. Diberikan masukan nomor rumah dari semua kerabatnya di suatu daerah, buatlah program rumahkerabat yang akan menyusun nomor-nomor rumah kerabatnya secara terurut membesar menggunakan algoritma selection sort.

Masukan dimulai dengan sebuah integer n (0 < n < 1000), banyaknya daerah kerabat Hercules tinggal. Isi n baris berikutnya selalu dimulai dengan sebuah integer m (0 < m < 1000000) yang menyatakan banyaknya rumah kerabat di daerah tersebut, diikuti dengan rangkaian bilangan bulat positif, nomor rumah para kerabat.

Keluaran terdiri dari n baris, yaitu rangkaian rumah kerabatnya terurut membesar di masingmasing daerah.

No	Masukan	Keluaran
1	3 5 2 1 7 9 13 6 189 15 27 39 75 133 3 4 9 1	1 2 7 9 13 15 27 39 75 133 189 1 4 9

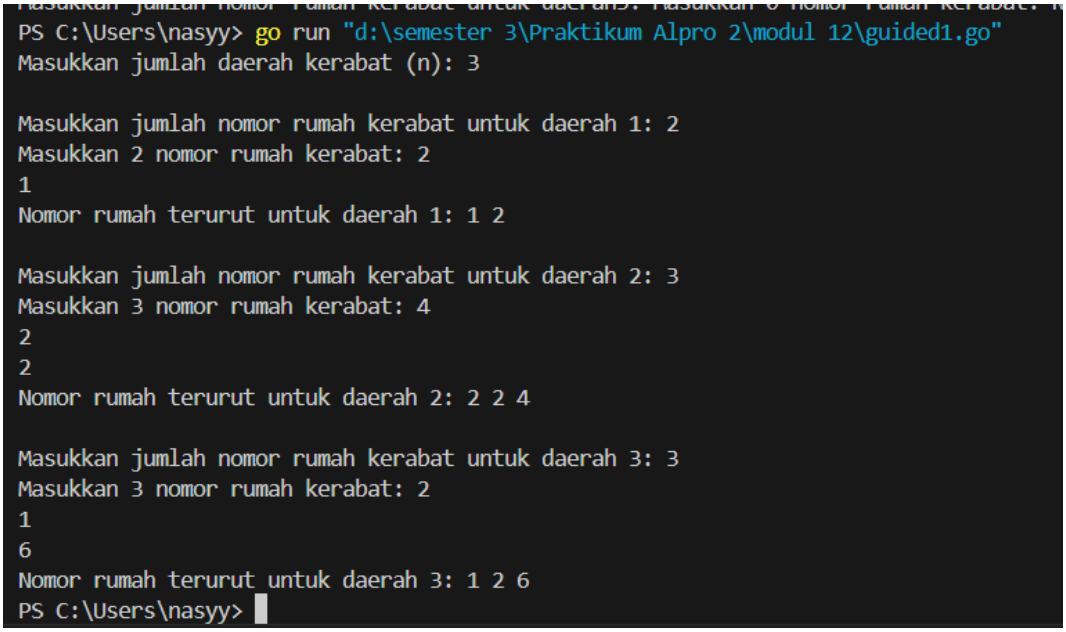
Keterangan: Terdapat 3 daerah dalam contoh input, dan di masing-masing daerah mempunyai 5, 6, dan 3 kerabat.

Sourcecode

```go
package main

import (
    "fmt"
)

func selectionSort(arr []int, n int) {
    for i := 0; i < n-1; i++ {
        idxMin := i
        for j := i + 1; j < n; j++ {
            if arr[j] < arr[idxMin] {
                idxMin = j
            }
        }
```

```
        arr[i], arr[idxMin] = arr[idxMin], arr[i]
    }
}

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

    for daerah := 1; daerah <= n; daerah++ {
        var m int
        fmt.Printf("\nMasukkan jumlah nomor rumah kerabat
untuk daerah %d: ", daerah)
        fmt.Scan(&m)

        arr := make([]int, m)
        fmt.Printf("Masukkan %d nomor rumah kerabat: ", m)
        for i := 0; i < m; i++ {
            fmt.Scan(&arr[i])
        }

        selectionSort(arr, m)

        fmt.Printf("Nomor rumah terurut untuk daerah %d: ",
daerah)
        for _, num := range arr {
            fmt.Printf("%d ", num)
        }
        fmt.Println()
    }

}
```

Screenshoot Output



Deskripsi Program

Program ini mengurutkan nomor rumah kerabat di beberapa daerah. Program ini akan meminta pengguna untuk memasukkan jumlah daerah dan jumlah nomor rumah di setiap daerah. Kemudian, pengguna akan diminta memasukkan nomor-nomor rumah tersebut. Selanjutnya, program akan menggunakan algoritma pengurutan selection sort untuk mengurutkan nomor rumah di setiap daerah secara ascending (dari yang terkecil ke terbesar). Outputnya berupa daftar nomor rumah yang sudah terurut untuk setiap daerah.

