

**LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2
MODUL 12 & 13
PENGURUTAN DATA**



Oleh:

MUHAMMAD AGHA ZULFADHLI

2311102015

S1-IF11-02

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

I. DASAR TEORI

Pengurutan secara seleksi ini idenya adalah mencari nilai ekstrem 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 tersisa 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 ekstrem 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	}

Pengurutan secara insertion ini idenya adalah menyisipkan suatu nilai pada posisi yang seharusnya. Berbeda dengan pengurutan seleksi, yang mana pada pengurutan ini tidak dilakukan pencarian nilai ekstrem terlebih dahulu, cukup memilih suatu nilai tertentu kemudian mencari posisinya secara sequential search. Pada penjelasan berikut ini data akan diurut mengecil (descending), dan data dengan indeks kecil ada di "kiri" dan indeks besar ada di "kanan".

- 1) Untuk satu data yang belum terurut dan sejumlah data yang sudah diurutkan:
Geser data yang sudah terurut tersebut (ke kanan), sehingga ada satu ruang kosong untuk memasukkan data yang belum terurut ke dalam data yang sudah terurut dan tetap menjaga keterurutan.

- 2) ulangi proses tersebut untuk setiap data yang belum terurut terhadap rangkaian data yang sudah terurut.

Algoritma ini dikenal juga dengan nama Insertion Sort, yang mana pada algoritma ini melibatkan dua proses yaitu pencarian sekuensial dan penyisipan.

	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	$j \leftarrow i$	$j = i$
4	$temp \leftarrow a[j]$	$temp = a[j]$
5	while $j > 0$ and $temp > a[j-1]$ do	for $j > 0 \ \&\& \ temp > a[j-1]$ {
6	$a[j] \leftarrow a[j-1]$	$a[j] = a[j-1]$
7	$j \leftarrow j - 1$	$j = j - 1$
8	endwhile	}
9	$a[j] \leftarrow temp$	$a[j] = temp$
10	$i \leftarrow i + 1$	$i = i + 1$
11	endwhile	}

II. GUIDED

1. Guided 1

Source code

```
package main

import "fmt"

// Fungsi untuk mengurutkan array menggunakan selection sort
func selectionSort(arr []int) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        maxIdx := i
        for j := i + 1; j < n; j++ {
            if arr[j] < arr[maxIdx] { // Cari elemen terbesar
                maxIdx = j
            }
        }
        arr[i], arr[maxIdx] = arr[maxIdx], arr[i] // Tukar elemen
    }
}

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

    if n <= 0 || n >= 1000 {
        fmt.Println("n harus lebih besar dari 0 dan kurang dari 1000.")
        return
    }

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

        if m <= 0 || m >= 1000000 {
```

```

        fmt.Println("m harus lebih besar dari 0 dan kurang
dari 1000000.")
        return
    }

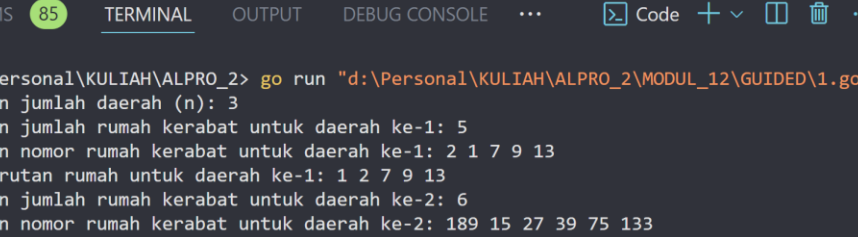
    // Masukkan nomor rumah
    houses := make([]int, m)
    fmt.Printf("Masukkan nomor rumah kerabat untuk daerah
ke-%d: ", i+1)
    for j := 0; j < m; j++ {
        fmt.Scan(&houses[j])
    }

    // Urutkan dengan selection sort
    selectionSort(houses)

    // Cetak hasil
    fmt.Printf("Hasil urutan rumah untuk daerah ke-%d: ",
i+1)
    for _, house := range houses {
        fmt.Printf("%d ", house)
    }
    fmt.Println()
}
}

```

Screenshoot program



```
1.go - ALPRO_2 - Visual Studio Code

PROBLEMS 85 TERMINAL OUTPUT DEBUG CONSOLE ... Code + - [ ] [ ] ... - X

PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_12\GUIDED\1.go"
Masukkan jumlah daerah (n): 3
Masukkan jumlah rumah kerabat untuk daerah ke-1: 5
Masukkan nomor rumah kerabat untuk daerah ke-1: 2 1 7 9 13
Hasil urutan rumah untuk daerah ke-1: 1 2 7 9 13
Masukkan jumlah rumah kerabat untuk daerah ke-2: 6
Masukkan nomor rumah kerabat untuk daerah ke-2: 189 15 27 39 75 133
Hasil urutan rumah untuk daerah ke-2: 15 27 39 75 133 189
Masukkan jumlah rumah kerabat untuk daerah ke-3: 3
Masukkan nomor rumah kerabat untuk daerah ke-3: 4 9 1
Hasil urutan rumah untuk daerah ke-3: 1 4 9
PS D:\Personal\KULIAH\ALPRO_2>
```

