

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

Modul 12

Pengurutan Data



Disusun Oleh :

Bintang Putra Angkasa (2311102255)

Kelas: S1-IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dasar Teori Selection Sort

Selection Sort adalah algoritma pengurutan sederhana yang bekerja dengan membagi array menjadi dua bagian: bagian yang sudah diurutkan dan bagian yang belum diurutkan. Algoritma ini mencari elemen terkecil dari bagian yang belum diurutkan, lalu menukarnya dengan elemen pertama di bagian tersebut. Proses ini diulangi hingga seluruh elemen berada pada posisi yang benar.

Langkah kerja Selection Sort dimulai dengan mencari elemen terkecil dalam array, kemudian menukarnya dengan elemen pertama. Selanjutnya, algoritma melanjutkan proses pada elemen berikutnya hingga seluruh array terurut.

Sebagai contoh, jika terdapat array [64, 25, 12, 22, 11], langkah-langkahnya adalah sebagai berikut:

1. Elemen terkecil adalah 11, ditukar dengan 64, hasilnya [11, 25, 12, 22, 64].
2. Elemen terkecil berikutnya adalah 12, ditukar dengan 25, hasilnya [11, 12, 25, 22, 64].
3. Elemen terkecil berikutnya adalah 22, ditukar dengan 25, hasilnya [11, 12, 22, 25, 64].

Array sudah terurut [11, 12, 22, 25, 64].

Selection Sort mudah diimplementasikan tetapi kurang efisien untuk dataset besar karena membutuhkan waktu komputasi yang relatif lama. Algoritma ini lebih cocok digunakan untuk dataset kecil atau pembelajaran dasar sorting.

II. Guided

GUIDED 1

■ Hercules, preman terkenal seantero ibukota, memiliki kerabat di banyak daerah. Tentunya Hercules sangat suka mengunjungi semua kerabatnya itu. Diberikan masukan nomor rumah dari semua kerabatnya di suatu daerah, buatlah program rumahkerabat yang akan menyusun nomor-nomor rumah kerabatnya secara terurut membesar menggunakan algoritma 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

```

go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktikum Alpro 2/Modul 12/Code/Guided1.go"
(base) bintangputraangkasa@Bintangs-MacBook-Air ~ % go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktik
um Alpro 2/Modul 12/Code/Guided1.go"
Masukkan jumlah daerah kerabat (n): 1

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 5
Masukkan 5 nomor rumah kerabat: 4 3 2 1 5
Nomor rumah terurut untuk daerah 1: 1 2 3 4 5
(base) bintangputraangkasa@Bintangs-MacBook-Air ~ %

```

Deskripsi program

Program ini ditulis menggunakan bahasa Go untuk mengurutkan nomor rumah kerabat di setiap daerah menggunakan algoritma Selection Sort. Pengguna akan diminta memasukkan jumlah daerah, jumlah nomor rumah di setiap daerah, dan daftar nomor rumah tersebut. Program kemudian akan mengurutkan nomor rumah secara ascending melalui fungsi selectionSort dan menampilkan hasilnya untuk setiap daerah

GUIDED 2

2) 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 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.

Sourcecode

```
package main

import (
    "fmt"
)

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

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

    for daerah := 1; daerah <= n; daerah++ {
        var m int
```

```

fmt.Printf("\nMasukkan jumlah nomor rumah kerabat
untuk daerah %d: ", daerah)
fmt.Scan(&m)

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

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

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

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

```

Screenshoot Output

```

(base) bintangputraangkasa@Bintang-MacBook-Air ~ % go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktikum Alpro 2/Modul 12/Code/Guided2.go"
Masukkan jumlah daerah kerabat (n): 1
Masukkan jumlah nomor rumah kerabat untuk daerah 1: 5
Masukkan 5 nomor rumah kerabat: 12 13 14 20 5
Nomor rumah terurut untuk daerah 1: 5 13 20 14 12
(base) bintangputraangkasa@Bintang-MacBook-Air ~ %