2. Guided 2

Soal Studi Case

Buatlah sebuah program yang digunakan untuk membaca data integer seperti contoh yang diberikan di bawah ini, kemudian diurutkan (menggunakan metoda *insertion sort*), dan memeriksa apakah data yang terurut berjarak sama terhadap data sebelumnya.

Masukan terdiri dari sekumpulan bilangan bulat yang diakhiri oleh bilangan negatif. Hanya bilangan non negatif saja yang disimpan ke dalam array.

Keluaran terdiri dari dua baris. Baris pertama adalah isi dari array setelah dilakukan pengurutan, sedangkan baris kedua adalah status jarak setiap bilangan yang ada di dalam array. "Data berjarak x" atau "data berjarak tidak tetap".

Contoh masukan dan keluaran

No	Masukan	Keluaran
1	31 13 25 43 1 7 19 37 -5	1 7 13 19 25 31 37 43 Data berjarak 6
2	4 40 14 8 26 1 38 2 32 -31	1 2 4 8 14 26 32 38 40 Data berjarak tidak tetap

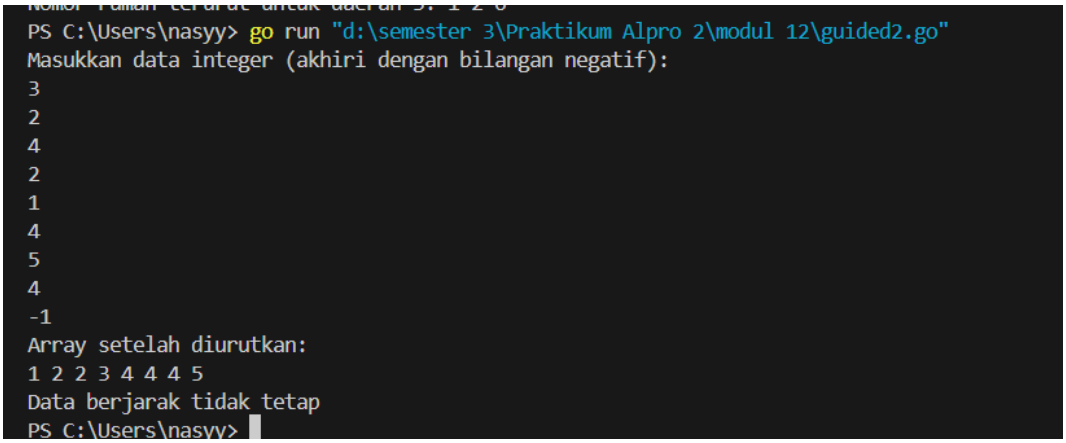
Sourcecode

```go
package main

import (
    "fmt"
)

// Fungsi untuk mengurutkan array menggunakan Insertion Sort
func insertionSort(arr []int, n int) {
    for i := 1; i < n; i++ {
        key := arr[i]
        j := i - 1

        // Geser elemen yang lebih besar dari key ke kanan
        for j >= 0 && arr[j] > key {
            arr[j+1] = arr[j]
            j--
        }
        arr[j+1] = key
    }
}

// Fungsi untuk memeriksa apakah selisih elemen array tetap
func isConstantDifference(arr []int, n int) (bool, int) {
    if n < 2 {
        return true, 0
```

```
    }

    difference := arr[1] - arr[0]
    for i := 1; i < n-1; i++ {
        if arr[i+1]-arr[i] != difference {
            return false, 0
        }
    }
    return true, difference
}

func main() {
    var arr []int
    var num int

    // Input data hingga bilangan negatif ditemukan
    fmt.Println("Masukkan data integer (akhiri dengan
bilangan negatif):")
    for {
        fmt.Scan(&num)
        if num < 0 {
            break
        }
        arr = append(arr, num)
    }

    n := len(arr)

    // Urutkan array menggunakan Insertion Sort
    insertionSort(arr, n)

    // Periksa apakah selisih elemen tetap
    isConstant, difference := isConstantDifference(arr, n)

    // Tampilkan hasil pengurutan
    fmt.Println("Array setelah diurutkan:")
    for _, val := range arr {
        fmt.Printf("%d ", val)
    }
    fmt.Println()

    // Tampilkan status jarak
    if isConstant {
        fmt.Printf("Data berjarak %d\n", difference)
    } else {
```

```
        fmt.Println("Data berjarak tidak tetap")
    }
}
```

