

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 12 & 13
PENGURUTAN DATA**



Disusun Oleh :

Aby Hakim Al Yasiry Faozi/2311102208

IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

A. Selection Sort

Selection sort adalah algoritma pengurutan yang bekerja dengan membagi daftar data menjadi dua bagian: bagian yang sudah terurut dan bagian yang belum terurut. Algoritma ini berulang kali memilih elemen terkecil (atau terbesar) dari bagian yang belum terurut dan menukarnya dengan elemen pertama dari bagian yang belum terurut, lalu memindahkan batas antara kedua bagian ini.

Langkah-langkah:

- **Inisialisasi:** Tentukan elemen pertama dari daftar sebagai elemen minimum (atau maksimum) saat ini.
- **Pencarian Minimum:** Bandingkan elemen minimum saat ini dengan elemen-elemen berikutnya dalam daftar untuk menemukan elemen terkecil dalam bagian yang belum terurut.
- **Penukaran:** Setelah menemukan elemen terkecil, tukar elemen tersebut dengan elemen pertama dari bagian yang belum terurut.
- **Penggeseran Batas:** Pindahkan batas antara bagian yang terurut dan belum terurut satu elemen ke kanan.
- **Pengulangan:** Ulangi langkah-langkah di atas untuk bagian yang belum terurut hingga seluruh daftar terurut.

B. Insertion Sort

Konsep: Insertion sort adalah algoritma pengurutan yang bekerja dengan membangun daftar terurut satu elemen pada satu waktu. Algoritma ini secara berulang mengambil elemen dari bagian yang belum terurut dan memasukkannya ke posisi yang tepat dalam bagian yang sudah terurut, sehingga bagian yang sudah terurut selalu tetap terurut.

Langkah-langkah:

- **Inisialisasi:** Mulai dengan elemen kedua dalam daftar (diasumsikan elemen pertama sudah terurut).

- Pencarian Posisi: Bandingkan elemen saat ini dengan elemen-elemen sebelumnya dalam bagian yang sudah terurut untuk menemukan posisi yang tepat.
- Penggeseran: Geser semua elemen dalam bagian yang sudah terurut yang lebih besar dari elemen saat ini ke satu posisi ke kanan.
- Penyisipan: Masukkan elemen saat ini ke dalam posisi yang tepat dalam bagian yang sudah terurut.
- Pengulangan: Pindahkan batas antara bagian yang sudah terurut dan bagian yang belum terurut satu elemen ke kanan, dan ulangi proses untuk elemen-elemen yang belum terurut.

II. GUIDED

1. Soal Studi Case

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

```

akim\Kuliah\Semester 3\prak alpro 2> go run "c:\Aby Hakim\K
uliah\Semester 3\prak alpro 2\modul12\guided1\guided1.go"
Masukkan jumlah daerah kerabat (n): 2

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 2
Masukkan 2 nomor rumah kerabat: 11 22
Nomor rumah terurut untuk daerah 1: 11 22

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 3
Masukkan 3 nomor rumah kerabat: 44 55 66
Nomor rumah terurut untuk daerah 2: 44 55 66
PS C:\Aby Hakim\Kuliah\Semester 3\prak alpro 2>

```

Deskripsi Program

1. Inisialisasi:

- Program dimulai dengan meminta pengguna memasukkan jumlah daerah kerabat yang akan diolah.

2. Proses Per Daerah:

- Untuk setiap daerah, program meminta pengguna memasukkan jumlah nomor rumah kerabat di daerah tersebut.
- Selanjutnya, program meminta pengguna memasukkan nomor rumah kerabat untuk daerah tersebut.

3. Pengurutan Menggunakan Selection Sort:

- Nomor-nomor rumah yang telah dimasukkan akan diurutkan menggunakan algoritma Selection Sort. Proses pengurutan ini dilakukan dengan cara:
 - Memilih elemen terkecil dari daftar yang belum terurut.
 - Menukar elemen terkecil tersebut dengan elemen pertama yang belum terurut.
 - Mengulangi proses ini untuk elemen-elemen berikutnya hingga seluruh daftar terurut.

4. Menampilkan Hasil:

- Setelah nomor rumah diurutkan, program akan menampilkan nomor rumah yang sudah terurut untuk setiap daerah.

