LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL XII

PENGURUTAN DATA

Disusun Oleh :

Dimas Akal Hernanda/2311102249

IF-11-06

Dosen Pengampu :

ABEDNEGO DWI SEPTIADI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Pengurutan data dalam bahasa pemrograman Go dapat dilakukan dengan menggunakan paket sort. Untuk mengurutkan angka, string, atau angka desimal, kita dapat menggunakan fungsi seperti sort.Ints(), sort.Strings(), atau sort.Float64s(). Jika data yang ingin diurutkan lebih kompleks, seperti struktur, kita bisa membuat aturan pengurutan sendiri dengan mengimplementasikan interface sort.Interface yang terdiri dari tiga fungsi: Len(), Less(), dan Swap(). Selain itu, Go juga menyediakan fungsi sort.Reverse() untuk mengurutkan data secara terbalik (descending). Dengan menggunakan paket sort, pengurutan data menjadi lebih mudah dan fleksibel.

II. Guided

1.Guided 1

Source Code

```
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

Program ini mengurutkan nomor rumah kerabat di beberapa daerah menggunakan algoritma *Selection Sort*. Pengguna memasukkan jumlah daerah, lalu untuk setiap daerah memasukkan jumlah nomor rumah dan daftar nomor tersebut. Program akan mengurutkan daftar nomor rumah secara *ascending* dan menampilkan hasilnya untuk setiap daerah.

2.Guided 2

Source Code

```
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

```
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 12> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 12\guided2.go"
Masukkan data integer (akhiri dengan bilangan negatif):
3
6
9
12
15
18
21
-23
Array setelah diurutkan:
3 6 9 12 15 18 21
Data berjarak 3
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 12> go run "c:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 12\guided2.go"
Masukkan data integer (akhiri dengan bilangan negatif):
2
7
5
3
4
-12
Array setelah diurutkan:
2 3 4 5 7
Data berjarak tidak tetap
PS C:\Users\acer\OneDrive\Documents\Laporan praktikum alpro 2\2311102249_Dimas Akal Hernanda_Modul 12>
```

Deskripsi Program

Program ini membaca serangkaian bilangan integer hingga menemukan bilangan negatif, mengurutkannya menggunakan *Insertion Sort*, lalu memeriksa apakah selisih antar elemen yang berurutan tetap. Hasil pengurutan dan status jarak elemen (tetap atau tidak) ditampilkan di akhir.

III. Unguided

1.Unguided 1

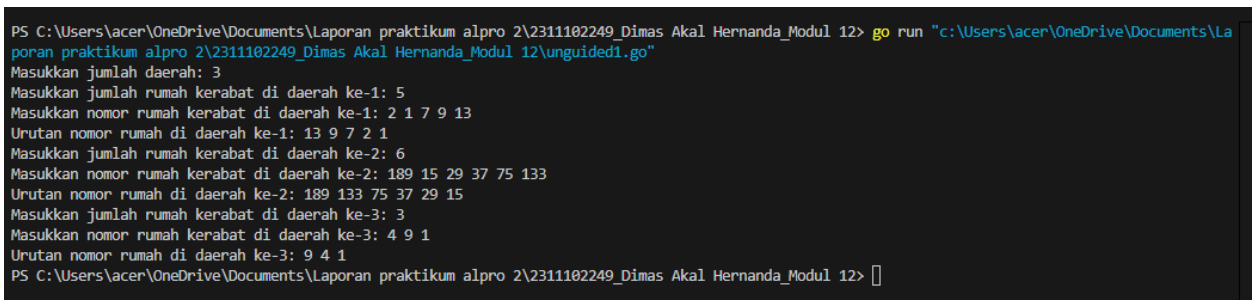
Source Code

```
package main


import (
    "fmt"
)


// Fungsi untuk mengurutkan array menggunakan selection sort
func selectionSortDesc(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
            }
        }
        // Tukar elemen
        arr[i], arr[maxIdx] = arr[maxIdx], arr[i]
    }
}

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

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

        // Input nomor rumah
```

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

        // Urutkan menggunakan selection sort
        selectionSortDesc(houses)

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

Screenshoot Output



Deskripsi Program

Program ini mengurutkan nomor rumah kerabat di beberapa daerah secara *descending* menggunakan algoritma *Selection Sort*. Pengguna memasukkan jumlah daerah dan nomor rumah di setiap daerah. Setelah diurutkan, program menampilkan daftar nomor rumah yang telah terurut untuk masing-masing daerah.