Deskripsi program

Program di atas adalah aplikasi berbasis terminal untuk mengurutkan daftar nomor rumah kerabat di beberapa daerah. Pengguna diminta untuk memasukkan jumlah daerah n dengan batasan n harus lebih besar dari 0 dan kurang dari 1000. Untuk setiap daerah, pengguna kemudian memasukkan jumlah rumah m (antara 1 dan 999,999) dan daftar nomor rumah kerabat di daerah tersebut.

Program mengurutkan nomor rumah untuk setiap daerah menggunakan algoritma **selection sort** dalam urutan menaik (ascending). Setelah pengurutan, program mencetak daftar nomor rumah yang telah diurutkan untuk setiap daerah dengan format terstruktur. Validasi input dilakukan untuk memastikan jumlah daerah dan rumah berada dalam rentang yang diperbolehkan. Jika input tidak valid, program akan memberikan pesan kesalahan dan berhenti.

2. Guided 2

Source code

```
package main

import (
    "fmt"
    "math"
)

// Fungsi insertion sort untuk mengurutkan array
func insertionSort(arr []int) {
    n := len(arr)
    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 data berjarak tetap
func isDataConsistentlySpaced(arr []int) (bool, int) {
    if len(arr) < 2 {
        return true, 0 // Array dengan kurang dari 2 elemen
        dianggap berjarak tetap
    }

    // Hitung selisih awal
    diff := int(math.Abs(float64(arr[1] - arr[0])))

    for i := 1; i < len(arr)-1; i++ {
        currentDiff := int(math.Abs(float64(arr[i+1] - arr[i])))
        if currentDiff != diff {
```

```

        return false, 0 // Jika ada selisih yang berbeda,
        tidak berjarak tetap
    }
}

return true, diff
}

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

    fmt.Println("Masukkan data (akhiri dengan bilangan
negatif):")
    for {
        fmt.Scan(&input)
        if input < 0 {
            break
        }
        data = append(data, input)
    }

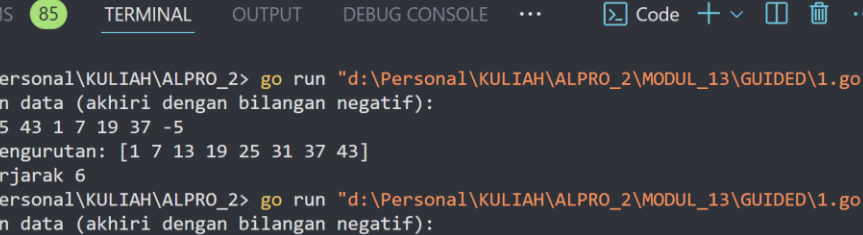
    // Urutkan data menggunakan insertion sort
    insertionSort(data)

    // Periksa apakah data berjarak tetap
    isConsistent, diff := isDataConsistentlySpaced(data)

    // Cetak hasil
    fmt.Println("Hasil pengurutan:", data)
    if isConsistent {
        fmt.Printf("Data berjarak %d\n", diff)
    } else {
        fmt.Println("Data berjarak tidak tetap")
    }
}

```


Screenshoot program



1.go - ALPRO_2 - Visual Studio Code

PROBLEMS 85 TERMINAL OUTPUT DEBUG CONSOLE ... Code + - [] [] ... - X

```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_13\GUIDED\1.go"
Masukkan data (akhiri dengan bilangan negatif):
31 13 25 43 1 7 19 37 -5
Hasil pengurutan: [1 7 13 19 25 31 37 43]
Data berjarak 6
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_13\GUIDED\1.go"
Masukkan data (akhiri dengan bilangan negatif):
4 40 14 8 26 1 38 2 32 -31
Hasil pengurutan: [1 2 4 8 14 26 32 38 40]
Data berjarak tidak tetap
PS D:\Personal\KULIAH\ALPRO_2>
```

Deskripsi program

Program di atas adalah aplikasi berbasis terminal yang memproses serangkaian data numerik untuk menentukan apakah data tersebut memiliki jarak tetap setelah diurutkan. Pengguna diminta untuk memasukkan angka secara berulang, dengan proses input berhenti ketika angka negatif dimasukkan. Data yang dimasukkan kemudian diurutkan menggunakan algoritma insertion sort, yang mengatur elemen dalam urutan menaik.

Setelah data diurutkan, program memeriksa apakah selisih antara elemen-elemen berturut-turut dalam array yang diurutkan adalah konstan (berjarak tetap). Jika semua selisih konsisten, program menampilkan bahwa data tersebut memiliki jarak tetap beserta nilai jarak tersebut. Jika selisih bervariasi, program menyatakan bahwa data tidak memiliki jarak tetap.