Screenshot Program



Deskripsi Program

Program ini memeriksa apakah suatu deret bilangan memiliki selisih yang konstan setelah diurutkan. Pertama, program akan meminta pengguna untuk memasukkan deret bilangan hingga pengguna memasukkan bilangan negatif. Kemudian, deret bilangan tersebut akan diurutkan menggunakan algoritma Insertion Sort. Setelah diurutkan, program akan memeriksa apakah selisih antara setiap bilangan yang berdekatan selalu sama. Jika selisihnya sama, maka program akan menampilkan nilai selisih tersebut. Jika tidak, maka program akan menampilkan pesan bahwa selisihnya tidak tetap.

II. UNGUIDED

1. Unguided 1

Soal Studi Case

Belakangan diketahui ternyata Hercules itu tidak berani menyeberang jalan, maka selalu diusahakan agar hanya menyeberang jalan sesedikit mungkin, hanya diujung jalan. Karena nomor rumah sisi kiri jalan selalu ganjil dan sisi kanan jalan selalu genap, maka buatlah program kerabat dekat yang akan menampilkan nomor rumah mulai dari nomor yang ganjil lebih dulu

terurut membesar dan kemudian menampilkan nomor rumah dengan nomor genap terurut mengecil.

Format **Masukan** masih persis sama seperti sebelumnya.
Keluaran terdiri dari n baris, yaitu rangkaian rumah kerabatnya terurut membesar untuk nomor ganjil, diikuti dengan terurut mengecil untuk nomor genap, di masing-masing daerah.

Keterangan: Terdapat 3 daerah dalam contoh masukan. Baris kedua berisi campuran bilangan ganjil dan genap. Baris berikutnya hanya berisi bilangan ganjil, dan baris terakhir hanya berisi bilangan genap.

Petunjuk:
Waktu pembacaan data, bilangan ganjil dan genap dipisahkan ke dalam dua array yang berbeda, untuk kemudian masing-masing diurutkan tersendiri. Atau, tetap disimpan dalam satu array, diurutkan secara keseluruhan. Tetapi pada waktu pencetakan, mulai dengan mencetak semua nilai ganjil lebih dulu, kemudian setelah selesai cetaklah semua nilai genapnya.

