

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL XII & XIII
PENGURUTAN DATA



Disusun Oleh :

Rasyid Nafsyarie / 2311102011

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

Pengurutan data merupakan salah satu operasi penting dalam pemrograman, termasuk dalam pengembangan aplikasi berbasis Golang. Operasi pengurutan bertujuan untuk mengatur elemen-elemen dalam suatu koleksi data, seperti array, slice, atau list, berdasarkan urutan tertentu, baik secara **ascending** (menaik) maupun **descending** (menurun).

1. Konsep Dasar Pengurutan

Pengurutan data bertujuan untuk mempermudah pencarian, pengolahan, dan analisis data. Metode pengurutan dapat dibedakan menjadi dua kategori utama:

- **Internal Sorting:** Dilakukan di dalam memori komputer, cocok untuk data kecil hingga sedang.
- **External Sorting:** Dilakukan dengan memanfaatkan memori eksternal, digunakan untuk dataset besar yang tidak dapat dimuat sepenuhnya ke dalam memori.

2. Metode Pengurutan

Golang menyediakan beberapa metode untuk mengurutkan data, baik menggunakan fungsi bawaan atau implementasi manual.

a. Pengurutan Bawaan Golang

Golang memiliki paket bawaan sort yang sangat efisien untuk mengurutkan data. Paket ini menyediakan beberapa fungsi utama:

- `sort.Ints(slice []int)`: Mengurutkan slice tipe int secara ascending.
- `sort.Strings(slice []string)`: Mengurutkan slice tipe string secara ascending.

- `sort.Float64s(slice []float64)`: Mengurutkan slice tipe `float64` secara ascending.

Selain itu, terdapat interface `sort.Interface` untuk pengurutan yang lebih kompleks, dengan tiga metode yang harus diimplementasikan:

- `Len() int`: Mengembalikan panjang data.
- `Less(i, j int) bool`: Menentukan apakah elemen `i` kurang dari elemen `j`.
- `Swap(i, j int)`: Menukar elemen `i` dan `j`.

b. Implementasi Manual (Custom Sorting)

Untuk kebutuhan khusus, algoritma pengurutan manual seperti **Bubble Sort**, **Selection Sort**, atau **Quick Sort** dapat diimplementasikan dalam Golang.

II. GUIDED

1. 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])
        }

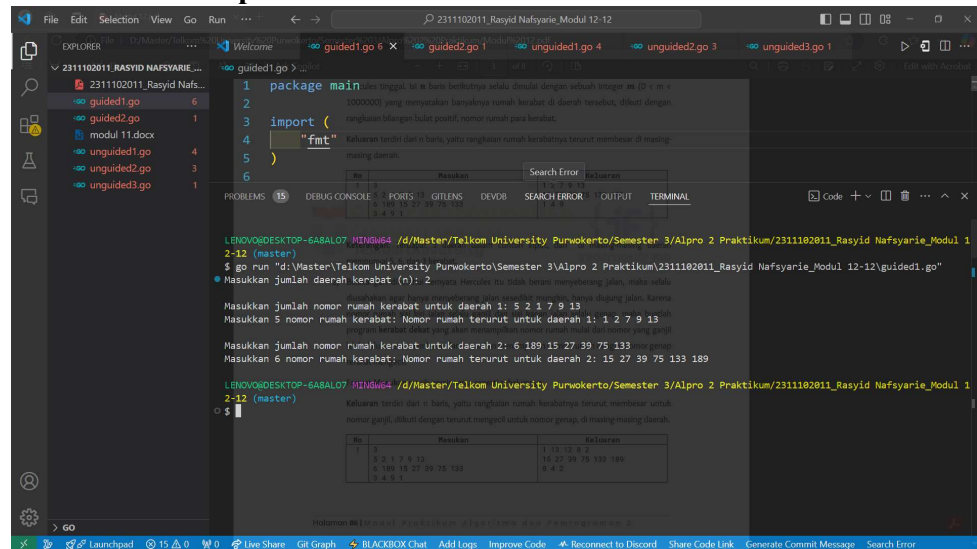
        // Urutkan array dari terkecil ke terbesar
        selectionSort(arr, m)
    }
}
```

```

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

```

Screenshoot Output



Deskripsi Program

Fungsi selectionSort: Fungsi ini bertanggung jawab untuk mengurutkan array. Fungsi main: Fungsi utama yang mengelola alur program, termasuk pengambilan input dari pengguna dan pemanggilan fungsi pengurutan.

2. Guided 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