2. Soal Studi Case

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)

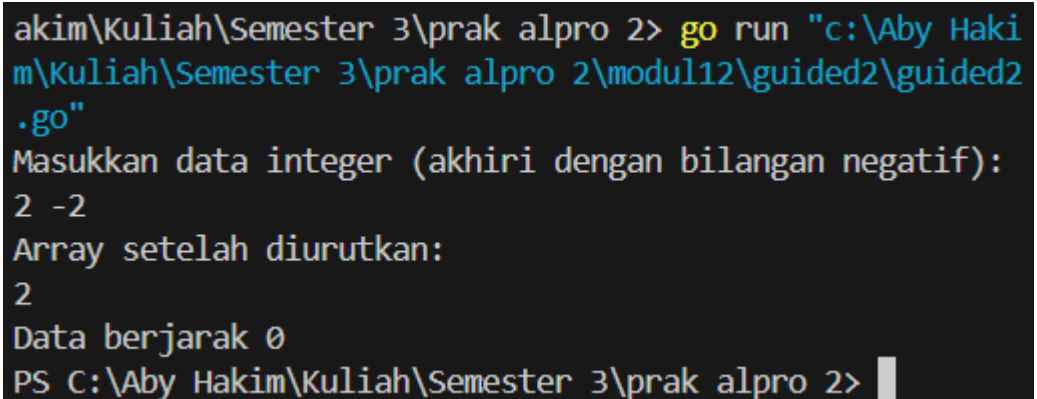
    // Periksa apakah selisih elemen tetap
    isConstant, difference := isConstantDifference(arr,
n)

    // 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



```
akim\Kuliah\Semester 3\prak alpro 2> go run "c:\Aby Hakim\Kuliah\Semester 3\prak alpro 2\modul12\guided2\guided2.go"
Masukkan data integer (akhiri dengan bilangan negatif):
2 -2
Array setelah diurutkan:
2
Data berjarak 0
PS C:\Aby Hakim\Kuliah\Semester 3\prak alpro 2> █
```

Deskripsi Program

1. Input Data:

- Program meminta pengguna untuk memasukkan sejumlah bilangan bulat satu per satu. Pengguna dapat terus memasukkan bilangan hingga memasukkan bilangan negatif yang menandakan akhir dari input data.

2. Pengurutan Menggunakan Insertion Sort:

- Setelah semua data dimasukkan, program mengurutkan bilangan-bilangan tersebut menggunakan algoritma Insertion Sort. Algoritma ini bekerja dengan cara:
 - Memulai dengan elemen kedua dalam daftar dan membandingkannya dengan elemen-elemen sebelumnya.
 - Memindahkan elemen-elemen yang lebih besar ke kanan untuk memberi ruang bagi elemen yang sedang diurutkan.
 - Menempatkan elemen tersebut ke posisi yang tepat dalam bagian yang sudah terurut.
- Proses ini diulang hingga seluruh array terurut.

3. Pemeriksaan Selisih Konstan:

- Setelah array diurutkan, program memeriksa apakah selisih antara setiap pasangan elemen berturut-turut tetap konstan.
- Jika selisih antara setiap pasangan elemen sama, program mencatat nilai selisih tersebut.
- Jika ada pasangan elemen yang selisihnya berbeda, program mencatat bahwa selisihnya tidak tetap.

III. UNGUIDED

1. Soal Studi Case

Sourcecode

```
package main

import (
    "fmt"
)

func ganjilGenap(arr []int) (ganjil []int, genap []int) {
    for _, nomor := range arr {
        if nomor%2 == 0 {
            genap = append(genap, nomor)
        } else {
            ganjil = append(ganjil, nomor)
        }
    }
    return
}

func selectionSort(arr []int, naik bool) {
    panjang := len(arr)
    for i := 0; i < panjang-1; i++ {
        posisi := i
        for j := i + 1; j < panjang; j++ {
            if (naik && arr[j] < arr[posisi]) || (!naik && arr[j] > arr[posisi]) {
                posisi = j
            }
        }
        arr[i], arr[posisi] = arr[posisi], arr[i]
    }
}

func main() {
    var jumlahDaerah int
    fmt.Print("Masukkan jumlah daerah: ")
}
```



```
        fmt.Scan(&jumlahDaerah)

        for i := 0; i < jumlahDaerah; i++ {
            fmt.Printf("\nMasukkan jumlah rumah di daerah ke-%d: ", i+1)
            var jumlahRumah int
            fmt.Scan(&jumlahRumah)

            nomorRumah := make([]int, jumlahRumah)
            fmt.Printf("Masukkan nomor rumah di daerah ke-%d:\n", i+1)
            for j := 0; j < jumlahRumah; j++ {
                fmt.Scan(&nomorRumah[j])
            }

            ganjil, genap := ganjilGenap(nomorRumah)

            selectionSort(ganjil, false)
            selectionSort(genap, true)

            fmt.Printf("\nHasil untuk daerah ke-%d:\n", i+1)
            for _, nomor := range ganjil {
                fmt.Printf("%d ", nomor)
            }
            for _, nomor := range genap {
                fmt.Printf("%d ", nomor)
            }
            fmt.Println()
        }
    }
```

