

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XII  
PENGURUTAN DATA**



**Disusun Oleh :**

**Rafi Bintang Maulana / 2311102327**

**Kelas IF-11-06**

**Dosen Pengampu :**

**ABEDNEGO DWI SEPTIADI**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## **I. DASAR TEORI**

Pengurutan data (sorting) adalah proses menyusun elemen-elemen dalam sebuah kumpulan data ke dalam urutan tertentu, baik itu secara menaik (ascending) maupun menurun (descending). Dalam dunia pemrograman komputer, pengurutan data merupakan salah satu operasi fundamental yang sering digunakan untuk mengoptimalkan pengolahan data, seperti pencarian data, analisis data, dan pengelompokan data.

Pengurutan membantu mempercepat proses pencarian dengan memanfaatkan algoritma pencarian yang lebih efisien. Terdapat berbagai algoritma pengurutan yang dirancang dengan tingkat efisiensi yang berbeda, seperti Bubble Sort, Insertion Sort, Merge Sort, Quick Sort, dan Heap Sort.

## II. GUIDED

### 1. Sourcecode

```
package main

import (
    "fmt"
)

// Fungsi untuk mengurutkan array menggunakan Selection Sort
func selectionSort(arr []int, n int) {
    for i := 0; i < n-1; i++ {
        idxMin := i
        for j := i + 1; j < n; j++ {
            // Cari elemen terkecil
            if arr[j] < arr[idxMin] {
                idxMin = j
            }
        }
        // Tukar elemen terkecil dengan elemen di posisi i
        arr[i], arr[idxMin] = arr[idxMin], arr[i]
    }
}

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

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

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

        // Urutkan array dari terkecil ke terbesar
        selectionSort(arr, m)

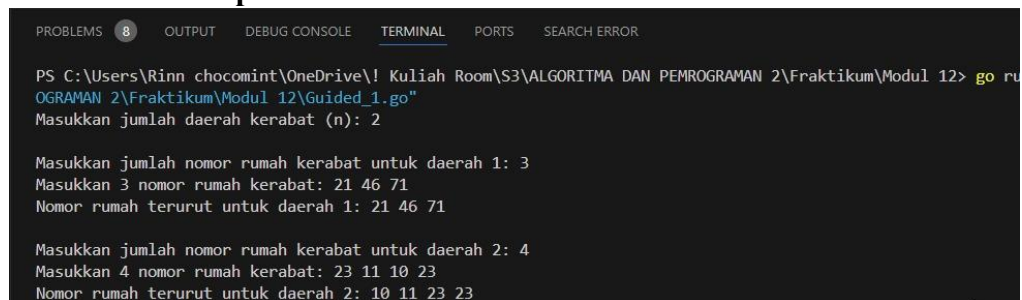
        // Tampilkan hasil
```

```

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

```

## Screenshoot Output



```

PS C:\Users\Rinn chocomint\OneDrive\! Kuliah Room\S3\ALGORITMA DAN PEMROGRAMAN 2\Fraktikum\Modul 12> go run OGRAMAN 2\Fraktikum\Modul 12\Guided_1.go
Masukkan jumlah daerah kerabat (n): 2

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 3
Masukkan 3 nomor rumah kerabat: 21 46 71
Nomor rumah terurut untuk daerah 1: 21 46 71

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 4
Masukkan 4 nomor rumah kerabat: 23 11 10 23
Nomor rumah terurut untuk daerah 2: 10 11 23 23

```

## Deskripsi Program

Program itu untuk mengurutkan nomor rumah dalam sebuah array menggunakan algoritma “**Selection Sort**”. Fungsi ini menerima array “**arr**” dan jumlah elemen “**n**”, kemudian memprosesnya secara iteratif untuk mencari elemen terkecil di setiap iterasi dan menukarnya dengan elemen pada indeks saat ini. Dimulai dengan meminta pengguna memasukkan jumlah daerah dan jumlah nomor rumah di setiap daerah, kemudian membaca nilai-nilai nomor rumah untuk tiap daerah. Selanjutnya, memanggil fungsi “**selectionSort**” untuk mengurutkan nomor rumah dari terkecil ke terbesar. Hasil akhirnya menampilkan nomor rumah yang sudah terurut untuk masing-masing daerah.