```

package main
import (
    "fmt"
)

func main() {
    // ... (code omitted for brevity) ...
}

```

LENOVO\DESKTOP-GABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Alpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 12-12 (master)

\$ go run "d:/Master/Telkom University Purwokerto/Semester 3/Alpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 12-12/guided1.go"

Masukkan jumlah daerah kerabat (n): 2

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 5 2 1 7 9 13

Masukkan 5 nomor rumah kerabat: Nomor rumah terurut untuk daerah 1: 1 2 7 9 13

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 6 189 15 27 39 75 133

Masukkan 6 nomor rumah kerabat: Nomor rumah terurut untuk daerah 2: 15 27 39 75 133 189

LENOVO\DESKTOP-GABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Alpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 12-12 (master)

\$ go run "d:/Master/Telkom University Purwokerto/Semester 3/Alpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 12-12/guided2.go"

Masukkan data integer (akhiri dengan bilangan negatif):

7 23 11 8 5 19 2 29 3 13 17 0 -5

Array setelah diurutkan:

0 2 3 5 7 8 11 13 17 19 23 29

Data berjarak tidak tetap

No	Nilai	Keluaran
1	7 23 11 8 5 19 2 29 3 13 17 0 -5	11

Deskripsi Program

Fungsi insertionSort: Mengurutkan array menggunakan metode Insertion Sort. Fungsi isConstantDifference: Memeriksa apakah selisih antara elemen-elemen dalam array konstan. Fungsi main: Mengelola input pengguna, memanggil fungsi pengurutan dan pemeriksaan, serta menampilkan hasil.

III. UNGUIDED

Unguided 1

Sourcecode

```
package main
import "fmt"

//Rasyid Nafsyarie 2311102011
func selectionSort(arr []int, asc bool) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        idx := i
        for j := i + 1; j < n; j++ {
            if (asc && arr[j] < arr[idx]) || (!asc &&
arr[j] > arr[idx]) {
                idx = j
            }
        }
        arr[i], arr[idx] = arr[idx], arr[i]
    }
}

func printSeparator() {
    fmt.Println("\n=====
=====")
}

func printDashedSeparator() {
    fmt.Println("-----
-----")
}

func processDaerah(i, m int) ([]int, []int) {
    var arr = make([]int, m)
    fmt.Printf("Masukkan %d nomor rumah kerabat untuk
daerah ke-%d: ", m, i+1)
    for j := 0; j < m; j++ {
        fmt.Scan(&arr[j])
    }

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

func printSortedResults(i int, odd, even []int) {
```

```

        selectionSort(odd, true)
        selectionSort(even, false)

        fmt.Printf("\nNomor rumah terurut untuk daerah ke-
%d:\n", i+1)

        for _, num := range odd {
            fmt.Printf("%d ", num)
        }

        for _, num := range even {
            fmt.Printf("%d ", num)
        }
        fmt.Println()
    }

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

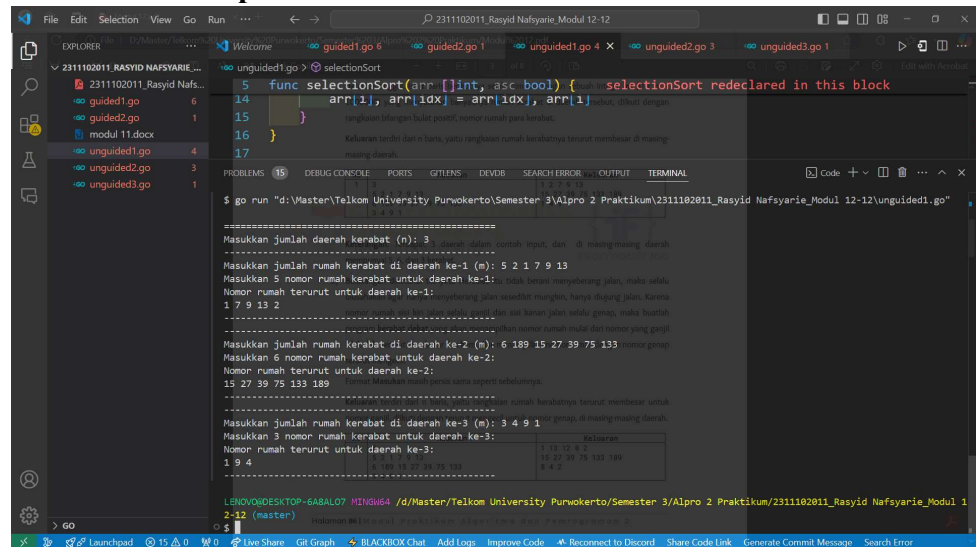
    for i := 0; i < n; i++ {
        var m int
        printDashedSeparator()
        fmt.Printf("Masukkan jumlah rumah kerabat di daerah
ke-%d (m): ", i+1)
        fmt.Scan(&m)