```

Deskripsi program

Program ini mengurutkan nomor rumah kerabat dalam beberapa daerah berdasarkan kriteria ganjil dan genap. Pengguna memasukkan jumlah daerah dan nomor rumah untuk masing-masing daerah. Nomor rumah ganjil diurutkan dari yang terkecil, sementara nomor rumah genap diurutkan secara dari yang terbesar. Fungsi `selectionSort` digunakan untuk melakukan pengurutan setelah pengurutan, program menampilkan nomor rumah yang terurut per daerah, dengan nomor ganjil terlebih dahulu, diikuti nomor genap.

III. UNGUIDED

Unguided 1

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.

Sourcecode

```
package main

import "fmt"

const maxSize = 1000000

func calculateMedian(data []int) int {
    n := len(data)
    if n%2 == 1 {
        return data[n/2]
    }
}
```

```

}
return (data[n/2-1] + data[n/2]) / 2
}

func insertionSort(data []int) []int {
for i := 1; i < len(data); i++ {
key := data[i]
j := i - 1
for j >= 0 && data[j] > key {
data[j+1] = data[j]
j--
}
data[j+1] = key
}
return data
}

func main() {
var data []int
var result []int
var num int

fmt.Println("Masukkan data (akhiri dengan -
5313):")

for {
fmt.Scan(&num)

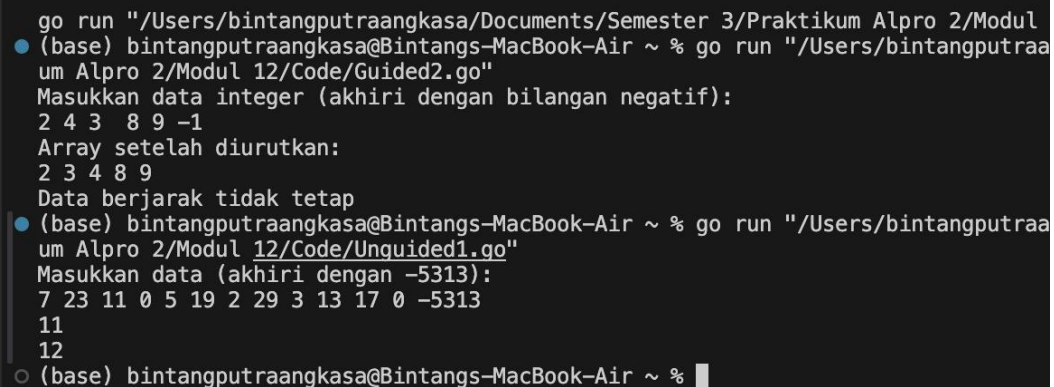
if num == -5313 {
for _, r := range result {
fmt.Println(r)
}
return
} else if num == 0 {
data = insertionSort(data)
median := calculateMedian(data)
result = append(result, median)
} else {
if len(data) >= maxArraySize {
fmt.Println("Error: Jumlah data melebihi batas
maksimum 1.000.000 elemen.")
continue
}
}
}
}

```



```
data = append(data, num)
}
}
}
```

Screenshoot Output



```
go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktikum Alpro 2/Modul
(base) bintangputraangkasa@Bintangs-MacBook-Air ~ % go run "/Users/bintangputraa
um Alpro 2/Modul 12/Code/Guided2.go"
Masukkan data integer (akhiri dengan bilangan negatif):
2 4 3 8 9 -1
Array setelah diurutkan:
2 3 4 8 9
Data berjarak tidak tetap
(base) bintangputraangkasa@Bintangs-MacBook-Air ~ % go run "/Users/bintangputraa
um Alpro 2/Modul 12/Code/Unguided1.go"
Masukkan data (akhiri dengan -5313):
7 23 11 0 5 19 2 29 3 13 17 0 -5313
11
12
(base) bintangputraangkasa@Bintangs-MacBook-Air ~ %
```

Deskripsi

Program ini ditulis dengan Go untuk membaca bilangan bulat, mengurutkannya menggunakan Insertion Sort, dan menghitung median setiap kali pengguna memasukkan angka 0. Data diterima terus-menerus hingga bilangan khusus -5313 dimasukkan untuk menghentikan program. Jika jumlah data melebihi 1.000.000 elemen, program akan menampilkan pesan error. Median dihitung dari data yang telah diurutkan, dengan hasil median disimpan dan ditampilkan saat program berakhir.