Program ini juga mengakomodasi kasus khusus seperti array dengan kurang dari dua elemen, yang dianggap berjarak tetap dengan jarak nol. Hasil pengurutan dan analisis ditampilkan ke layar untuk pengguna.

III. UNGUIDED

1. Latihan Soal No. 2 Modul 12

Source code

```
package main

import (
    "fmt"
    "slices"
)

// Fungsi untuk mengurutkan array menggunakan selection sort
func selectionSort(arr []int) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        maxIdx := i
        for j := i + 1; j < n; j++ {
            if arr[j] < arr[maxIdx] { // Cari elemen terbesar
                maxIdx = j
            }
        }
        arr[i], arr[maxIdx] = arr[maxIdx], arr[i] // Tukar elemen
    }
}

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

    if n <= 0 || n >= 1000 {
        fmt.Println("n harus lebih besar dari 0 dan kurang dari 1000.")
        return
    }

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

```

        fmt.Scan(&m)

        if m <= 0 || m >= 1000000 {
            fmt.Println("m harus lebih besar dari 0 dan kurang
dari 1000000.")
            return
        }

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

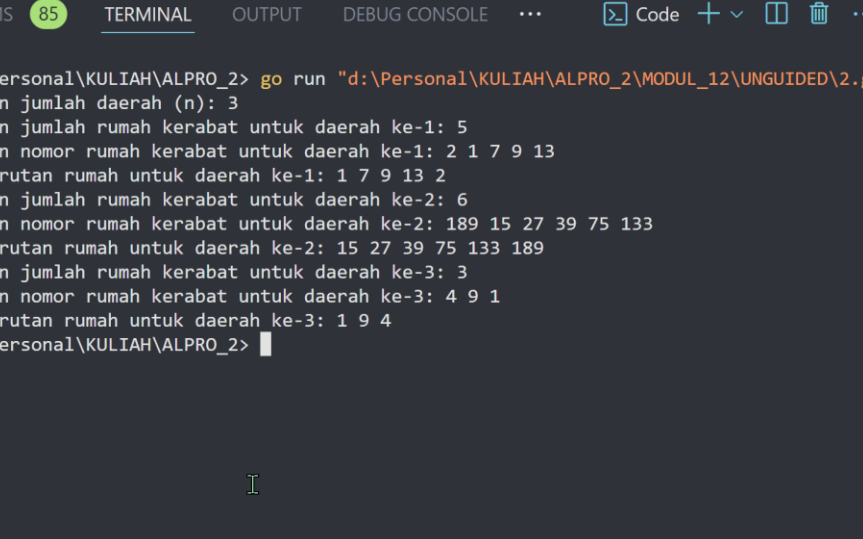
        var rmhGanjil, rmhGenap []int
        for _, v := range houses {
            if v%2 == 0 {
                rmhGenap = append(rmhGenap, v)
            }
            if v%2 == 1 {
                rmhGanjil = append(rmhGanjil, v)
            }
        }
        // Urutkan dengan selection sort
        selectionSort(rmhGanjil)
        selectionSort(rmhGenap)

        slices.Reverse(rmhGenap)
        rmhGanjil = append(rmhGanjil, rmhGenap...)

        // Cetak hasil
        fmt.Printf("Hasil urutan rumah untuk daerah ke-%d: ",
i+1)
        for _, house := range rmhGanjil {
            fmt.Printf("%d ", house)
        }
        fmt.Println()
    }
}

```

Screenshoot program



2.go - ALPRO_2 - Visual Studio Code

PROBLEMS 85 TERMINAL OUTPUT DEBUG CONSOLE ... Code + ▾ ▢ 🗑️ ... ▾ ✕