## 2. Sourcecode

```

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")
    }
}

```

### Screenshoot Output

```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH
PS C:\Users\Rinn chocomint\OneDrive\! Kuliah Room\S3\ALGORITMA 2\Fraktikum\Modul 12\Guided_2.go
Masukkan data integer (akhiri dengan bilangan negatif):
23 11 10 23 27 -1
Array setelah diurutkan:
10 11 23 23 27
Data berjarak tidak tetap

```

### Deskripsi Program

Program bertujuan untuk mengurutkan data integer dalam sebuah array menggunakan algoritma “**Insertion Sort**” dan memeriksa apakah selisih antara elemen-elemen dalam array tetap. Fungsi “**insertionSort**” menerima array “**arr**” dan jumlah elemen “**n**”, kemudian memprosesnya secara iteratif untuk menyisipkan elemen ke posisi yang tepat. Selanjutnya, fungsi “**isConstantDifference**” memeriksa konsistensi selisih antar elemen array dan mengembalikan statusnya beserta nilai selisih jika tetap. Dimulai dengan meminta pengguna memasukkan data integer hingga bilangan negatif ditemukan, kemudian mengurutkan array menggunakan fungsi “**insertionSort**”. Hasil akhir menampilkan array yang telah diurutkan serta status konsistensi jarak elemen array, apakah tetap atau tidak tetap.

## III. UNGUIDED

1. 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**

**dekati** 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.

### Sourcecode

```
package main

import (
    "bufio"
    "fmt"
    "os"
    "sort"
    "strconv"
    "strings"
)

func main() {
    fmt.Println("Masukan")
    fmt.Println("=====")

    scanner := bufio.NewScanner(os.Stdin)

    scanner.Scan()
    jumlahDaerah, _ := strconv.Atoi(scanner.Text())

    var hasil [][]int

    for i := 0; i < jumlahDaerah; i++ {
        scanner.Scan()
        baris := scanner.Text()
        data := strings.Fields(baris)

        var ganjil, genap []int
        for _, nomorStr := range data[1:] {
            nomor, _ := strconv.Atoi(nomorStr)
            if nomor%2 == 0 {
                genap = append(genap, nomor)
            } else {
                ganjil = append(ganjil, nomor)
            }
        }
    }
}
```

```

        sort.Ints(genap)
        sort.Sort(sort.Reverse(sort.IntSlice(ganjil)))

        hasil = append(hasil, append(ganjil, genap...))
    }
    fmt.Println("=====")

    fmt.Printf("\n")
    fmt.Println("Keluaran")
    fmt.Println("=====")

    for _, daerah := range hasil {
        for i, nomor := range daerah {
            if i > 0 {
                fmt.Print(" ")
            }
            fmt.Print(nomor)
        }
        fmt.Println()
    }
    fmt.Printf("=====\n")
}

```

## Screenshot Output

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS C:\Users\Rinn chocomint\OneDrive\! Kuliah Room\S3\ALGORITMA DAN PEMROGRAMAN
OGRAMAN 2\Fraktikum\Modul 12\Unguided_1.go"
Masukan
=====
3
5 2 1 7 9 13
6 189 15 27 39 75 133
3 4 9 1
=====

Keluaran
=====
13 9 7 1 2
189 133 75 39 27 15
9 1 4
=====

```

## Deskripsi Program

Program itu untuk memproses data nomor rumah dari beberapa daerah, memisahkan nomor ganjil dan genap, mengurutkannya secara terpisah, lalu menggabungkannya kembali dalam format tertentu. Dimulai dengan meminta jumlah daerah dan daftar nomor rumah dari setiap daerah sebagai input. Nomor rumah ganjil diurutkan menurun, sementara nomor genap



diurutkan naik menggunakan fungsi bawaan “**sort**”. Setelah pengurutan, nomor ganjil dan genap digabungkan dalam satu array untuk tiap daerah. Hasil akhirnya menampilkan daftar nomor rumah yang telah diurutkan sesuai aturan untuk setiap daerah.

2. Kompetisi pemrograman yang baru saja berlalu diikuti oleh 17 tim dari berbagai perguruan tinggi ternama. Dalam kompetisi tersebut, setiap tim berlomba untuk menyelesaikan sebanyak mungkin problem yang diberikan. Dari 13 problem yang diberikan, ada satu problem yang menarik. Problem tersebut mudah dipahami, hampir semua tim mencoba untuk menyelesaikannya, tetapi hanya 3 tim yang berhasil. Apa sih problemnya?

“Median adalah nilai tengah dari suatu koleksi data yang sudah terurut. Jika jumlah data genap, maka nilai median adalah rerata dari kedua nilai tengahnya. Pada problem ini, semua data merupakan bilangan bulat positif, dan karenanya rerata nilai tengah dibulatkan ke bawah.”

Buatlah program **median** yang mencetak nilai median terhadap seluruh data yang sudah terbaca, jika data yang dibaca saat itu adalah 0.

**Masukan** berbentuk rangkaian bilangan bulat. Masukan tidak akan berisi lebih dari 1000000 data, tidak termasuk bilangan 0. Data 0 merupakan tanda bahwa median harus dicetak, tidak termasuk data yang dicari mediannya. Data masukan diakhiri dengan bilangan bulat -5313.

**Keluaran** adalah median yang diminta, satu data per baris.

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

### Sourcecode

