

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 12 & 13  
PENGURUTAN DATA**



**Disusun Oleh :**

**Zahra Tsurayya Poetri / 231110217**

**IF 11 - 05**

**Dosen Pengampu :**

**Arif Amrulloh**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. DASAR TEORI

### 1. Ide Algoritma Selection Sort

Pengurutan secara seleksi ini idenya adalah mencari nilai ekstrim pada sekumpulan data, kemudian meletakkan pada posisi yang seharusnya. Pada penjelasan berikut ini data akan diurut membesar (*ascending*), dan data dengan indeks kecil ada di “kiri” dan indeks besar ada di “kanan”.

- 1) Cari nilai terkecil di dalam rentang data tersisa
- 2) Pindahkan/tukar tempat dengan data yang berada pada posisi paling kiri pada rentang data tersebut.
- 3) Ulangi proses ini sampai tersisa hanya satu data saja.

Algoritma ini dikenal juga dengan nama ***Selection Sort***, yang mana pada algoritma ini melibatkan dua proses yaitu pencarian indeks nilai ekstrim dan proses pertukaran dua nilai atau *swap*.

	Notasi Algoritma	Notasi dalam bahasa Go
1	$i \leftarrow 1$	$i = 1$
2	while $i \leq n-1$ do	for $i \leq n-1$ {
3	$idx\_min \leftarrow i - 1$	$idx\_min = i - 1$
4	$j \leftarrow i$	$j = i$
5	while $j < n$ do	for $j < n$ {
6	if $a[idx\_min] > a[j]$ then	if $a[idx\_min] > a[j]$ {
7	$idx\_min \leftarrow j$	$idx\_min = j$
8	endif	}
9	$j \leftarrow j + 1$	$j = j + 1$
10	endwhile	}
11	$t \leftarrow a[idx\_min]$	$t = a[idx\_min]$
12	$a[idx\_min] \leftarrow a[i-1]$	$a[idx\_min] = a[i-1]$
13	$a[i-1] \leftarrow t$	$a[i-1] = t$
14	$i \leftarrow i + 1$	$i = i + 1$
15	endwhile	}

### 2. Algoritma Selection Sort

Adapun algoritma *selection sort* pada untuk mengurutkan array bertipe data bilangan bulat secara membesar atau ascending adalah sebagai berikut ini!

```

..   ...
5   type arrInt [4321]int
..   ...
15  func selectionSort1(T *arrInt, n int){
16  /* I.S. terdefinisi array T yang berisi n bilangan bulat
17     F.S. array T terurut secara ascending atau membesar dengan SELECTION SORT */
18     var t, i, j, idx_min int
19
20     i = 1
21     for i <= n-1 {
22         idx_min = i - 1
23         j = i
24         for j < n {
25             if T[idx_min] > T[j] {
26                 idx_min = j
27             }
28             j = j + 1
29         }
30         t = T[idx_min]
31         T[idx_min] = T[i-1]
32         T[i-1] = t
33         i = i + 1
34     }
35 }

```

Sama halnya apabila array yang akan diurutkan adalah bertipe data struct, maka tambahkan field pada saat proses perbandingan nilai ekstrim, kemudian tipe data dari variabel *t* sama dengan struct dari arraynya.

```

..   ...
5   type mahasiswa struct {
..       nama, nim, kelas, jurusan string
..       ipk float64
..   }
..   type arrMhs [2023]mahasiswa
..   ...
15  func selectionSort2(T * arrMhs, n int){
16  /* I.S. terdefinisi array T yang berisi n data mahasiswa
17     F.S. array T terurut secara ascending atau membesar berdasarkan ipk dengan
18     menggunakan algoritma SELECTION SORT */
19     var i, j, idx_min int
20     var t mahasiswa
21     i = 1
22     for i <= n-1 {
23         idx_min = i - 1
24         j = i
25         for j < n {
26             if T[idx_min].ipk > T[j].ipk {
27                 idx_min = j
28             }
29             j = j + 1
30         }
31         t = T[idx_min]
32         T[idx_min] = T[i-1]
33         T[i-1] = t
34         i = i + 1
35     }
36 }

```

## II. GUIDED

### Guided 1

#### Program Mengurutkan Nomor Rumah Kerabat Menggunakan Selection Sort

##### 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 D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "d:\k
Masukkan jumlah daerah kerabat (n): 3