```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_12\UNGUIDED\2.go"
Masukkan jumlah daerah (n): 3
Masukkan jumlah rumah kerabat untuk daerah ke-1: 5
Masukkan nomor rumah kerabat untuk daerah ke-1: 2 1 7 9 13
Hasil urutan rumah untuk daerah ke-1: 1 7 9 13 2
Masukkan jumlah rumah kerabat untuk daerah ke-2: 6
Masukkan nomor rumah kerabat untuk daerah ke-2: 189 15 27 39 75 133
Hasil urutan rumah untuk daerah ke-2: 15 27 39 75 133 189
Masukkan jumlah rumah kerabat untuk daerah ke-3: 3
Masukkan nomor rumah kerabat untuk daerah ke-3: 4 9 1
Hasil urutan rumah untuk daerah ke-3: 1 9 4
PS D:\Personal\KULIAH\ALPRO_2>
```

Deskripsi program

. Program di atas adalah sebuah aplikasi berbasis terminal yang bertujuan untuk mengurutkan nomor rumah kerabat di berbagai daerah berdasarkan aturan tertentu. Pengguna diminta untuk memasukkan jumlah daerah n , dengan batasan nilai n harus lebih besar dari 0 dan kurang dari 1000. Untuk setiap daerah, pengguna kemudian memasukkan jumlah rumah m (harus antara 1 dan 999,999) dan daftar nomor rumah kerabat di daerah tersebut.

Program memproses daftar nomor rumah sebagai berikut:

1. Memisahkan nomor rumah menjadi dua kategori: nomor ganjil dan nomor genap.
2. Mengurutkan nomor ganjil dalam urutan menaik (ascending) menggunakan algoritma selection sort.
3. Mengurutkan nomor genap dalam urutan menurun (descending) dengan membalik hasil urutan ascending.
4. Menggabungkan nomor ganjil yang telah diurutkan dengan nomor genap yang sudah dibalik, sehingga nomor ganjil muncul terlebih dahulu.

Setelah proses di atas, program mencetak daftar nomor rumah yang telah diurutkan untuk setiap daerah sesuai format yang ditentukan. Program juga

memvalidasi input agar tetap berada dalam batasan yang diperbolehkan, dengan memberikan pesan kesalahan jika input tidak valid.

2. Latihan Soal No. 3 Modul 12

Source code

```
package main
package main

import (
    "fmt"
    "math"
    "strconv"
)

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

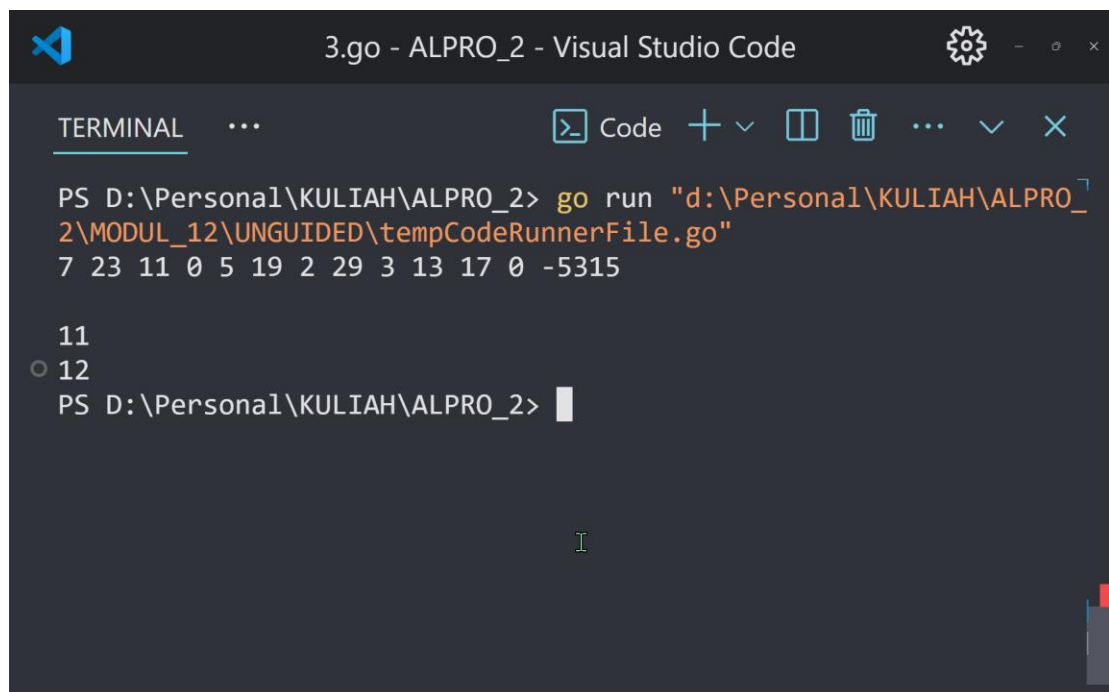
func main() {
    var x int
    var arr []int
    var text string
    for x >= 0 {
        fmt.Scan(&x)
        if x == 0 {
            selectionSort(arr)
            median := len(arr) / 2
            if len(arr)%median == 0 && median != 1 {
                text += "\n" +
                strconv.FormatFloat(math.Floor(float64(arr[median-
                1]+arr[median])/2), 'f', 0, 64)
```

```

    } else {
        text += "\n" + strconv.Itoa(arr[median])
    }
    continue
} else if x < 0 {
    break
}
arr = append(arr, x)
}
fmt.Println(text)
}

```

Screenshoot program



```

3.go - ALPRO_2 - Visual Studio Code
TERMINAL
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_12\UNGUIDED\tempCodeRunnerFile.go"
7 23 11 0 5 19 2 29 3 13 17 0 -5315