```
package main

import (
    "bufio"
    "fmt"
    "os"
    "sort"
    "strconv"
    "strings"
)

func main() {
    fmt.Println("Masukkan :")
    fmt.Println("=====")
    =")
}
```

```

scanner := bufio.NewScanner(os.Stdin)
scanner.Scan()
masukan := scanner.Text()

angkaInput := strings.Split(masukan, " ")
fmt.Println("=====
=)

fmt.Printf("\n")

var angka []int
fmt.Println("Keluaran :")
fmt.Println("=====
=)

for _, bilanganStr := range angkaInput {
    bilangan, _ := strconv.Atoi(bilanganStr)

    if bilangan == -5313 {
        break
    }

    if bilangan == 0 {
        median := hitungMedian(angka)
        fmt.Println(median)
    } else {
        angka = append(angka, bilangan)
    }
}
fmt.Println("=====
=)
}

func hitungMedian(angka []int) int {
    sort.Ints(angka)

    jumlah := len(angka)
    if jumlah%2 == 1 {
        return angka[jumlah/2]
    } else {
        return (angka[jumlah/2-1] + angka[jumlah/2]) / 2
    }
}

```

## Screenshoot Output

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS C:\Users\Rinn chocomint\OneDrive\! Kuliah Room\S3\ALGORITMA DAN PR
OGRAMAN 2\Fraktikum\Modul 12\Unguided_2.go
Masukkan :
=====
7 23 11 0 5 19 2 29 3 13 17 0 -5313
=====

Keluaran :
=====
11
12
=====

```

### Deskripsi Program

Program itu untuk membaca serangkaian angka, menghitung median dari angka-angka yang sudah dimasukkan, dan menampilkan hasilnya setiap kali angka “0” ditemukan. Dimulai dengan meminta pengguna memasukkan data angka dalam satu baris yang dipisahkan oleh spasi. Program akan membaca angka-angka tersebut secara berurutan, mengabaikan angka -”5313” sebagai tanda berhenti, dan memproses angka “0” sebagai perintah untuk menghitung dan menampilkan median dari angka yang telah dimasukkan sebelumnya. Fungsi “**hitungMedian**” digunakan untuk menghitung median dengan cara mengurutkan array angka menggunakan fungsi bawaan “**sort.Ints**”. Hasil akhir menampilkan nilai median setiap kali angka “0” ditemukan hingga program dihentikan.