2.Unguided 2

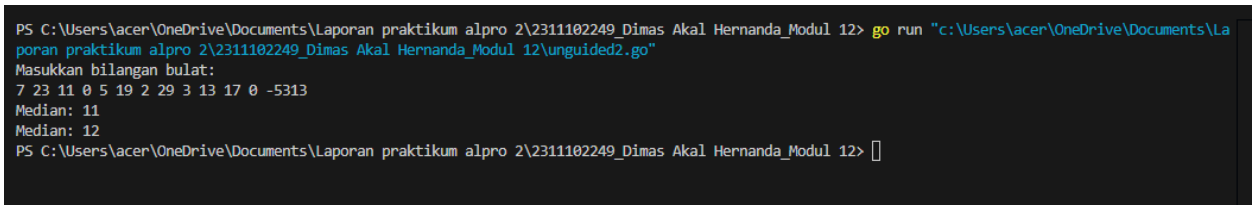
Source Code

```
package main


import (
    "bufio"
    "fmt"
    "os"
    "sort"
    "strconv"
    "strings"
)


// Fungsi untuk menghitung median dari array yang sudah diurutkan
func calculateMedian(arr []int) int {
    n := len(arr)
    if n%2 == 1 {
        // Jika jumlah data ganjil, ambil nilai tengah
        return arr[n/2]
    } else {
```

```
        // Jika jumlah data genap, hitung rata-rata dua nilai tengah
        return (arr[n/2-1] + arr[n/2]) / 2
    }
}

func main() {
    var data []int
    scanner := bufio.NewScanner(os.Stdin)

    fmt.Println("Masukkan bilangan bulat:")

    // Membaca input dari pengguna
    if scanner.Scan() {
        input := scanner.Text()
        strNums := strings.Fields(input)

        for _, str := range strNums {
            num, err := strconv.Atoi(str)
            if err != nil {
                fmt.Println("Masukkan hanya bilangan bulat!")
                return
            }

            if num == -5313 {
                // Marker akhir input, keluar dari loop
```

```
            break
        } else if num == 0 {
            // Ketika menemukan angka 0, urutkan data dan cetak median
            sort.Ints(data) // Mengurutkan data
            if len(data) > 0 {
                median := calculateMedian(data)
                fmt.Println("Median:", median)
            } else {
                fmt.Println("Tidak ada data untuk menghitung median.")
            }
        } else if num > 0 {
            // Tambahkan bilangan positif ke array
            data = append(data, num)
        }
    }
}
}
```

Screenshoot Output



Deskripsi Program

Program ini membaca bilangan bulat dari pengguna, menyimpan bilangan positif, lalu menghitung dan mencetak median setiap kali angka 0 dimasukkan. Median dihitung dari bilangan yang sudah diurutkan.

3.Unguided 3

Source Code

```
package main


import (
    "fmt"
)


const MaxBooks int = 7919


type Book struct {
    ID, Title, Author, Publisher string
    Copies, Year, Rating         int
}


type BookList [MaxBooks]Book


func RegisterBooks(library *BookList, count *int) {
    fmt.Print("Enter the number of books: ")
    fmt.Scanln(count)
```