11
12
PS D:\Personal\KULIAH\ALPRO_2>

```

Deskripsi program

Program di atas adalah sebuah aplikasi berbasis terminal yang menerima masukan angka secara bertahap dari pengguna dan menghentikan input saat angka negatif dimasukkan. Setiap kali angka 0 dimasukkan, program akan mengurutkan elemen-elemen dalam array menggunakan algoritma selection sort dalam urutan menurun (descending). Setelah data terurut, program menghitung nilai median dari array yang dihasilkan.

Median dihitung dengan cara berikut:

- Jika jumlah elemen genap, median dihitung sebagai rata-rata dari dua elemen tengah.
- Jika jumlah elemen ganjil, median adalah elemen tengah.

Hasil perhitungan median untuk setiap kali angka 0 dimasukkan akan disimpan dalam string text, yang kemudian ditampilkan setelah program selesai dijalankan. Program ini menggunakan fungsi bawaan seperti `strconv` untuk mengubah angka menjadi string dan `math.Floor` untuk pembulatan jika diperlukan.

3. Latihan Soal No. 2 Modul 13

Source code

```
package main

import (
    "fmt"
)

const nMax = 7919

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

type DaftarBuku [nMax]Buku

func DaftarkanBuku(pustaka *DaftarBuku, n *int) {
    fmt.Println("Masukkan data buku:")
    for i := 0; i < *n; i++ {
        var buku Buku
        fmt.Printf("Masukkan ID Buku: ")
        fmt.Scan(&buku.id)
        fmt.Printf("Masukkan Judul Buku: ")
        fmt.Scan(&buku.judul)
        fmt.Printf("Masukkan Penulis Buku: ")
    }
}
```

```

        fmt.Scan(&buku.penulis)
        fmt.Printf("Masukkan Penerbit Buku: ")
        fmt.Scan(&buku.penerbit)
        fmt.Printf("Masukkan Jumlah Eksemplar: ")
        fmt.Scan(&buku.eksemplar)
        fmt.Printf("Masukkan Tahun Terbit: ")
        fmt.Scan(&buku.tahun)
        fmt.Printf("Masukkan Rating Buku: ")
        fmt.Scan(&buku.rating)
        pustaka[i] = buku
    }
}

func CetakTerfavorit(pustaka DaftarBuku, n int) {
    if n == 0 {
        fmt.Println("Tidak ada data buku.")
        return
    }

    var maxRatingBuku Buku
    for i := 0; i < n; i++ {
        if pustaka[i].rating > maxRatingBuku.rating {
            maxRatingBuku = pustaka[i]
        }
    }

    fmt.Println("Buku Terfavorit:")
    fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s, Tahun: %d\n", maxRatingBuku.judul, maxRatingBuku.penulis, maxRatingBuku.penerbit, maxRatingBuku.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 Cetak5Terbesar(pustaka DaftarBuku, n int) {
    fmt.Println("5 Buku dengan Rating Tertinggi:")
    for i := 0; i < 5 && i < n; i++ {
        buku := pustaka[i]
        fmt.Printf("Judul: %s, Rating: %d\n", buku.judul,
buku.rating)
    }
}

func CariBuku(pustaka DaftarBuku, n int, rating int) {
    low := 0
    high := n - 1
    found := false

    for low <= high {
        mid := (low + high) / 2

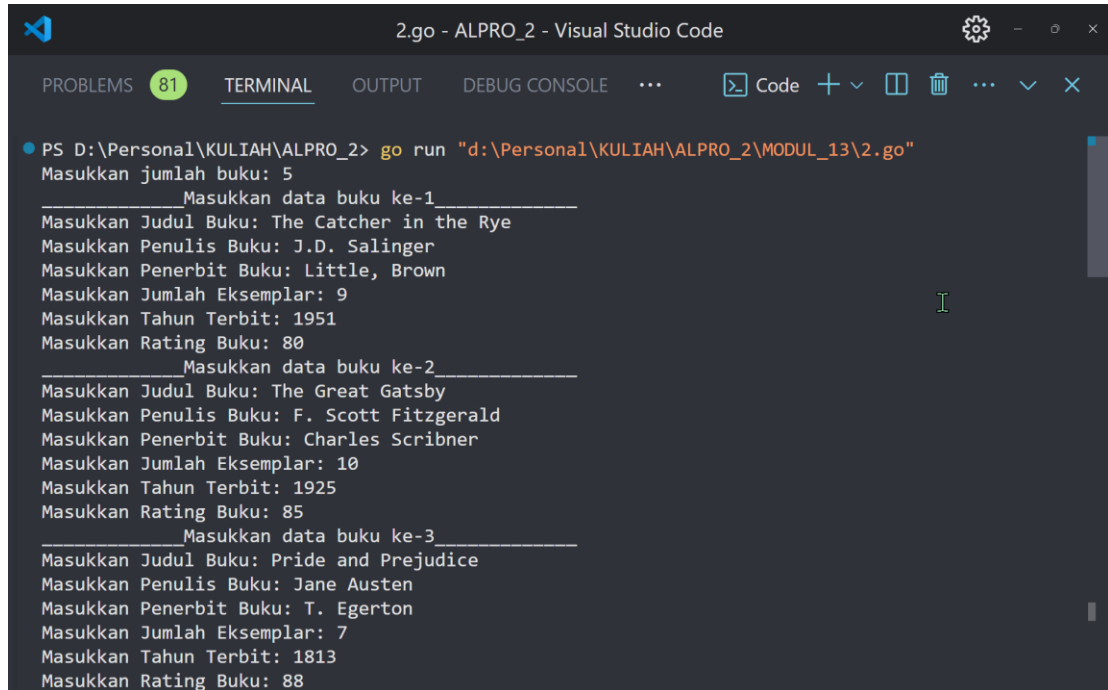
        if pustaka[mid].rating == rating {
            fmt.Println("Buku Ditemukan:")
            fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s,
Eksemplar: %d, Tahun: %d, Rating: %d\n",
                pustaka[mid].judul, pustaka[mid].penulis,
pustaka[mid].penerbit, pustaka[mid].eksemplar,
pustaka[mid].tahun, pustaka[mid].rating)
            found = true
            break
        } else if pustaka[mid].rating < rating {
            low = mid + 1
        } else {
            high = mid - 1
        }
    }