3. 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

```
package main
import (
    "fmt"
)

type Buku struct {
    id      int
    judul   string
    penulis string
    penerbit string
    tahun   int
    rating  int
}

type DaftarBuku []Buku

func DaftarkanBuku(pustaka *DaftarBuku, n *int) {
    fmt.Println("Masukkan data buku (id, judul, penulis, penerbit, tahun, rating):")
    for i := 0; i < *n; i++ {
        var id, tahun, rating int
        var judul, penulis, penerbit string
        fmt.Scanln(&id, &judul, &penulis, &penerbit, &tahun, &rating)
        *pustaka = append(*pustaka, Buku{id, judul, penulis, penerbit, tahun, rating})
    }
    fmt.Printf("===== \n")
}

func CetakTerfavorit(pustaka DaftarBuku, n int) {
    maxRating := pustaka[0].rating
    var favorit Buku
    for _, buku := range pustaka {
        if buku.rating > maxRating {
            maxRating = buku.rating
            favorit = buku
        }
    }
    fmt.Printf("Buku Terfavorit: %s oleh %s (%s, %d)\n", favorit.judul, favorit.penulis, favorit.penerbit, favorit.tahun)
}

func UrutBuku(pustaka *DaftarBuku, n int) {
    for i := 1; i < n; i++ {
```

```

        key := (*pustaka)[i]
        j := i - 1
        for j >= 0 && (*pustaka)[j].rating < key.rating {
            (*pustaka)[j+1] = (*pustaka)[j]
            j--
        }
        (*pustaka)[j+1] = key
    }
}

func Cetak5Terbaru(pustaka DaftarBuku, n int) {
    fmt.Println("5 Buku dengan Rating Tertinggi:")
    for i := 0; i < 5 && i < n; i++ {
        buku := pustaka[i]
        fmt.Printf("%s oleh %s (%s, %d) - Rating: %d\n", buku.judul,
buku.penulis, buku.penerbit, buku.tahun, buku.rating)
    }
}

func CariBuku(pustaka DaftarBuku, n, r int) {
    low, high := 0, n-1
    for low <= high {
        mid := (low + high) / 2
        if pustaka[mid].rating == r {
            buku := pustaka[mid]
            fmt.Printf("Ditemukan: %s oleh %s (%s, %d) -
Rating: %d\n", buku.judul, buku.penulis, buku.penerbit, buku.tahun, buku.rating)
            return
        } else if pustaka[mid].rating < r {
            high = mid - 1
        } else {
            low = mid + 1
        }
    }
    fmt.Println("Tidak ada buku dengan rating seperti itu.")
}

func main() {
    var pustaka DaftarBuku
    var n int

    fmt.Print("Input Jumlah Buku: ")
    fmt.Scanln(&n)

    DaftarkanBuku(&pustaka, &n)
    CetakTerfavorit(pustaka, n)
    UrutBuku(&pustaka, n)
}

```

```

    Cetak5Terbaru(pustaka, n)

    var r int
    fmt.Print("Masukkan rating yang ingin dicari: ")
    fmt.Scanln(&r)
    CariBuku(pustaka, n, r)
}

```

## Screenshoot Output

```

PS C:\Users\Rinn chocomint\OneDrive\! Kuliah Room\S3\ALGORITMA DAN PEMROGRAMAN 2\Fraktikum\Modul 12\Unguided_3.go"
Input Jumlah Buku: 5
Masukkan data buku (id, judul, penulis, penerbit, tahun, rating):
1 Roshidere SunSunSun Kadokawa 2023 86
2 Harmony RinChocomint Gramedia 2024 91
3 JLPT Irma GentaGroup 2024 84
4 Kalkulus Ridwan GegasMedia 2024 76
5 PemerogramnC&C++ AdamMukharil GegasMedia 2018 78
=====
Buku Terfavorit: Harmony oleh RinChocomint (Gramedia, 2024)
5 Buku dengan Rating Tertinggi:
Harmony oleh RinChocomint (Gramedia, 2024) - Rating: 91
Roshidere oleh SunSunSun (Kadokawa, 2023) - Rating: 86
JLPT oleh Irma (GentaGroup, 2024) - Rating: 84
PemerogramnC&C++ oleh AdamMukharil (GegasMedia, 2018) - Rating: 78
Kalkulus oleh Ridwan (GegasMedia, 2024) - Rating: 76
Masukkan rating yang ingin dicari: 91
Ditemukan: Harmony oleh RinChocomint (Gramedia, 2024) - Rating: 91

```

## Deskripsi Program

Program itu untuk mengelola data koleksi buku dengan fitur pendaftaran buku, pencarian buku berdasarkan rating, pengurutan berdasarkan rating, dan menampilkan buku-buku terbaik. Dimulai dengan meminta jumlah buku yang akan didaftarkan, lalu menginputkan data buku seperti id, judul, penulis, penerbit, tahun, dan rating melalui fungsi “DaftarkanBuku”. Program ini menampilkan buku dengan rating tertinggi melalui fungsi “**CetakTerfavorit**”, kemudian mengurutkan koleksi berdasarkan rating tertinggi ke terendah menggunakan fungsi “**UrutBuku**”. Selanjutnya, program menampilkan hingga 5 buku terbaik berdasarkan rating dengan fungsi “**Cetak5Terbaru**”. Akhirnya, pengguna dapat mencari buku dengan rating tertentu menggunakan fungsi “**CariBuku**”, yang menggunakan metode pencarian biner untuk efisiensi.