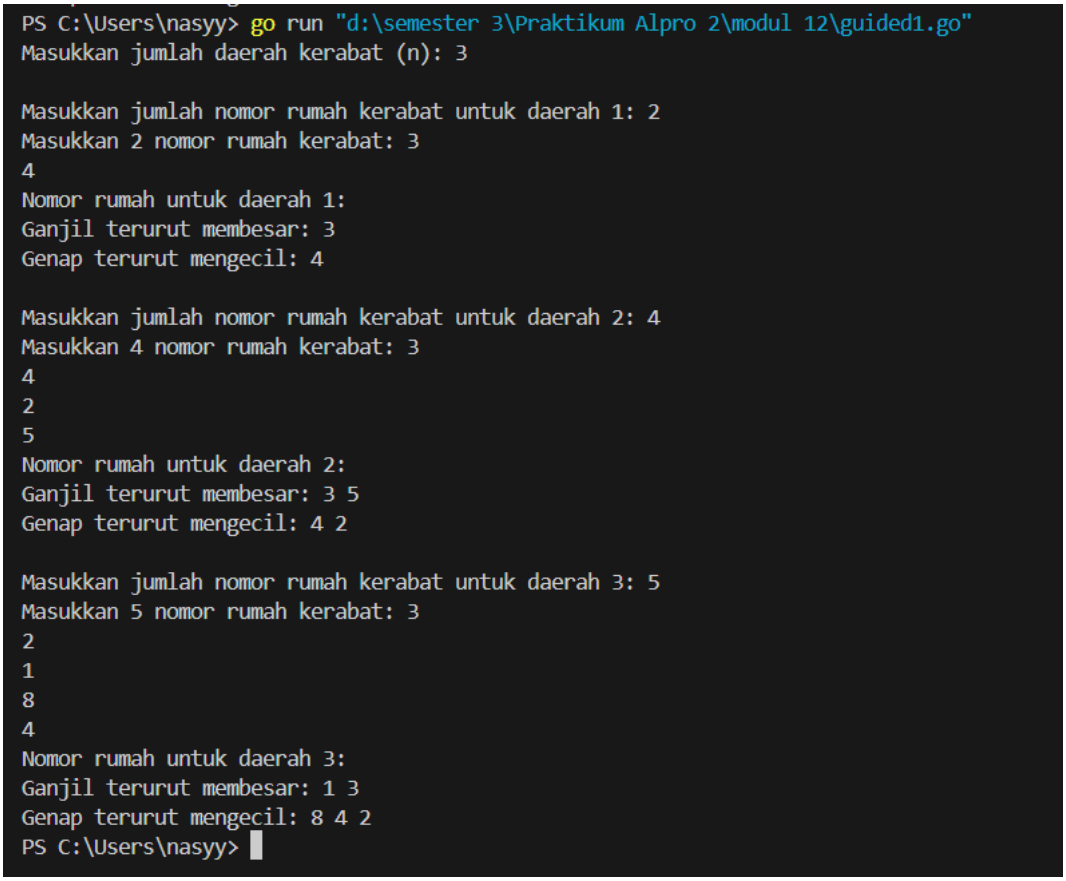
Sourcecode

```
package main

import (
    "fmt"
)

func selectionSort(arr []int, ascending bool) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        idx := i
        for j := i + 1; j < n; j++ {
            if ascending {
                if arr[j] < arr[idx] {
                    idx = j
                }
            } else {
                if arr[j] > arr[idx] {
                    idx = j
                }
            }
        }
        arr[i], arr[idx] = arr[idx], arr[i]
```

```
    }
}

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

    for daerah := 1; daerah <= n; daerah++ {
        var m int
        fmt.Printf("\nMasukkan jumlah nomor rumah kerabat
untuk daerah %d: ", daerah)
        fmt.Scan(&m)

        arr := make([]int, m)
        fmt.Printf("Masukkan %d nomor rumah kerabat: ", m)
        for i := 0; i < m; i++ {
            fmt.Scan(&arr[i])
        }


        var ganjil, genap []int
        for _, num := range arr {
            if num%2 == 0 {
                genap = append(genap, num)
            } else {
                ganjil = append(ganjil, num)
            }
        }


        selectionSort(ganjil, true)

        selectionSort(genap, false)


        fmt.Printf("Nomor rumah untuk daerah %d:\n", daerah)
        fmt.Print("Ganjil terurut membesar: ")
        for _, num := range ganjil {
            fmt.Printf("%d ", num)
        }
        fmt.Println()
        fmt.Print("Genap terurut mengecil: ")
        for _, num := range genap {
            fmt.Printf("%d ", num)
```

```
        }
        fmt.Println()
    }
}
```

Screenshoot Output



Deskripsi Program

Program ini mengurutkan nomor rumah kerabat berdasarkan ganjil dan genap. Pertama, program meminta pengguna untuk memasukkan jumlah daerah dan jumlah rumah di setiap daerah. Kemudian, pengguna memasukkan nomor-nomor rumah tersebut. Program selanjutnya akan memisahkan nomor rumah menjadi dua kelompok: nomor ganjil dan nomor genap. Masing-masing kelompok kemudian diurutkan menggunakan algoritma Selection Sort. Nomor ganjil diurutkan dari yang terkecil ke terbesar, sedangkan nomor genap diurutkan dari yang terbesar ke terkecil. Hasil akhir berupa daftar nomor rumah yang sudah terurut berdasarkan jenisnya (ganjil atau genap) untuk setiap daerah.

2. Unguided 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.

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.

No	Masukan	Keluaran
1	7 23 11 0 5 19 2 29 3 13 17 0 -5313	11 12

Keterangan:

Sampai bilangan O yang pertama, data terbaca adalah 7 23 11, setelah tersusun: 7 11 23, maka median saat itu adalah 11.

Sampai bilangan O yang kedua, data adalah 7 23 11 5 19 2 29 3 13 17, setelah tersusun diperoleh: 2 3 5 7 11 13 17 19 23 29. Karena ada 10 data, genap, maka median adalah (11+13)/2=12.