\

Unguided 2

Buatlah sebuah program yang digunakan untuk membaca data integer seperti contoh yang diberikan di bawah ini, kemudian diurutkan (menggunakan metoda insertion sort), dan memeriksa apakah data yang terurut berjarak sama terhadap data sebelumnya.

Sourcecode

```
package main

import "fmt"

func insertionSort(arr []int) {
    for i := 1; i < len(arr); i++ {
        key := arr[i]
        j := i - 1
        for j >= 0 && arr[j] > key {
            arr[j+1] = arr[j]
            j--
        }
        arr[j+1] = key
    }
}

func hitungJarak(arr []int) (bool, int) {
    if len(arr) < 2 {
        return true, 0
    }
    spacing := arr[1] - arr[0]
    for i := 2; i < len(arr); i++ {
        if arr[i]-arr[i-1] != spacing {
            return false, 0
        }
    }
    return true, spacing
}

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

    fmt.Println("Masukkan bilangan bulat (akhiri dengan bilangan negatif):")
}
```

```

for {
    _, err := fmt.Scan(&input)
    if err != nil {
        fmt.Println("Terjadi kesalahan input:", err)
        return
    }
    if input < 0 {
        break
    }
    data = append(data, input)
}

if len(data) == 0 {
    fmt.Println("Tidak ada data yang diproses.")
    return
}

insertionSort(data)

fmt.Println("Data setelah diurutkan:")
for _, num := range data {
    fmt.Printf("%d ", num)
}
fmt.Println()

isConstant, spacing := hitungJarak(data)
if isConstant {
    fmt.Printf("Data berjarak %d\n", spacing)
} else {
    fmt.Println("Data berjarak tidak tetap")
}
}

```

Screenshoot program

```
go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktikum Alpro 2/Modul 12/Code/unguided2.go"
(base) bintangputraangkasa@Bintang-MacBook-Air ~ % go run "/Users/bintangputraangkasa/Documents/Semester 3/Praktikum Alpro 2/Modul 12/Code/unguided2.go"
Masukkan bilangan bulat (akhiri dengan bilangan negatif):
2 3 5 6 4 1 -5
Data setelah diurutkan:
1 2 3 4 5 6
Data berjarak 1
(base) bintangputraangkasa@Bintang-MacBook-Air ~ %
```

Deskripsi Program

Program ini ditulis dalam bahasa Go untuk mengurutkan data bilangan bulat menggunakan Insertion Sort dan memeriksa apakah selisih antar elemen dalam array yang diurutkan tetap sama. Pengguna memasukkan bilangan bulat secara berulang hingga memasukkan bilangan negatif, yang menandakan akhir input. Jika konsisten, program menampilkan selisih tetap antar elemen; jika tidak, program menginformasikan bahwa selisih tidak tetap