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 3
Masukkan 3 nomor rumah kerabat: 2 7 9
Nomor rumah terurut untuk daerah 1: 2 7 9

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 3
Masukkan 3 nomor rumah kerabat: 5 2 1
Nomor rumah terurut untuk daerah 2: 1 2 5

Masukkan jumlah nomor rumah kerabat untuk daerah 3: 3
Masukkan 3 nomor rumah kerabat: 4 5 7 1
Nomor rumah terurut untuk daerah 3: 4 5 7
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> █
```

## Deskripsi Program

Program di atas digunakan untuk mengurutkan nomor rumah kerabat dari beberapa daerah menggunakan algoritma Selection Sort. Pertama, program meminta pengguna untuk memasukkan jumlah daerah yang memiliki rumah kerabat. Kemudian, untuk setiap daerah, program meminta pengguna untuk memasukkan jumlah nomor rumah kerabat di daerah tersebut. Setelah itu, pengguna diminta untuk memasukkan nomor rumah untuk setiap daerah, yang kemudian disimpan dalam array. Program selanjutnya akan mengurutkan nomor rumah di setiap daerah dari terkecil ke terbesar menggunakan algoritma Selection Sort. Program akan menampilkan hasil nomor rumah yang telah terurut untuk setiap daerah

## Guided 2

### Program Pengurutan dan Pemeriksaan Selisih Jarak Tetap pada Array Menggunakan Selection Sort

#### 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

```

PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "d:\k
Masukkan data integer (akhiri dengan bilangan negatif):
10 6 2 4 8 -1
Array setelah diurutkan:
2 4 6 8 10
Data berjarak 2
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "d:\k
Masukkan data integer (akhiri dengan bilangan negatif):
5 2 8 14 7 -1
Array setelah diurutkan:
2 5 7 8 14
Data berjarak tidak tetap
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12>

```

### Deskripsi Program

Program di atas digunakan untuk mengurutkan data integer menggunakan algoritma Insertion Sort dan memeriksa apakah selisih antar elemen array tetap (konstan). Program meminta pengguna untuk menginputkan data integer satu per satu, dan proses input akan berhenti ketika pengguna memasukkan bilangan negatif. Setelah data dimasukkan, program mengurutkan array menggunakan Insertion Sort, lalu memeriksa apakah selisih antar elemen dalam array tetap konstan. Program akan

menampilkan hasil array yang telah diurutkan dan, jika selisih antar elemen tetap, menampilkan jarak (selisih) antara elemen-elemen tersebut.

### III. 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 biri 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"
)

// Fungsi untuk mengurutkan array menggunakan Selection Sort
func selectionSort(arr []int, ascending bool) {
    for i := 0; i < len(arr)-1; i++ {
        idx := i
        for j := i + 1; j < len(arr); j++ {
            // Cari elemen terkecil (untuk ascending) atau terbesar (untuk descending)
            if (ascending && arr[j] < arr[idx]) ||
                (!ascending && arr[j] > arr[idx]) {
                idx = j
            }
        }
        // Tukar elemen terkecil/terbesar dengan elemen di posisi i
        arr[i], arr[idx] = arr[idx], 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])
    }

    // Pisahkan nomor ganjil dan genap
    ganjil := []int{}
    genap := []int{}
    for _, num := range arr {
        if num%2 != 0 {
            ganjil = append(ganjil, num)
        } else {
            genap = append(genap, num)
        }
    }

    // Urutkan nomor ganjil secara ascending dan
nomor genap secara descending
    selectionSort(ganjil, true) // Urutkan
ganjil secara menaik (ascending)
    selectionSort(genap, false) // Urutkan
genap secara menurun

    // Tampilkan hasil
    fmt.Printf("Nomor rumah terurut untuk
daerah %d: ", daerah)

```

```

        // Tampilkan nomor ganjil
        for _, num := range ganjil {
            fmt.Printf("%d ", num)
        }
        // Tampilkan nomor genap
        for _, num := range genap {
            fmt.Printf("%d ", num)
        }
        fmt.Println()
    }
}

