

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL XII & MODUL XIII  
PENGURUTAN DATA**



**Disusun Oleh :**

**Didik Setiawan**

**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

### Dasar Teori

Sorting adalah proses pengurutan elemen data dalam susunan tertentu, seperti menaik (ascending) atau menurun (descending). Proses ini bertujuan untuk mempermudah pencarian, pengolahan data, dan mengoptimalkan algoritma lainnya. Contoh:

- Mengurutkan angka 3, 1, 4, 2 menjadi 1, 2, 3, 4 (ascending).
- Mengurutkan daftar nama berdasarkan abjad.

### Jenis-Jenis Algoritma Sorting

Berikut beberapa algoritma sorting yang umum digunakan:

#### a. Bubble Sort

- **Konsep:** Membandingkan elemen yang bersebelahan dan menukar posisinya jika urutannya tidak sesuai.
- **Kelebihan:** Mudah untuk diimplementasikan.
- **Kekurangan:** Tidak efisien untuk dataset besar, dengan kompleksitas waktu sebesar  $O(n^2)$ .

#### b. Selection Sort

- **Konsep:** Menemukan elemen terkecil dari bagian data yang belum terurut, lalu menukarnya dengan elemen pertama.
- **Kelebihan:** Sederhana dalam implementasi.
- **Kekurangan:** Kurang efisien untuk data besar, dengan kompleksitas waktu  $O(n^2)$ .

#### c. Insertion Sort

- **Konsep:** Mengambil elemen dari bagian data yang belum terurut satu per satu, kemudian menempatkannya di posisi yang sesuai pada bagian data yang sudah terurut.
- **Kelebihan:** Efektif untuk dataset kecil.

- **Kekurangan:** Tidak cocok untuk dataset besar karena kompleksitas waktu  $O(n^2)$ .

#### d. Merge Sort

- **Konsep:** Membagi data menjadi dua bagian, mengurutkan setiap bagian secara rekursif, kemudian menggabungkannya kembali.
- **Kelebihan:** Memiliki kompleksitas waktu yang lebih baik, yaitu  $O(n \log n)$ .
- **Kekurangan:** Membutuhkan ruang memori tambahan untuk proses penggabungan.

#### e. Quick Sort

- **Konsep:** Memilih elemen pivot, membagi data menjadi dua bagian berdasarkan pivot (elemen lebih kecil di kiri, lebih besar di kanan), lalu mengurutkannya secara rekursif.
- **Kelebihan:** Sangat cepat untuk dataset besar, dengan kompleksitas waktu  $O(n \log n)$ .

## II. GUIDED

1. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

### 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)
```

```

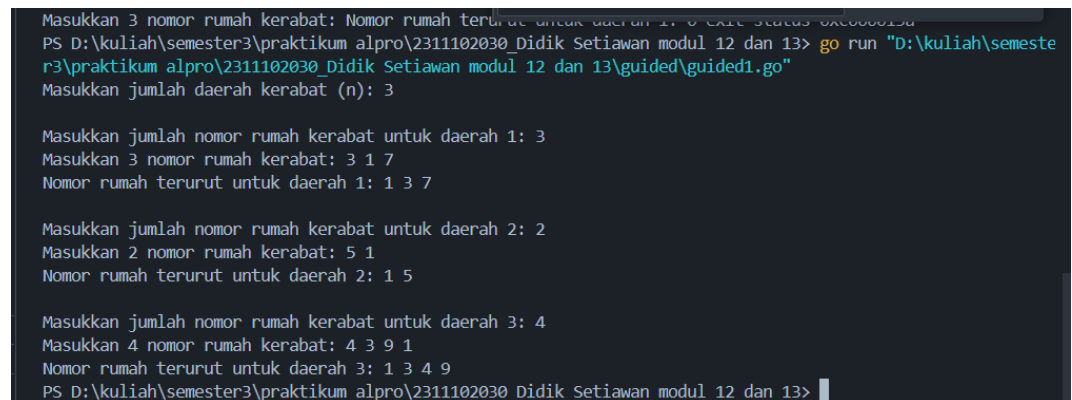
    fmt.Printf("Masukkan %d nomor rumah kerabat: ", m)
    for i := 0; i < m; i++ {
        fmt.Scan(&arr[i])
    }

    // 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



```

Masukkan 3 nomor rumah kerabat: Nomor rumah terurut untuk daerah 1: 0 Exit status: 0x0000013a
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13> go run "D:\kuliah\semeste
r3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13\guided\guided1.go"
Masukkan jumlah daerah kerabat (n): 3

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

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

Masukkan jumlah nomor rumah kerabat untuk daerah 3: 4
Masukkan 4 nomor rumah kerabat: 4 3 9 1
Nomor rumah terurut untuk daerah 3: 1 3 4 9
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13>

```

## Deskripsi Program

implementasi dari algoritma **Selection Sort** untuk mengurutkan nomor rumah kerabat di beberapa daerah secara terurut dari terkecil ke terbesar. Program meminta pengguna untuk memasukkan jumlah daerah dan jumlah nomor rumah di setiap daerah, lalu mengurutkan nomor rumah tersebut menggunakan Selection Sort. Hasilnya ditampilkan dalam urutan yang telah diurutkan untuk masing-masing daerah, sehingga mempermudah pengelolaan data nomor rumah.

2. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

## 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

```

Masukkan jumlah nomor rumah kerabat untuk daerah 3: 4
Masukkan 4 nomor rumah kerabat: 4 3 9 1
Nomor rumah terurut untuk daerah 3: 1 3 4 9
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13> go run "D:\kuliah\seme
r3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13\guided\guided2.go"
Masukkan data integer (akhiri dengan bilangan negatif):
2 1 4 -1
Array setelah diurutkan:
1 2 4
Data berjarak tidak tetap
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13>

```

### **Deskripsi Program**

untuk mengurutkan sekumpulan bilangan bulat yang dimasukkan pengguna hingga bilangan negatif ditemukan. Setelah pengurutan, program memeriksa apakah selisih antar elemen dalam array tetap konstan. Jika selisihnya tetap, program menampilkan nilai jarak tersebut; jika tidak, program memberi tahu bahwa selisihnya tidak tetap



### III. UNGUIDED

1. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

#### Soal Studi Case

#### Sourcecode

```
package main

import (
    "fmt"
    "sort"
)

func processDaerah(m int) ([]int, []int) {
    arr := make([]int, m)
    fmt.Printf("Masukkan %d nomor rumah kerabat: ", m)
    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) {
    sort.Ints(odd) // Sort ascending
    sort.Sort(sort.Reverse(sort.IntSlice(even))) // Sort descending

    fmt.Printf("\nNomor rumah terurut untuk daerah ke-%d:\n", i+1)
    fmt.Println("Odd numbers: ", odd)
    fmt.Println("Even numbers:", even)
}

func main() {
```

```

var n int
fmt.Println("\n=====
=====")
fmt.Print("Masukkan jumlah daerah kerabat (n): ")
fmt.Scan(&n)

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

    odd, even := processDaerah(m)
    printSortedResults(i, odd, even)
}

fmt.Println("\nProgram selesai.")
}

```

## Screenshoot Output

```
PS D:\kuliah\semester3\praktikum alpro\2311102030\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13\unguided\unguided1.go"

=====
Masukkan jumlah daerah kerabat (n): 3

-----
Masukkan jumlah rumah kerabat untuk daerah ke-1: 5
Masukkan 5 nomor rumah kerabat: 5 10 6 7 2

Nomor rumah terurut untuk daerah ke-1:
Odd numbers: [5 7]
Even numbers: [10 6 2]

-----
Masukkan jumlah rumah kerabat untuk daerah ke-2: 3
Masukkan 3 nomor rumah kerabat: 101 5 23

Nomor rumah terurut untuk daerah ke-2:
Odd numbers: [5 23 101]
Even numbers: []

-----
Masukkan jumlah rumah kerabat untuk daerah ke-3: 4
Masukkan 4 nomor rumah kerabat: 5 4 6 1

Nomor rumah terurut untuk daerah ke-3:
Odd numbers: [1 5]
Even numbers: [6 4]
```

### Deskripsi Program

memasukkan jumlah daerah dan jumlah rumah di setiap daerah. Untuk setiap daerah, nomor rumah dipisahkan menjadi dua kelompok: bilangan ganjil dan bilangan genap. Nomor ganjil diurutkan secara menaik (ascending), sedangkan nomor genap diurutkan secara menurun (descending). Hasil pengelompokan dan pengurutan ditampilkan secara terpisah untuk setiap daerah, memudahkan pengguna dalam mengelola data nomor rumah secara terstruktur.

2. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

### Soal Studi Case

#### Sourcecode

```
package main
```

```

import (
    "fmt"
    "sort"
)

func median(arr []int) int {
    n := len(arr)
    if n%2 == 0 {
        return (arr[n/2-1] + arr[n/2]) / 2
    }
    return arr[n/2]
}

func main() {
    var arr []int
    var arrMedian []int

    fmt.Println("Masukkan data (akhiri dengan -5313): ")
    for {
        var datum int
        fmt.Scan(&datum)
        if datum == -5313 {
            break
        }
        if datum == 0 {
            sort.Ints(arr) // Sort array
            if len(arr) > 0 {
                arrMedian = append(arrMedian, median(arr))
            }
        } else {
            arr = append(arr, datum)
        }
    }

    fmt.Println("Output:")
    for _, med := range arrMedian {
        fmt.Println(med)
    }
}

```

## Screenshoot Output

```
Output:
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13> go run "D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13\unguided\unguided2.go"
Masukkan data (akhiri dengan -5313):
7 5 13 101 8 12 15 19 55 22 34 12 59 29 0 -5313
Output:
17
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13>
```

## Deskripsi Program

Data dimasukkan sebagai bilangan bulat, dengan beberapa aturan berikut:

1. Jika pengguna memasukkan angka **0**, program menghitung median dari data yang sudah dimasukkan hingga saat itu (dengan data diurutkan lebih dahulu) dan menyimpannya dalam daftar median.
2. Jika pengguna memasukkan angka **-5313**, program berhenti menerima input.
3. Angka lainnya ditambahkan ke dalam daftar data untuk diproses.

Setelah semua input selesai, program mencetak daftar median yang dihitung setiap kali angka **0** dimasukkan selama sesi input

3. Berisi source code dan output dari kegiatan praktikum yang telah dilaksanakan. Source Code diberi penjelasan maka akan menjadi nilai ++

## Soal Studi Case

### Sourcecode

```
package main

import (
    "fmt"
    "sort"
)

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

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) {
    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) {
    sort.Slice(pustaka, func(i, j int) bool {
        return (pustaka[i].rating > pustaka[j].rating)
    })
}

func Cetak5Terbaik(pustaka []Buku) {
    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, rating int) {
    ditemukan := false
    for _, buku := range pustaka {
        if buku.rating == rating {
            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)
    UrutkanBuku(&pustaka)
    Cetak5Terbaik(pustaka)

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

```

## Screenshoot Output

```
r3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13\unguided\unguided3.go"
Masukkan jumlah buku di perpustakaan: 4
Masukkan data untuk buku ke-1 (id, judul, penulis, penerbit, eksemplar, tahun, rating):
2 matahari mamat gramedia 2 2013 5
Masukkan data untuk buku ke-2 (id, judul, penulis, penerbit, eksemplar, tahun, rating):
1
rembulan ipul gramedia 2 2015 3
Masukkan data untuk buku ke-3 (id, judul, penulis, penerbit, eksemplar, tahun, rating):
3 airdanapi naif awanhitam 3 2016 3
Masukkan data untuk buku ke-4 (id, judul, penulis, penerbit, eksemplar, tahun, rating):
4 mahasiswa didik indiebuku 5 2024 4
Buku dengan rating tertinggi:
ID: 2, Judul: matahari, Penulis: mamat, Penerbit: gramedia, Eksemplar: 2, Tahun: 2013, Rating: 5
Lima buku dengan rating tertinggi:
ID: 2, Judul: matahari, Penulis: mamat, Penerbit: gramedia, Eksemplar: 2, Tahun: 2013, Rating: 5
ID: 4, Judul: mahasiswa, Penulis: didik, Penerbit: indiebuku, Eksemplar: 5, Tahun: 2024, Rating: 4
ID: 1, Judul: rembulan, Penulis: ipul, Penerbit: gramedia, Eksemplar: 2, Tahun: 2015, Rating: 3
ID: 3, Judul: airdanapi, Penulis: naif, Penerbit: awanhitam, Eksemplar: 3, Tahun: 2016, Rating: 3
Masukkan rating buku yang ingin dicari: 5
ID: 2, Judul: matahari, Penulis: mamat, Penerbit: gramedia, Eksemplar: 2, Tahun: 2013, Rating: 5
PS D:\kuliah\semester3\praktikum alpro\2311102030_Didik Setiawan modul 12 dan 13>
```

## Deskripsi Program

mengelola data buku di sebuah perpustakaan dengan menyediakan fitur untuk mendaftarkan buku, mencari buku berdasarkan rating, dan menampilkan informasi buku dengan rating tertinggi. Fitur-fitur utama meliputi:

1. **Pendaftaran Buku:** Pengguna dapat memasukkan data buku seperti ID, judul, penulis, penerbit, jumlah eksemplar, tahun, dan rating.
2. **Pencarian Buku Favorit:** Menampilkan buku dengan rating tertinggi di pustaka.
3. **Pengurutan Buku:** Mengurutkan daftar buku berdasarkan rating secara menurun.
4. **Menampilkan 5 Buku Terbaik:** Setelah pengurutan, program mencetak lima buku dengan rating tertinggi.
5. **Pencarian Buku Berdasarkan Rating:** Pengguna dapat mencari buku yang memiliki rating tertentu, jika ada.

Program membatasi jumlah buku maksimum hingga 7919 untuk memastikan data tetap terkelola dengan baik