Screenshoot Output

```
akim\Kuliah\Semester 3\prak alpro 2> go run "c:\Aby Hakim\Kuliah\Semester 3\prak alpro 2\modul12\unguided1\unguided1.go"
Masukkan jumlah daerah: 2

Masukkan jumlah rumah di daerah ke-1: 6
Masukkan nomor rumah di daerah ke-1:
1 5 7 2 9 19

Hasil untuk daerah ke-1:
19 9 7 5 1 2

Masukkan jumlah rumah di daerah ke-2: 5
Masukkan nomor rumah di daerah ke-2:
11 56 73 44 99

Hasil untuk daerah ke-2:
99 73 11 44 56
PS C:\Aby Hakim\Kuliah\Semester 3\prak alpro 2> |
```

Deskripsi Program

1. Inisialisasi:
 - Program dimulai dengan meminta pengguna memasukkan jumlah daerah yang akan diolah.
2. Input Data Per Daerah:
 - Untuk setiap daerah, program meminta pengguna memasukkan jumlah rumah di daerah tersebut.
 - Pengguna kemudian memasukkan nomor-nomor rumah di daerah tersebut.
3. Pisahkan Ganjil dan Genap:
 - Program memisahkan nomor-nomor rumah ke dalam dua kategori: ganjil dan genap. Bilangan ganjil dimasukkan ke dalam daftar ganjil, sedangkan bilangan genap dimasukkan ke dalam daftar genap.
4. Pengurutan Menggunakan Selection Sort:
 - Program mengurutkan daftar bilangan ganjil dan genap secara terpisah menggunakan algoritma Selection Sort.

- Untuk bilangan ganjil, diurutkan secara menurun (descending).
- Untuk bilangan genap, diurutkan secara menaik (ascending).

2. Soal Studi Case

Sourcecode

```
package main

import "fmt"

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 median(arr []int) int {
    n := len(arr)
    if n%2 == 1 {
        return arr[n/2]
    }
    return (arr[n/2-1] + arr[n/2]) / 2
}

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

    fmt.Println("Masukkan data:")

    for {
        fmt.Scan(&input)
        if input == -531351 {
            break
        }

        if input == 0 {
            if len(data) > 0 {
                temp := make([]int, len(data))
                copy(temp, data)
                selectionSort(temp)
            }
        }
    }
}
```

```

                                fmt.Println("Median saat ini:",
median(temp))
                                } else {
                                    fmt.Println("Tidak ada data untuk
dihitung.")
                                }
                                } else {
                                    data = append(data, input)
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

```

Screenshoot Output

```

akim\Kuliah\Semester 3\prak alpro 2> go run "c:\Aby Haki
m\Kuliah\Semester 3\prak alpro 2\modul12\unguided2\ungui
ded2.go"
Masukkan data:
7 23 11 0 5 19 2 29 3 13 17 0 -5313
Median saat ini: 11
Median saat ini: 12

```

Deskripsi Program

1. Input Data:

- Program dimulai dengan meminta pengguna memasukkan sejumlah bilangan bulat satu per satu. Pengguna terus memasukkan bilangan hingga memasukkan bilangan 0 (untuk menghitung median) atau bilangan -531351 (untuk mengakhiri program).

2. Pengurutan Menggunakan Selection Sort:

- Setiap kali pengguna memasukkan bilangan 0, program mengurutkan angka-angka yang telah dimasukkan menggunakan algoritma Selection Sort. Algoritma ini bekerja dengan cara memilih elemen terkecil dari bagian yang belum terurut dan menukarnya dengan elemen pertama dari bagian yang belum terurut. Proses ini diulang hingga seluruh daftar terurut.