```

### Screenshoot Output

```

PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "d:\kul
Masukkan jumlah daerah kerabat (n): 3

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 6
Masukkan 6 nomor rumah kerabat: 1 6 13 7 2 8
Nomor rumah terurut untuk daerah 1: 1 7 13 8 6 2

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 7
Masukkan 7 nomor rumah kerabat: 151 21 35 75 3 89 9
Nomor rumah terurut untuk daerah 2: 3 9 21 35 75 89 151

Masukkan jumlah nomor rumah kerabat untuk daerah 3: 4
Masukkan 4 nomor rumah kerabat: 2 16 22 8
Nomor rumah terurut untuk daerah 3: 22 16 8 2
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12>

```

### Deskripsi Program

Program ini digunakan untuk mengurutkan nomor rumah kerabat dengan cara memisahkan nomor ganjil dan genap. Nomor rumah ganjil akan diurutkan secara menaik (ascending), sedangkan nomor rumah genap diurutkan secara menurun (descending). Program menggunakan algoritma

Selection Sort untuk mengurutkan nomor rumah. Pengguna dapat memasukkan jumlah daerah kerabat, jumlah nomor rumah di setiap daerah, dan nomor rumah kerabat masing-masing. Program akan memisahkan nomor rumah menjadi dua kelompok: ganjil dan genap, lalu mengurutkannya sesuai dengan aturan yang telah disebutkan. Setelah itu, program menampilkan nomor rumah terurut, dimulai dengan nomor ganjil yang terurut menaik, diikuti dengan nomor genap yang terurut menurun.

## 2. Soal Studi Case

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.

**Keterangan:**

Sampai bilangan 0 yang pertama, data terbaca adalah 7 23 11, setelah tersusun: 7 11 23, maka median saat itu adalah 11.

Sampai bilangan 0 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 0, urutkanlah data yang sudah tersimpan dengan menggunakan metode insertion sort dan ambil mediannya.

**Sourcecode**

```
package main

import (
    "fmt"
)

func selectionSort(arr []int) {
    // Mengurutkan array dengan menggunakan selection
    sort
    for i := 0; i < len(arr)-1; i++ {
        // Temukan elemen terkecil dalam array yang
        belum terurut
        minIdx := i
        for j := i + 1; j < len(arr); j++ {
            if arr[j] < arr[minIdx] {
                minIdx = j
            }
        }
        // Tukar elemen
        arr[i], arr[minIdx] = arr[minIdx], arr[i]
    }
}
```



```

    }

    }

    // Tukar elemen terkecil dengan elemen
    pertama yang belum terurut
    arr[i], arr[minIdx] = arr[minIdx], arr[i]
}

func cariMedian(arr []int) int {
    // Mencari median
    n := len(arr)
    if n%2 == 1 {
        // Jika jumlah elemen ganjil, median adalah
        elemen tengah
        return arr[n/2]
    } else {
        // Jika jumlah elemen genap, median adalah
        rata rata dua elemen tengah
        tengah1, tengah2 := arr[n/2-1], arr[n/2]
        return (tengah1 + tengah2) / 2
    }
}

func main() {
    var input int
    var data []int

    // Membaca input sampai menemukan angka -5313
    for {
        fmt.Scan(&input)

        // Jika input adalah -5313, maka berhenti
        if input == -5313 {
            break
        }
    }
}

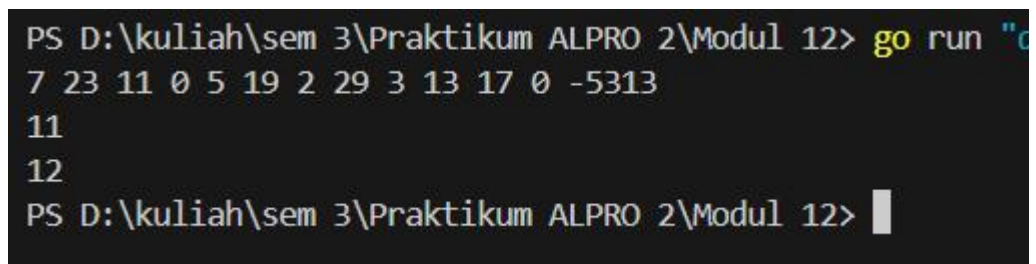
```

```

        // Jika input adalah 0, cetak median
        if input == 0 {
            // urutkan data dengan selection sort
            selectionSort(data)
            // Cetak median
            median := cariMedian(data)
            fmt.Println(median)
        } else {
            // Jika input bukan 0, simpan data
            data = append(data, input)
        }
    }
}