        odd, even := processDaerah(i, m)

        printSortedResults(i, odd, even)
        printDashedSeparator()
    }
}

```

Screenshoot Output



The screenshot shows a Go IDE with a file explorer on the left containing files like `guided1.go`, `guided2.go`, `modul 11.docx`, `unguided1.go`, `unguided2.go`, and `unguided3.go`. The main editor displays a Go program implementing a selection sort function. The function signature is `func selectionSort(arr []int, asc bool) {`. The code includes comments in Indonesian explaining the sorting process. The terminal output shows the program being run with the command `$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Alpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 12-12\unguided1.go"`. The output displays the input array `5 2 1 7 9 13` and the sorted array `1 2 5 7 9 13`. It also shows the process of selecting the minimum element and swapping it with the first element of the unsorted subarray.

Deskripsi Program

Fungsi `selectionSort`: Mengurutkan array berdasarkan urutan yang ditentukan (ascending atau descending). Fungsi `processDaerah`: Mengumpulkan input nomor rumah dari pengguna dan memisahnya menjadi dua kategori: ganjil dan genap. Fungsi `printSortedResults`: Mencetak hasil pengurutan nomor rumah. Fungsi `main`: Titik masuk program yang mengatur alur eksekusi.

Unguided 2

Sourcecode

```
package main
import (
    "bufio"
    "fmt"
    "os"
    "strconv"
    "strings"
)

//Rasyid Nafsyarie 2311102011
func selectionSort(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 calculateMedian(arr []int) float64 {
    n := len(arr)
    if n%2 == 0 {
        return float64(arr[n/2-1]+arr[n/2]) / 2.0
    }
    return float64(arr[n/2])
}

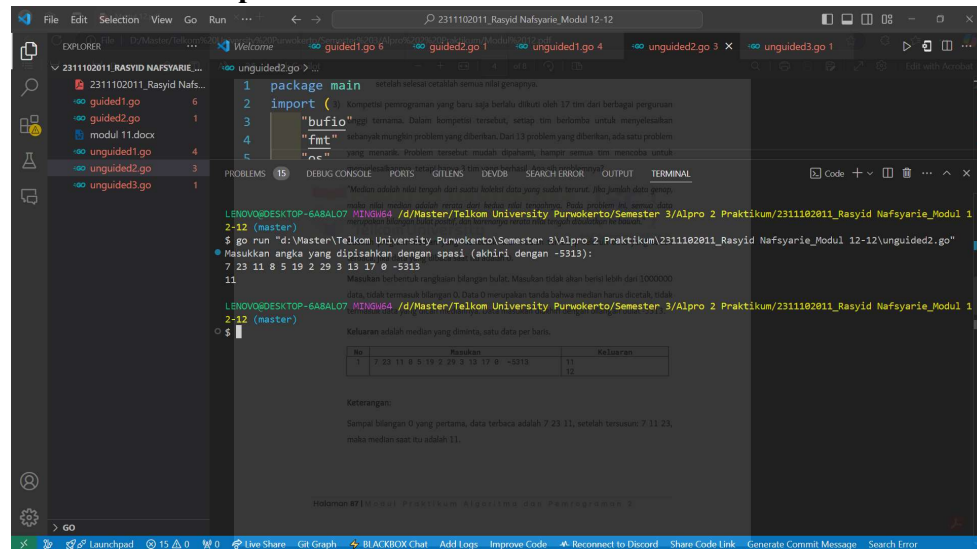
func main() {
    scanner := bufio.NewScanner(os.Stdin)
    fmt.Println("Masukkan angka yang dipisahkan dengan
spasi (akhiri dengan -5313):")
    scanner.Scan()
    line := scanner.Text()
    parts := strings.Split(line, " ")
    var data []int

    for _, part := range parts {
        num, err := strconv.Atoi(part)
        if err != nil {
            fmt.Println("Input tidak valid, harap masukkan
angka bulat saja.")
            return
        }

        if num == -5313 {
            break
        } else if num == 0 {
            selectionSort(data)
            median := calculateMedian(data)
            fmt.Printf("%.0f\n", median)
        } else {
            data = append(data, num)
        }
    }
}

```

Screenshoot Output



Deskripsi Program

Fungsi selectionSort: Mengurutkan array menggunakan algoritma Selection Sort. Fungsi calculateMedian: Menghitung median dari array yang telah diurutkan. Fungsi main: Mengelola input pengguna dan memanggil fungsi pengurutan dan perhitungan median.

Unguided 3

Sourcecode