3. Menghitung Median:

- Setelah array diurutkan, program menghitung median dari angka-angka tersebut.
 - Jika jumlah elemen dalam array ganjil, median adalah elemen di tengah.

- Jika jumlah elemen dalam array genap, median adalah rata-rata dari dua elemen di tengah.
- Median yang dihitung ditampilkan kepada pengguna.

4. Menampilkan Hasil:

- Program menampilkan "Tidak ada data untuk dihitung" jika tidak ada angka yang dimasukkan ketika pengguna mencoba menghitung median.
- Jika ada angka yang dimasukkan, program menampilkan median dari angka-angka tersebut.

3. Soal Studi Case

Sourcecode

```
package main

import (
    "fmt"
)

const nMax = 7919

type Buku struct {
    Judul, Penulis, Penerbit      string
    Eksemplar, Tahun, Rating, ID int
}

type DaftarBuku struct {
    Data      [nMax]Buku
    nPustaka int
}

func DaftarkanBuku(pustaka *DaftarBuku, n int) {
    pustaka.nPustaka = n
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan data buku ke-%d (ID, Judul, Penulis, Penerbit, Eksemplar, Tahun, Rating):\n", i+1)
        fmt.Scan(&pustaka.Data[i].ID,
            &pustaka.Data[i].Judul, &pustaka.Data[i].Penulis,
            &pustaka.Data[i].Penerbit,
            &pustaka.Data[i].Eksemplar,
            &pustaka.Data[i].Tahun, &pustaka.Data[i].Rating)
    }
}

func CetakTerfavorit(pustaka DaftarBuku) {
    if pustaka.nPustaka == 0 {
        fmt.Println("Tidak ada buku dalam daftar.")
        return
    }
}
```

```

        favorit := pustaka.Data[0]
        for i := 1; i < pustaka.nPustaka; i++ {
            if pustaka.Data[i].Rating > favorit.Rating {
                favorit = pustaka.Data[i]
            }
        }

        fmt.Println("Buku Terfavorit:")
        fmt.Printf("ID: %d, Judul: %s, Penulis: %s,
Penerbit: %s, Tahun: %d\n",
            favorit.ID, favorit.Judul, favorit.Penulis,
            favorit.Penerbit, favorit.Tahun)
    }

    func UrutBuku(pustaka *DaftarBuku) {
        for i := 0; i < pustaka.nPustaka-1; i++ {
            for j := 0; j < pustaka.nPustaka-i-1; j++ {
                if pustaka.Data[j].Rating <
pustaka.Data[j+1].Rating {
                    temp := pustaka.Data[j]
                    pustaka.Data[j] = pustaka.Data[j+1]
                    pustaka.Data[j+1] = temp
                }
            }
        }
    }

    func Cetak5Buku(pustaka DaftarBuku) {
        fmt.Println("5 Buku Dengan Rating Tertinggi:")
        for i := 0; i < pustaka.nPustaka && i < 5; i++ {
            fmt.Printf("ID: %d, Judul: %s, Penulis: %s,
Penerbit: %s, Rating: %d\n",
                pustaka.Data[i].ID, pustaka.Data[i].Judul,
                pustaka.Data[i].Penulis,
                pustaka.Data[i].Penerbit,
                pustaka.Data[i].Rating)
        }
    }

    func CariBuku(pustaka DaftarBuku, target int) {
        found := false
        for i := 0; i < pustaka.nPustaka; i++ {
            if pustaka.Data[i].Rating == target {
                fmt.Println("Buku ditemukan:")
                fmt.Printf("ID: %d, Judul: %s, Penulis: %s,
Penerbit: %s, Tahun: %d, Rating: %d\n",
                    pustaka.Data[i].ID,
                    pustaka.Data[i].Judul, pustaka.Data[i].Penulis,
                    pustaka.Data[i].Penerbit,
                    pustaka.Data[i].Tahun, pustaka.Data[i].Rating)
                found = true
            }
        }
    }