```

### Screenshoot Output



```

PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "c
7 23 11 0 5 19 2 29 3 13 17 0 -5313
11
12
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12>

```

### Deskripsi Program

Program di atas merupakan program menghitung nilai median dari serangkaian bilangan bulat yang dimasukkan oleh pengguna. Program menerima input berupa bilangan bulat satu per satu. Proses input berlanjut hingga pengguna memasukkan angka -5313, yang menandakan akhir dari input. Jika pengguna memasukkan angka 0, program akan mengurutkan data yang sudah dimasukkan menggunakan algoritma Selection Sort. Setelah data terurut, program menghitung median. Jika jumlah elemen ganjil, maka median elemen yang ada di posisi tengah. Sedangkan jika jumlah elemen genap, median dihitung dengan cara menjumlahkan dua elemen tengah dan membaginya dengan dua. Setiap kali angka 0

dimasukkan, program akan menampilkan hasil perhitungan median berdasarkan data yang sudah dimasukkan.

### 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 Daftar Buku = array [1..nMax] of Buku

Pustaka : Daftar Buku

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"
)

const nMax int = 7919

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

type DaftarBuku [nMax]Buku

var Pustaka DaftarBuku

```

```

var nPustaka int

// DaftarkanBuku untuk memasukkan data buku
func DaftarkanBuku(pustaka *DaftarBuku, n int) {
    fmt.Println("Masukkan data buku (id, judul,
penulis, penerbit, tahun, rating, eksemplar):")
    for i := 0; i < n; i++ {
        fmt.Printf("Buku ke-%d:\n", i+1)
        // Pastikan memasukkan rating yang valid
        fmt.Scan(&pustaka[i].id, &pustaka[i].judul,
&pustaka[i].penulis, &pustaka[i].penerbit,
&pustaka[i].tahun, &pustaka[i].rating,
&pustaka[i].eksemplar)
    }
}

// CetakTerfavorit untuk menampilkan buku dengan rating
tertinggi
func CetakTerfavorit(pustaka DaftarBuku, n int) {
    maxRating := -1
    var terfavorit Buku
    for i := 0; i < n; i++ {
        if pustaka[i].rating > maxRating {
            maxRating = pustaka[i].rating
            terfavorit = pustaka[i]
        }
    }
    fmt.Println("\nBuku Terfavorit:")
    fmt.Printf("Judul: %s\nPenulis: %s\nPenerbit: %s\nT
ahun: %d\n", terfavorit.judul, terfavorit.penulis,
terfavorit.penerbit, terfavorit.tahun)
}

// UrutBuku untuk mengurutkan buku berdasarkan rating
menggunakan Insertion Sort
func UrutBuku(pustaka *DaftarBuku, n int) {

```

```

        for i := 1; i < n; i++ {
            key := pustaka[i]
            j := i - 1
            // Urutkan buku berdasarkan rating dengan
metode Insertion Sort
            for j >= 0 && pustaka[j].rating < key.rating
{
                pustaka[j+1] = pustaka[j]
                j--
            }
            pustaka[j+1] = key
        }
    }

// Cetak5Terbaru untuk menampilkan 5 buku dengan rating
tertinggi
func Cetak5Terbaru(pustaka DaftarBuku, n int) {
    fmt.Println("\n5 Buku dengan Rating Tertinggi:")
    for i := 0; i < 5 && i < n; i++ {
        fmt.Printf("%d. %s\n", i+1, pustaka[i].judul)
    }
}

// CariBuku untuk mencari buku berdasarkan rating
menggunakan pencarian biner
func CariBuku(pustaka DaftarBuku, n int, r int) {
    // Pencarian biner
    idxAwal, idxAkhir := 0, n-1
    var hasil []Buku
    for idxAwal <= idxAkhir {
        idxTengah := (idxAwal + idxAkhir) / 2
        if pustaka[idxTengah].rating == r {
            // Cari ke kiri
            i := idxTengah
            for i >= 0 && pustaka[i].rating == r {
                hasil = append(hasil, pustaka[i])
            }
        }
    }
}

```