```
package main
import (
    "fmt"
)
//Rasyid Nafsyarie 2311102011
type Buku struct {
    id        int
    judul     string
    penulis   string
    penerbit  string
    eksemplar int
    tahun     int
    rating    int
}

func DaftarkanBuku(pustaka *[]Buku, n int) {
    for i := 0; i < n; i++ {
        var buku Buku
```

```

        fmt.Printf("Masukkan data untuk buku ke-%d (id,
judul, penulis, penerbit, eksemplar, tahun, rating):\n",
i+1)
        fmt.Scan(&buku.id, &buku.judul, &buku.penulis,
&buku.penerbit, &buku.eksemplar, &buku.tahun,
&buku.rating)
        *pustaka = append(*pustaka, buku)
    }
}

func CetakFavorit(pustaka []Buku, n int) {
    if len(pustaka) == 0 {
        fmt.Println("Pustaka kosong.")
        return
    }
    terfavorit := pustaka[0]
    for _, buku := range pustaka {
        if buku.rating > terfavorit.rating {
            terfavorit = buku
        }
    }
    fmt.Println("Buku dengan rating tertinggi:")
    fmt.Printf("ID: %d, Judul: %s, Penulis: %s, Penerbit:
%s, Eksemplar: %d, Tahun: %d, Rating: %d\n",
                terfavorit.id, terfavorit.judul,
terfavorit.penulis, terfavorit.penerbit,
terfavorit.eksemplar, terfavorit.tahun,
terfavorit.rating)
}

func UrutkanBuku(pustaka *[]Buku, n int) {
    for i := 1; i < len(*pustaka); 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 Cetak5Terbaik(pustaka []Buku, n int) {
    fmt.Println("Lima buku dengan rating tertinggi:")
    for i := 0; i < 5 && i < len(pustaka); i++ {
        buku := pustaka[i]
        fmt.Printf("ID: %d, Judul: %s, Penulis: %s,
Penerbit: %s, Eksemplar: %d, Tahun: %d, Rating: %d\n",
                buku.id, buku.judul, buku.penulis,
buku.penerbit, buku.eksemplar, buku.tahun, buku.rating)
    }
}

```

```

func CariBuku(pustaka []Buku, n int, r int) {
    ditemukan := false
    for _, buku := range pustaka {
        if buku.rating == r {
            ditemukan = true
            fmt.Printf("ID: %d, Judul: %s, Penulis: %s,
Penerbit: %s, Eksemplar: %d, Tahun: %d, Rating: %d\n",
                buku.id, buku.judul, buku.penulis,
buku.penerbit, buku.eksemplar, buku.tahun, buku.rating)
        }
    }
    if !ditemukan {
        fmt.Println("Tidak ada buku dengan rating
tersebut.")
    }
}

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

    if n <= 0 || n > 7919 {
        fmt.Println("Jumlah buku harus antara 1 hingga
7919.")
        return
    }

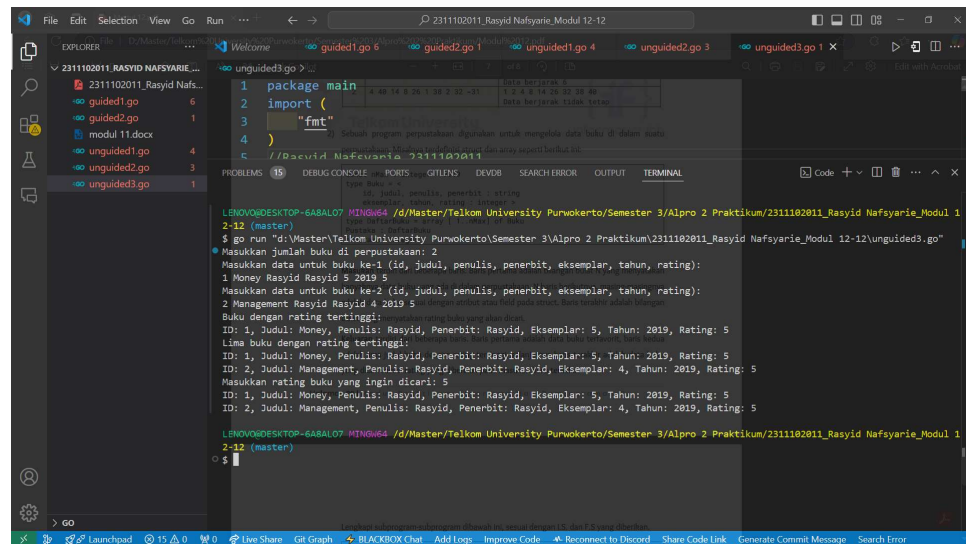
    var pustaka []Buku

    DaftarkanBuku(&pustaka, n)
    CetakFavorit(pustaka, n)
    UrutkanBuku(&pustaka, n)