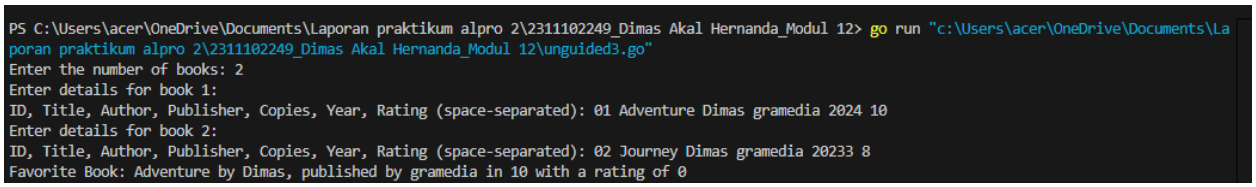
```
    for i := 0; i < *count; i++ {
        fmt.Printf("Enter details for book %d:\n", i+1)
        fmt.Print("ID, Title, Author, Publisher, Copies, Year, Rating (space-separated): ")
        fmt.Scanln(&library[i].ID, &library[i].Title, &library[i].Author, &library[i].Publisher,
&library[i].Copies, &library[i].Year, &library[i].Rating)
    }
}

func PrintFavoriteBook(library BookList, count int) {
    favorite := library[0]
    for i := 1; i < count; i++ {
        if library[i].Rating > favorite.Rating {
            favorite = library[i]
        }
    }
    fmt.Printf("Favorite Book: %s by %s, published by %s in %d with a rating of %d\n",
        favorite.Title, favorite.Author, favorite.Publisher, favorite.Year, favorite.Rating)
}

func SortBooksByRating(library *BookList, count int) {
    for i := 1; i < count; i++ {
        temp := library[i]
        j := i - 1
        for j >= 0 && library[j].Rating < temp.Rating {
            library[j+1] = library[j]
```

```
            j--
        }
        library[j+1] = temp
    }
}

func PrintTop5Books(library BookList, count int) {
    fmt.Println("Top 5 Books by Rating:")
    limit := 5
    if count < 5 {
        limit = count
    }
    for i := 0; i < limit; i++ {
        fmt.Printf("%d. %s (Rating: %d)\n", i+1, library[i].Title, library[i].Rating)
    }
}

func SearchBookByRating(library BookList, count int, rating int) {
    low, high := 0, count-1
    found := false
    var idx int

    for low <= high {
        mid := (low + high) / 2
        if library[mid].Rating == rating {
```

```
            found = true
            idx = mid
            break
        } else if library[mid].Rating < rating {
            low = mid + 1
        } else {
            high = mid - 1
        }
    }

    if found {
        fmt.Printf("Book found: %s by %s, published by %s in %d, %d copies, Rating: %d\n",
            library[idx].Title, library[idx].Author, library[idx].Publisher, library[idx].Year, library[idx].Copies, library[idx].Rating)
    } else {
        fmt.Println("No book found with the given rating.")
    }
}

func main() {
    var library BookList
    var bookCount, searchRating int

    RegisterBooks(&library, &bookCount)
```

```
    PrintFavoriteBook(library, bookCount)
    SortBooksByRating(&library, bookCount)
    PrintTop5Books(library, bookCount)


    fmt.Print("Enter the rating of the book to search for: ")
    fmt.Scanln(&searchRating)
    SearchBookByRating(library, bookCount, searchRating)
}
```

Screenshoot Output



Deskripsi Program

Program di atas adalah aplikasi manajemen perpustakaan sederhana yang memungkinkan pengguna untuk:

1. **Mendaftarkan Buku**: Memasukkan data buku seperti ID, judul, penulis, penerbit, jumlah salinan, tahun, dan rating.
2. **Menampilkan Buku Favorit**: Menentukan buku dengan rating tertinggi.
3. **Mengurutkan Buku Berdasarkan Rating**: Mengurutkan daftar buku dalam urutan menurun berdasarkan rating.
4. **Menampilkan Buku Teratas**: Menampilkan buku dengan rating tertinggi.
5. **Mencari Buku Berdasarkan Rating**: Mencari buku dengan rating tertentu menggunakan algoritma pencarian biner.

Program ini menggunakan array tetap untuk menyimpan data buku dan berfokus pada manipulasi serta pencarian data dengan efisien.