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

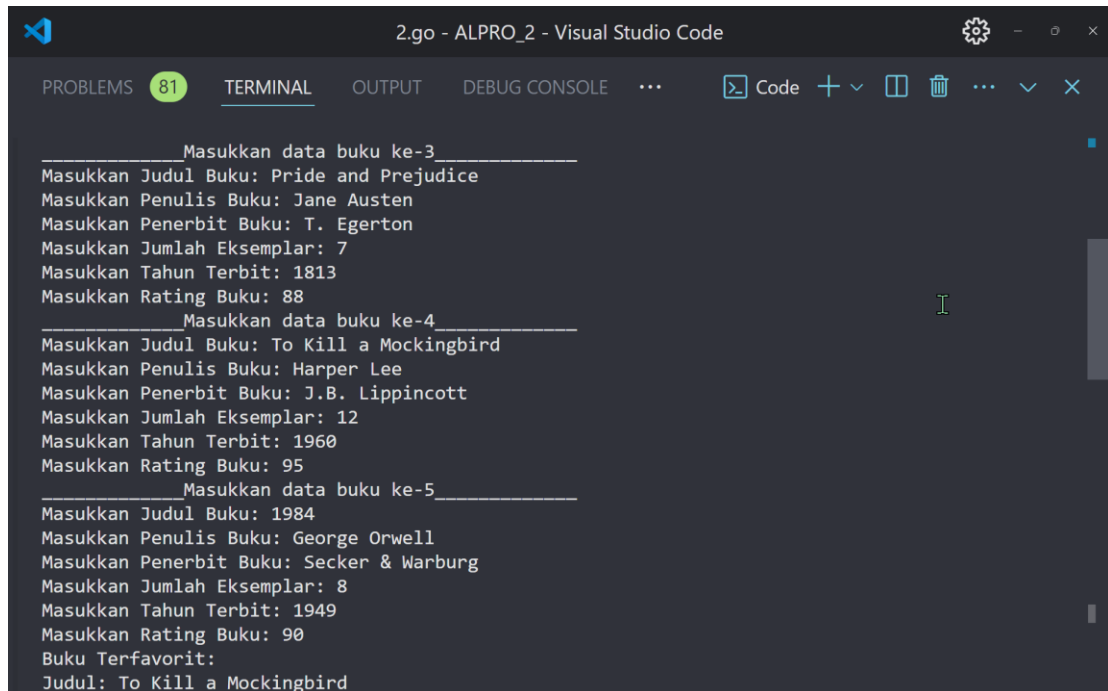
```

```
func main() {  
    var pustaka DaftarBuku  
    var n int  
    fmt.Print("Masukkan jumlah buku: ")  
    fmt.Scan(&n)  
    DaftarkanBuku(&pustaka, &n)  
    UrutBuku(&pustaka, n)  
    CetakTerfavorit(pustaka, n)  
    Cetak5Terbesar(pustaka, n)  
    var rating int  
    fmt.Print("Masukkan rating untuk mencari buku: ")  
    fmt.Scan(&rating)  
    CariBuku(pustaka, n, rating)  
}
```

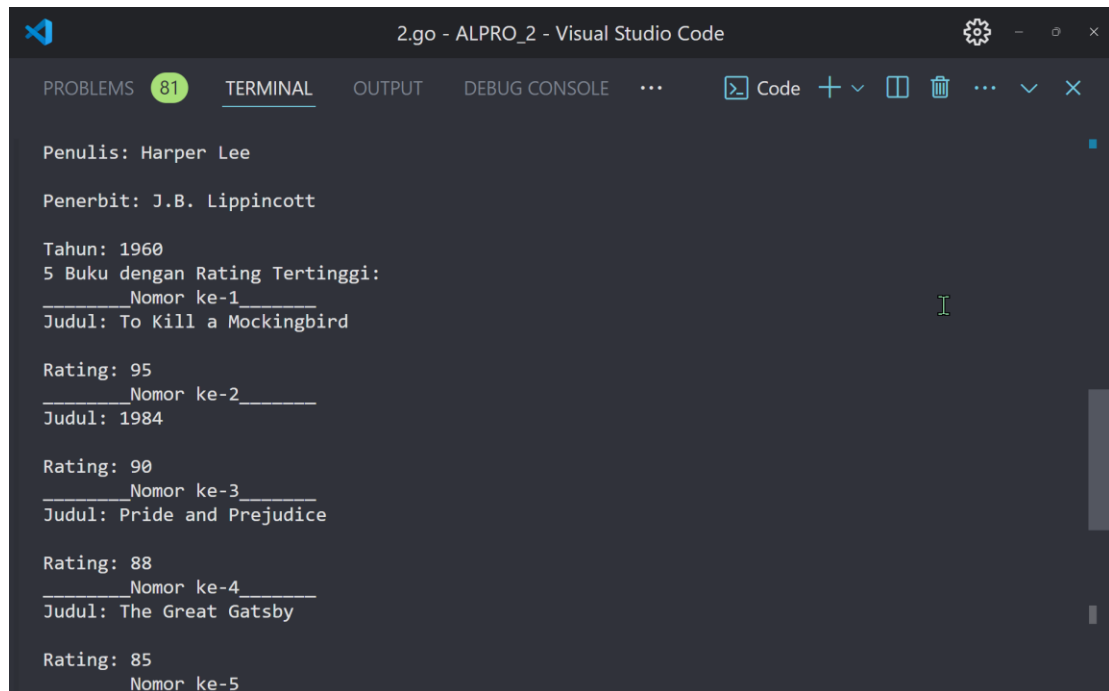
Screenshoot program



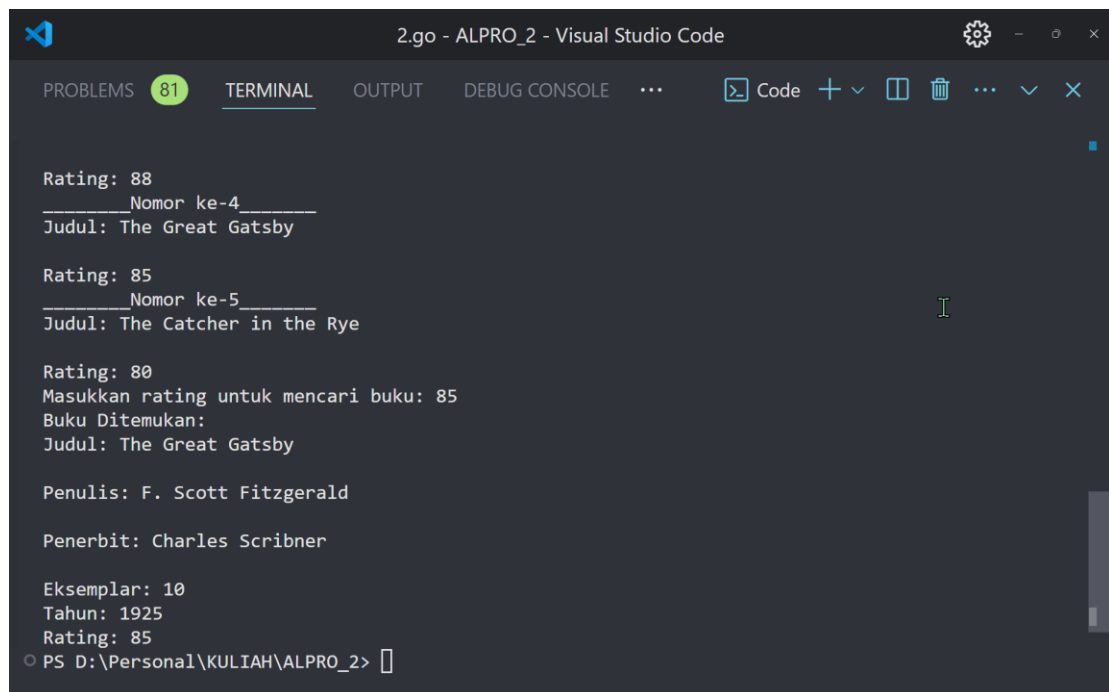
```
2.go - ALPRO_2 - Visual Studio Code
PROBLEMS 81 TERMINAL OUTPUT DEBUG CONSOLE ... Code + - [ ] [ ] ... v x
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_13\2.go"
Masukkan jumlah buku: 5
_____Masukkan data buku ke-1_____
Masukkan Judul Buku: The Catcher in the Rye
Masukkan Penulis Buku: J.D. Salinger
Masukkan Penerbit Buku: Little, Brown
Masukkan Jumlah Eksemplar: 9
Masukkan Tahun Terbit: 1951
Masukkan Rating Buku: 80
_____Masukkan data buku ke-2_____
Masukkan Judul Buku: The Great Gatsby
Masukkan Penulis Buku: F. Scott Fitzgerald
Masukkan Penerbit Buku: Charles Scribner
Masukkan Jumlah Eksemplar: 10
Masukkan Tahun Terbit: 1925
Masukkan Rating Buku: 85
_____Masukkan data buku ke-3_____
Masukkan Judul Buku: Pride and Prejudice
Masukkan Penulis Buku: Jane Austen
Masukkan Penerbit Buku: T. Egerton
Masukkan Jumlah Eksemplar: 7
Masukkan Tahun Terbit: 1813
Masukkan Rating Buku: 88
```