```

        i--
    }
    // Cari ke kanan
    i = idxTengah + 1
    for i < n && pustaka[i].rating == r {
        hasil = append(hasil, pustaka[i])
        i++
    }
    break
} else if pustaka[idxTengah].rating > r {
    idxAkhir = idxTengah - 1
} else {
    idxAwal = idxTengah + 1
}
}

// Periksa apakah ada buku dengan rating tersebut
if len(hasil) > 0 {
    fmt.Printf("\nBuku dengan rating %d
ditemukan:\n", r)
    for _, buku := range hasil {

        fmt.Printf("Judul: %s\nPenulis: %s\nPenerbit: %s\nT
ahun: %d\nEksemplar: %d\nRating: %d\n\n",
            buku.judul, buku.penulis,
buku.penerbit, buku.tahun, buku.eksemplar, buku.rating)
    }
} else {
    fmt.Println("\nTidak ada buku dengan rating
seperti itu.")
}
}

func main() {
    // Input jumlah buku

```

```
    fmt.Print("Masukkan jumlah buku di perpustakaan: ")
    fmt.Scan(&nPustaka)

    // Daftarkan buku-buku
    DaftarkanBuku(&Pustaka, nPustaka)

    // Cetak buku terfavorit (rating tertinggi)
    CetakTerfavorit(Pustaka, nPustaka)

    // Urutkan buku berdasarkan rating menggunakan
Insertion Sort
    UrutBuku(&Pustaka, nPustaka)

    // Cetak 5 buku dengan rating tertinggi
    Cetak5Terbaru(Pustaka, nPustaka)

    // Input rating yang dicari
    var rating int
    fmt.Print("\nMasukkan rating yang ingin dicari: ")
    fmt.Scan(&rating)

    // Cari buku dengan rating tersebut
    CariBuku(Pustaka, nPustaka, rating)
}
```



## Screenshoot Output

```
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> go run "d:\kuliah\sem 3\Praktikum
Masukkan jumlah buku di perpustakaan: 5
Masukkan data buku (id, judul, penulis, penerbit, tahun, rating, eksemplar):
Buku ke-1:
buku101 AtomicHabits JamesClear KompasGramedia 2018 5 3000
Buku ke-2:
buku102 FilosofiTeras HenryManampiring Kompas 2018 4 2500
Buku ke-3:
buku103 BicaraItuAdaSeninya OhSuHyang BIP 2018 4 2700
Buku ke-4:
buku104 KomunikasiItuAdaSeninya OhSuHyang BIP 2020 3 1900
Buku ke-5:
buku105 TheCompass HenryManampiring GagasMedia 2023 4 2500

Buku Terfavorit:
Judul: AtomicHabits
Penulis: JamesClear
Penerbit: KompasGramedia
Tahun: 2018

5 Buku dengan Rating Tertinggi:
1. AtomicHabits
2. FilosofiTeras
3. BicaraItuAdaSeninya
4. TheCompass
5. KomunikasiItuAdaSeninya

Masukkan rating yang ingin dicari: 2

Tidak ada buku dengan rating seperti itu.
PS D:\kuliah\sem 3\Praktikum ALPRO 2\Modul 12> █
```

## Deskripsi Program

Program di atas merupakan program mengelola data buku di sebuah perpustakaan dengan menggunakan struktur data struct dan array. Pengguna diminta untuk memasukkan jumlah buku dan kemudian menginputkan data buku yang mencakup ID, judul, penulis, penerbit, tahun terbit, rating, dan jumlah eksemplar. Setelah data buku dimasukkan, program akan mengurutkan buku berdasarkan rating tertinggi menggunakan algoritma Insertion Sort. Program kemudian menampilkan buku terfavorit, yaitu buku dengan rating tertinggi, serta lima buku dengan rating tertinggi. Selain itu, pengguna dapat mencari buku berdasarkan

rating tertentu, dan jika ada buku dengan rating tersebut, program akan menampilkan semua buku yang memiliki rating yang sama sesuai masukan pengguna.

### **Daftar Pustaka**

- [1] Susilowati, A. Zahara, A. (2024). *Modul 12 & 13 Praktikum Algoritma Pemrograman 2*. Program Studi Teknik Informatika, Telkom University Purwokerto.