    Cetak5Terbaik(pustaka, n)
    var rating int
    fmt.Print("Masukkan rating buku yang ingin dicari: ")
    fmt.Scan(&rating)
    CariBuku(pustaka, n, rating)
}

```

Screenshoot Output



```
1 package main
2 import (
3     "fmt"
4 )
5
6 // David Nafevanta 2311102011
7
8 type Buku struct {
9     ID int
10    Judul string
11    Penulis string
12    Penerbit string
13    Eksemplar int
14    Tahun int
15    Rating float32
16 }
17
18 func main() {
19     fmt.Println("Selamat datang di program manajemen buku.")
20     fmt.Println("Masukkan jumlah buku yang akan ditambahkan:")
21     var n int
22     for n = 0; n < 10; n++ {
23         fmt.Println("Masukkan data untuk buku ke-", n+1, "(id, judul, penulis, penerbit, eksemplar, tahun, rating):")
24         b := Buku{}
25         b.ID = n + 1
26         b.Judul = "Money Rasyid"
27         b.Penulis = "Rasyid"
28         b.Penerbit = "Rasyid"
29         b.Eksemplar = 5
30         b.Tahun = 2019
31         b.Rating = 5
32         books = append(books, b)
33     }
34     fmt.Println("Masukkan data untuk buku ke-2 (id, judul, penulis, penerbit, eksemplar, tahun, rating):")
35     b := Buku{}
36     b.ID = 2
37     b.Judul = "Management Rasyid"
38     b.Penulis = "Rasyid"
39     b.Penerbit = "Rasyid"
40     b.Eksemplar = 4
41     b.Tahun = 2019
42     b.Rating = 5
43     books = append(books, b)
44     fmt.Println("Buku dengan rating tertinggi:")
45     for i, b := range books {
46         fmt.Println("ID: ", b.ID, "Judul: ", b.Judul, "Penulis: ", b.Penulis, "Penerbit: ", b.Penerbit, "Eksemplar: ", b.Eksemplar, "Tahun: ", b.Tahun, "Rating: ", b.Rating)
47     }
48     fmt.Println("Lima buku dengan rating tertinggi:")
49     for i, b := range books {
50         fmt.Println("ID: ", b.ID, "Judul: ", b.Judul, "Penulis: ", b.Penulis, "Penerbit: ", b.Penerbit, "Eksemplar: ", b.Eksemplar, "Tahun: ", b.Tahun, "Rating: ", b.Rating)
51     }
52     fmt.Println("Masukkan rating buku yang ingin dicari:")
53     var r float32
54     for r = 0; r < 10; r++ {
55         fmt.Println("ID: ", b.ID, "Judul: ", b.Judul, "Penulis: ", b.Penulis, "Penerbit: ", b.Penerbit, "Eksemplar: ", b.Eksemplar, "Tahun: ", b.Tahun, "Rating: ", b.Rating)
56     }
57     fmt.Println("ID: ", b.ID, "Judul: ", b.Judul, "Penulis: ", b.Penulis, "Penerbit: ", b.Penerbit, "Eksemplar: ", b.Eksemplar, "Tahun: ", b.Tahun, "Rating: ", b.Rating)
58 }
59
```

Deskripsi Program

Strukt Buku: Menyimpan atribut buku seperti ID, judul, penulis, penerbit, jumlah eksemplar, tahun terbit, dan rating. Fungsi DaftarkanBuku: Mengumpulkan data buku dari pengguna dan menambahkannya ke dalam slice pustaka. Fungsi CetakFavorit: Mencetak buku dengan rating tertinggi. Fungsi UrutkanBuku: Mengurutkan buku berdasarkan rating secara descending. Fungsi Cetak5Terbaik: Mencetak lima buku dengan rating tertinggi. Fungsi CariBuku: Mencari dan mencetak buku berdasarkan rating yang diminta pengguna.