```

```

    }
}
if !found {
    fmt.Println("Tidak ada buku dengan rating
tersebut.")
}
}

func main() {
    var pustaka DaftarBuku
    var jumlahBuku, cariRating int

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

    DaftarkanBuku(&pustaka, jumlahBuku)

    fmt.Println("\nBuku Terfavorit")
    CetakTerfavorit(pustaka)

    UrutBuku(&pustaka)

    fmt.Println("\n5 Buku Dengan Rating Tertinggi")
    Cetak5Buku(pustaka)

    fmt.Print("\nMasukkan rating buku yang ingin dicari:
")
    fmt.Scan(&cariRating)

    CariBuku(pustaka, cariRating)
}

```

Screenshoot Output

```
akim\Kuliah\Semester 3\prak alpro 2> go run "c:\Aby Hakim\Kuliah\Semester 3\prak alpro 2\modul12\unguided3\unguided3.go"
Masukkan jumlah buku: 2
Masukkan data buku ke-1 (ID, Judul, Penulis, Penerbit, Eksemplar, Tahun, Rating):
208 air aby erlangga 500 2024 10
Masukkan data buku ke-2 (ID, Judul, Penulis, Penerbit, Eksemplar, Tahun, Rating):
109 api hakim erlangga 600 2022 9

Buku Terfavorit
Buku Terfavorit:
ID: 208, Judul: air, Penulis: aby, Penerbit: erlangga, Tahun: 2024

5 Buku Dengan Rating Tertinggi
5 Buku Dengan Rating Tertinggi:
ID: 208, Judul: air, Penulis: aby, Penerbit: erlangga, Rating: 10
ID: 109, Judul: api, Penulis: hakim, Penerbit: erlangga, Rating: 9

Masukkan rating buku yang ingin dicari: 10
Buku ditemukan:
ID: 208, Judul: air, Penulis: aby, Penerbit: erlangga, Tahun: 2024, Rating: 10
PS C:\Aby Hakim\Kuliah\Semester 3\prak alpro 2> |
```

Deskripsi Program

1. Definisi Struktur Data:

- Buku: Struktur data untuk menyimpan informasi buku, termasuk ID, judul, penulis, penerbit, jumlah eksemplar, tahun penerbitan, dan rating.
- DaftarBuku: Struktur data untuk menyimpan koleksi buku. Ini termasuk array dari buku dan jumlah buku dalam koleksi.

2. Pendaftaran Buku:

- Fungsi `DaftarkanBuku` digunakan untuk memasukkan data buku ke dalam koleksi. Pengguna diminta untuk memasukkan data buku seperti ID, judul, penulis, penerbit, jumlah eksemplar, tahun penerbitan, dan rating untuk setiap buku.
3. Pencetakan Buku Terfavorit:
 - Fungsi `CetakTerfavorit` digunakan untuk mencari dan mencetak buku dengan rating tertinggi dalam koleksi. Jika tidak ada buku dalam koleksi, fungsi ini akan menampilkan pesan bahwa tidak ada buku dalam daftar.
 4. Pengurutan Buku:
 - Fungsi `UrutBuku` mengurutkan buku-buku dalam koleksi berdasarkan rating dari yang tertinggi ke terendah menggunakan algoritma pengurutan bubble sort.
 5. Pencetakan 5 Buku Dengan Rating Tertinggi:
 - Fungsi `Cetak5Buku` mencetak informasi 5 buku dengan rating tertinggi dalam koleksi.
 6. Pencarian Buku Berdasarkan Rating:
 - Fungsi `CariBuku` digunakan untuk mencari dan mencetak buku berdasarkan rating yang diberikan oleh pengguna. Jika tidak ada buku dengan rating tersebut, fungsi ini akan menampilkan pesan bahwa tidak ada buku dengan rating tersebut.
 7. Fungsi main:
 - Fungsi utama dari program ini mengelola alur dari proses pendaftaran buku, pencetakan buku terfavorit, pengurutan buku, pencetakan 5 buku dengan rating tertinggi, dan pencarian buku berdasarkan rating.
 - Program dimulai dengan meminta pengguna untuk memasukkan jumlah buku yang akan didaftarkan.
 - Buku-buku tersebut kemudian didaftarkan dan berbagai operasi seperti pencetakan buku terfavorit, pengurutan buku, dan pencetakan buku dengan rating tertinggi dilakukan.
 - Terakhir, pengguna dapat mencari buku berdasarkan rating yang mereka inginkan.