Petunjuk:

Untuk setiap data bukan 0 (dan bukan marker -5313541) simpan ke dalam array, Dan setiap kali menemukan bilangan O, urutkanlah data yang sudah tersimpan dengan menggunakan metode insertion sort dan ambil mediannya.

Sourcecode

```go
package main

import (
    "fmt"
    "math"
)
```

```go
func sort(arr []int) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        minIdx := i
        for j := i + 1; j < n; j++ {
            if arr[j] < arr[minIdx] {
                minIdx = j
            }
        }
        arr[i], arr[minIdx] = arr[minIdx], arr[i]
    }
}

func median(arr []int) int {
    n := len(arr)
    if n == 0 {
        return 0
    }
    if n%2 == 1 {
        return arr[n/2]
    }
    return int(math.Floor(float64(arr[n/2-1]+arr[n/2]) /
2.0))
}

func main() {
    var n int
    var nums []int

    fmt.Println("Masukkan angka (akhiri dengan 0 atau -5313
sebagai penanda akhir):")
    for {
        fmt.Scan(&n)
        if n == 0 || n == -5313 {
            if len(nums) > 0 {
                sort(nums)
                fmt.Println("Median:", median(nums))
            }
            if n == 0 {
                nums = []int{}
            } else {
                break
            }
        } else {
            nums = append(nums, n)
```

```
        }
    }
}
```

Screenshoot Output

```
PS C:\Users\nasyy> go run "d:\semester 3\Praktikum Alpro 2\modul 12\unguided2.go"
Masukkan angka (akhiri dengan 0 atau -5313 sebagai penanda akhir):
6 7 8 -5313
Median: 7
PS C:\Users\nasyy>
```

Deskripsi Program

Program ini menghitung median dari serangkaian angka yang diinputkan oleh pengguna. Pertama, program terus meminta pengguna untuk memasukkan angka hingga pengguna memasukkan 0 atau -5313 sebagai tanda akhir input. Kemudian, program mengurutkan angka-angka tersebut menggunakan algoritma Selection Sort. Setelah diurutkan, program menghitung median berdasarkan jumlah angka: jika jumlah angka ganjil, median adalah angka tengah, jika jumlah angka genap, median adalah rata-rata dari dua angka tengah. Hasil outputnya, program akan mencetak median dari angka-angka yang telah diinputkan.

3. Unguided 3

Soal Studi Case

Sebuah program perpustakaan digunakan untuk mengelola data buku di dalam suatu perpustakaan. Misalnya terdefinisi struct dan array seperti berikut ini:

```
const nMax : integer = 7919
type Buku = <
    id, judul, penulis, penerbit : string
    eksemplar, tahun, rating : integer >

type DaftarBuku = array [ 1..nMax] of Buku
Pustaka : DaftarBuku
nPustaka: integer
```

Masukan terdiri dari beberapa baris. Baris pertama adalah bilangan bulat N yang menyatakan banyaknya data buku yang ada di dalam perpustakaan. N baris berikutnya, masing-masingnya adalah data buku sesuai dengan atribut

atau field pada struct. Baris terakhir adalah bilangan bulat yang menyatakan rating buku yang akan dicari.

Keluaran terdiri dari beberapa baris. Baris pertama adalah data buku terfavorit, baris kedua adalah lima judul buku dengan rating tertinggi, selanjutnya baris terakhir adalah data buku yang dicari sesuai rating yang diberikan pada masukan baris terakhir

Lengkapi subprogram-subprogram dibawah ini, sesuai dengan I.S. dan F.S yang diberikan.

```
procedure DaftarkanBuku(in/out pustaka : DaftarBuku, n : integer)
{I.S. sejumlah n data buku telah siap para piranti masukan
 F.S. n berisi sebuah nilai, dan pustaka berisi sejumlah n data buku}

procedure CetakTerfavorit(in pustaka : DaftarBuku, in n : integer)
{I.S. array pustaka berisi n buah data buku dan belum terurut
 F.S. Tampilan data buku (judul, penulis, penerbit, tahun)
terfavorit, yaitu memiliki rating tertinggi}

procedure UrutBuku( in/out pustaka : DaftarBuku, n : integer )
{I.S. Array pustaka berisi n data buku
 F.S. Array pustaka terurut menurun/mengecil terhadap rating.
Catatan: Gunakan metoda Insertion sort}

procedure Cetak5Terbaru( in pustaka : DaftarBuku, n integer )
{I.S. pustaka berisi n data buku yang sudah terurut menurut rating
 F.S. Laporan 5 judul buku dengan rating tertinggi
 Catatan: Isi pustaka mungkin saja kurang dari 5}

procedure CariBuku(in pustaka : DaftarBuku, n : integer, r : integer )
{I.S. pustaka berisi n data buku yang sudah terurut menurut rating
 F.S. Laporan salah satu buku (judul, penulis, penerbit, tahun,
eksemplar, rating) dengan rating yang diberikan. Jika tidak ada buku
dengan rating yang ditanyakan, cukup tuliskan "Tidak ada buku dengan
rating seperti itu". Catatan: Gunakan pencarian biner/belah dua.}
```