```
_____Masukkan data buku ke-3_____
Masukkan Judul Buku: Pride and Prejudice
Masukkan Penulis Buku: Jane Austen
Masukkan Penerbit Buku: T. Egerton
Masukkan Jumlah Eksemplar: 7
Masukkan Tahun Terbit: 1813
Masukkan Rating Buku: 88
_____Masukkan data buku ke-4_____
Masukkan Judul Buku: To Kill a Mockingbird
Masukkan Penulis Buku: Harper Lee
Masukkan Penerbit Buku: J.B. Lippincott
Masukkan Jumlah Eksemplar: 12
Masukkan Tahun Terbit: 1960
Masukkan Rating Buku: 95
_____Masukkan data buku ke-5_____
Masukkan Judul Buku: 1984
Masukkan Penulis Buku: George Orwell
Masukkan Penerbit Buku: Secker & Warburg
Masukkan Jumlah Eksemplar: 8
Masukkan Tahun Terbit: 1949
Masukkan Rating Buku: 90
Buku Terfavorit:
Judul: To Kill a Mockingbird
```



```
2.go - ALPRO_2 - Visual Studio Code
PROBLEMS 81 TERMINAL OUTPUT DEBUG CONSOLE ... Code + - [ ] [ ] ... - X
Penulis: Harper Lee
Penerbit: J.B. Lippincott
Tahun: 1960
5 Buku dengan Rating Tertinggi:
_____ Nomor ke-1 _____
Judul: To Kill a Mockingbird
Rating: 95
_____ Nomor ke-2 _____
Judul: 1984
Rating: 90
_____ Nomor ke-3 _____
Judul: Pride and Prejudice
Rating: 88
_____ Nomor ke-4 _____
Judul: The Great Gatsby
Rating: 85
_____ Nomor ke-5 _____
```



```
2.go - ALPRO_2 - Visual Studio Code
PROBLEMS 81 TERMINAL OUTPUT DEBUG CONSOLE ... Code + - [ ] [ ] ... - X
Rating: 88
_____ Nomor ke-4 _____
Judul: The Great Gatsby
Rating: 85
_____ Nomor ke-5 _____
Judul: The Catcher in the Rye
Rating: 80
Masukkan rating untuk mencari buku: 85
Buku Ditemukan:
Judul: The Great Gatsby
Penulis: F. Scott Fitzgerald
Penerbit: Charles Scribner
Eksemplar: 10
Tahun: 1925
Rating: 85
PS D:\Personal\KULIAH\ALPRO_2> [ ]
```

Deskripsi program

Program di atas adalah sebuah aplikasi manajemen data buku berbasis teks yang ditulis dalam bahasa Go. Program ini memungkinkan pengguna untuk

memasukkan data buku ke dalam koleksi, mencakup informasi seperti judul, penulis, penerbit, jumlah eksemplar, tahun terbit, dan rating. Data buku disimpan dalam array statis dengan kapasitas maksimum 7919.

Setelah pengguna memasukkan data buku, program menyediakan beberapa fitur utama:

1. Menampilkan Buku Terfavorit: Buku dengan rating tertinggi ditampilkan berdasarkan data yang dimasukkan.
2. Mengurutkan Buku Berdasarkan Rating: Data buku diurutkan secara menurun berdasarkan rating menggunakan metode penyisipan (insertion sort).
3. Menampilkan 5 Buku dengan Rating Tertinggi: Program mencetak hingga lima buku dengan rating tertinggi.
4. Pencarian Buku Berdasarkan Rating: Menggunakan algoritma pencarian biner pada data yang sudah diurutkan, pengguna dapat mencari buku dengan rating tertentu.

Program ini memanfaatkan pengolahan input menggunakan `bufio` untuk membaca string dari pengguna, serta menyediakan antarmuka sederhana berbasis terminal yang menampilkan hasil pengolahan data secara terstruktur.