Unguided 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.

```
package main

import "fmt"

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

const nMax = 7919
type DaftarBuku [nMax]Buku

func DaftarkanBuku(pustaka *DaftarBuku, n *int) {
    fmt.Println("Masukkan jumlah buku:")
    fmt.Scanln(n)
```

```

for i := 0; i < *n; i++ {
fmt.Println("Masukkan data buku ke-", i+1)
fmt.Print("ID: ")
fmt.Scan(&pustaka[i].id)
fmt.Print("Judul: ")
fmt.Scan(&pustaka[i].judul)
fmt.Print("Penulis: ")
fmt.Scan(&pustaka[i].penulis)
fmt.Print("Penerbit: ")
fmt.Scan(&pustaka[i].penerbit)
fmt.Print("Tahun: ")
fmt.Scan(&pustaka[i].tahun)
fmt.Print("Eksemplar: ")
fmt.Scan(&pustaka[i].eksemplar)
fmt.Print("Rating: ")
fmt.Scan(&pustaka[i].rating)
}
}

func CetakTerfavorit(pustaka DaftarBuku, n int) {
terfavorit := pustaka[0]
for i := 1; i < n; i++ {
if pustaka[i].rating > terfavorit.rating {
terfavorit = pustaka[i]
}
}
fmt.Println("Buku Terfavorit:")
fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s,
Tahun: %d, Rating: %d\n",
terfavorit.judul, terfavorit.penulis,
terfavorit.penerbit, terfavorit.tahun,
terfavorit.rating)
}

func UrutBuku(pustaka *DaftarBuku, n int) {
for i := 1; i < n; i++ {
key := pustaka[i]
j := i - 1
// Mengurutkan berdasarkan rating secara menurun
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("Lima Buku dengan Rating Tertinggi:")
for i := 0; i < 5 && i < n; i++ {
fmt.Printf("%d. Judul: %s, Rating: %d\n", i+1,
pustaka[i].judul, pustaka[i].rating)
}
}

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

for low <= high {
mid := (low + high) / 2
if pustaka[mid].rating == r {
fmt.Printf("Buku dengan rating %d ditemukan:\n",
r)
fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s,
Tahun: %d, Eksemplar: %d, Rating: %d\n",
pustaka[mid].judul, pustaka[mid].penulis,
pustaka[mid].penerbit, pustaka[mid].tahun,
pustaka[mid].eksemplar, pustaka[mid].rating)
found = true
break
} else if pustaka[mid].rating < r {
high = mid - 1
} else {
low = mid + 1
}
}

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

```

```

func main() {
var pustaka DaftarBuku
var nPustaka int

DaftarkanBuku(&pustaka, &nPustaka)
UrutBuku(&pustaka, nPustaka)
CetakTerfavorit(pustaka, nPustaka)
Cetak5Terbaru(pustaka, nPustaka)

var ratingCari int
fmt.Print("Masukkan rating yang akan dicari: ")
fmt.Scan(&ratingCari)
CariBuku(pustaka, nPustaka, ratingCari)
}

```

screenshoot

```

Masukkan data buku ke- 5
ID: 5
Judul: Meinkampf
Penulis: Adlofhitler
Penerbit: nazi
Tahun: 1936
Eksemplar: 100000
Rating: 10
Buku Terfavorit:
Judul: Meinkampf, Penulis: Adlofhitler, Penerbit: nazi, Tahun: 1936, Rating: 10
Lima Buku dengan Rating Tertinggi:
1. Judul: Meinkampf, Rating: 10
2. Judul: Machinelearning, Rating: 9
3. Judul: Madilog, Rating: 8
4. Judul: Hindiabelanda, Rating: 7
5. Judul: Duniastophie, Rating: 6
Masukkan rating yang akan dicari: 10
Buku dengan rating 10 ditemukan:
Judul: Meinkampf, Penulis: Adlofhitler, Penerbit: nazi, Tahun: 1936, Eksemplar: 100000, Rating: 10
(base) bintangputraangkasa@Bintang-MacBook-Air ~ %

```

Deskripsi Program

Program menggunakan bahasa go ,program i mengelola daftar buku dengan fitur pendaftaran, pengurutan berdasarkan rating, pencetakan buku terfavorit, dan pencarian buku berdasarkan rating. Buku disimpan dalam struktur Buku, yang mencakup ID, judul, penulis, penerbit, tahun, eksemplar, dan rating. Data buku dapat diurutkan menggunakan Insertion Sort berdasarkan rating, dan buku terfavorit serta dengan rating tertinggi ditampilkan kemudian user juga dapat mencari buku sesuai rating yang di inginkan