Sourcecode

```go
package main

import (
    "fmt"
    "sort"
)

type Buku struct {
    ID        string
    Judul     string
    Penulis   string
    Penerbit  string
    Eksemplar int
    Tahun     int
```

```
    Rating    int
}

func UrutBuku(buku []Buku) {
    sort.Slice(buku, func(i, j int) bool {
        return buku[i].Rating > buku[j].Rating
    })
}

func CetakTerfavorit(buku Buku) {
    fmt.Println("Buku Terfavorit:")
    fmt.Printf("%+v\n", buku)
}

func Cetak5Terbaru(buku []Buku) {
    fmt.Println("5 Buku dengan Rating Tertinggi:")
    for i := 0; i < 5 && i < len(buku); i++ {
        fmt.Println(buku[i].Judul)
    }
}

func CariBuku(buku []Buku, ratingDicari int) int {
    left, right := 0, len(buku)-1
    for left <= right {
        mid := (left + right) / 2
        if buku[mid].Rating == ratingDicari {
            return mid
        } else if buku[mid].Rating < ratingDicari {
            right = mid - 1
        } else {
            left = mid + 1
        }
    }
    return -1
}

func main() {
    var n int
    var pustaka []Buku

    fmt.Print("Masukkan jumlah buku: ")
    fmt.Scan(&n)

    pustaka = make([]Buku, n)
    for i := 0; i < n; i++ {
```

```
        fmt.Printf("Masukkan data buku ke-%d (ID Judul
Penulis Penerbit Eksemplar Tahun Rating): ", i+1)
        fmt.Scan(&pustaka[i].ID, &pustaka[i].Judul,
&pustaka[i].Penulis, &pustaka[i].Penerbit,
&pustaka[i].Eksemplar, &pustaka[i].Tahun,
&pustaka[i].Rating)
    }

    fmt.Print("Masukkan rating buku yang ingin dicari: ")
    fmt.Scan(&n)

    UrutBuku(pustaka)

    CetakTerfavorit(pustaka[0])

    Cetak5Terbaru(pustaka)

    index := CariBuku(pustaka, n)
    if index != -1 {
        fmt.Printf("Buku dengan rating %d ditemukan: %+v\n",
n, pustaka[index])
    } else {
        fmt.Printf("Buku dengan rating %d tidak
ditemukan\n", n)
    }
}
```

Screenshoot Output

```
Masukkan jumlah buku: 2
Masukkan data buku ke-1 (ID Judul Penulis Penerbit Eksemplar Tahun Rating): 001 Matahari TereLiye Gramedia 10 2016 5
Masukkan data buku ke-2 (ID Judul Penulis Penerbit Eksemplar Tahun Rating): 002 Hujan Tereliye Gramedia 6 2017 4
Masukkan rating buku yang ingin dicari: 5
Buku Terfavorit:
{ID:001 Judul:Matahari Penulis:TereLiye Penerbit:Gramedia Eksemplar:10 Tahun:2016 Rating:5}
5 Buku dengan Rating Tertinggi:
Matahari
Hujan
Buku dengan rating 5 ditemukan: {ID:001 Judul:Matahari Penulis:TereLiye Penerbit:Gramedia Eksemplar:10 Tahun:2016 Rating:5}
PS C:\Users\nasyy>
```

Deskripsi Program

Program ini mengelola data buku dalam sebuah perpustakaan. Program ini memungkinkan pengguna untuk memasukkan data buku secara manual, mengurutkan buku berdasarkan rating, mencari buku berdasarkan rating tertentu, dan menampilkan informasi buku yang paling favorit serta 5 buku dengan rating tertinggi. Data buku yang disimpan meliputi ID, judul, penulis, penerbit, jumlah eksemplar, tahun terbit, dan rating. Program ini

menggunakan algoritma sorting dan binary search untuk efisiensi dalam mengolah data buku.