LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 8

MATERI

Oleh:

DAFFA TSAQIFNA FAUZTSANY

103112400032

S1 IF-12-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

Pencarian Nilai Acak

1. Sequential Search (Pencarian Berurutan)

Mencari data dari elemen pertama hingga terakhir satu per satu sampai ditemukan atau habis.

Contoh:

```
func seqSearch(arr []int, n, x int) int {
    for i := 0; i < n; i++ {
        if arr[i] == x {
            return i
        }
    }
    return -1 // tidak ditemukan
}
```

2. Binary Search (Pencarian Biner)

Pencarian lebih cepat dari sequential, tapi syaratnya data harus terurut. Pencarian dilakukan dengan membagi dua data secara terus-menerus.

Contoh (ascending):

```
func binarySearch(arr []int, n, x int) int {
    left, right := 0, n-1
    for left <= right {
        mid := (left + right) / 2
        if arr[mid] == x {
            return mid
        } else if arr[mid] < x {
            left = mid + 1
        } else {
            right = mid - 1
        }
    }
    return -1
}
```

3. Pencarian pada Struct

Pencarian pada array of struct bisa dilakukan berdasarkan field tertentu (misal nama, nim, ipk). Binary search bisa digunakan jika data sudah diurutkan berdasarkan field yang dicari.

4. Perbandingan Singkat

Jenis Pencarian	Kecepatan	Syarat Data
Sequential Search	Lambat	Tidak perlu terurut
Binary Search	Cepat	Harus terurut

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

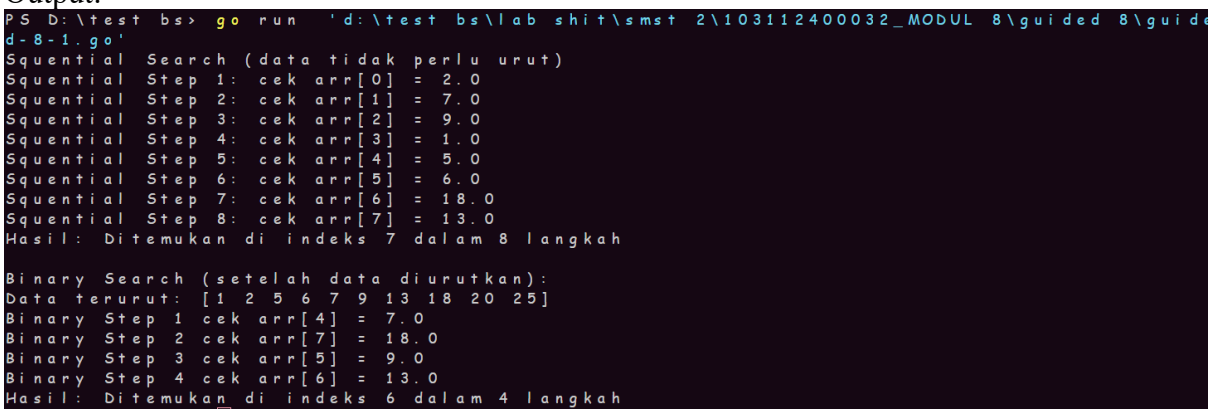
1. **GUIDED 1**

Source Code:

```
package main

import (
    "fmt"
    "sort"
)

func sequentialSearch(arr []float64, target float64) (int, int) {
    iterations := 0
    for i, val := range arr {
        iterations++
        fmt.Printf("Squential Step %d: cek arr[%d] = %.1f\n", iterations, i, val)
        if val == target {
            return i, iterations
        }
    }
    return -1, iterations
}

func binarySearch(arr []float64, target float64) (int, int) {
    iterations := 0
    low := 0
    high := len(arr) - 1

    for low <= high {
        iterations++
        mid := (low + high) / 2
        fmt.Printf("Binary Step %d cek arr[%d] = %.1f\n", iterations, mid, arr[mid])

        if arr[mid] == target {
            return mid, iterations
        } else if target < arr[mid] {
            high = mid - 1
        } else {
            low = mid + 1
        }
    }
    return -1, iterations
}

func main() {
    data := []float64{2, 7, 9, 1, 5, 6, 18, 13, 25, 20}
    target := 13.0
```

```
    fmt.Println("Squential Search (data tidak perlu urut)")
    idxSeq, iterSeq := sequentialSearch(data, target)
    if idxSeq != 1 {
        fmt.Printf("Hasil: Ditemukan di indeks %d dalam %d langkah\n\n", idxSeq,
iterSeq)
    } else {
        fmt.Printf("Hasil: Tidak ditemukan setelah %d langkah\n\n", iterSeq)
    }

    sort.Float64s(data)
    fmt.Println("Binary Search (setelah data diurutkan):")
    fmt.Println("Data terurut:", data)

    idxBin, iterBin := binarySearch(data, target)
    if idxBin != -1 {
        fmt.Printf("Hasil: Ditemukan di indeks %d dalam %d langkah\n", idxBin,
iterBin)
    } else {
        fmt.Printf("Hasil: Tidak ditemukan setelah %d langkah\n", iterBin)
    }
}
```

Output:



Deskripsi Program:

Program ini digunakan untuk membandingkan proses pencarian data menggunakan dua metode: sequential search dan binary search. Sequential search mencari elemen target secara linear tanpa membutuhkan data terurut, sementara binary search bekerja lebih efisien tetapi membutuhkan data dalam kondisi terurut terlebih dahulu. Program menampilkan langkah-langkah pencarian dari kedua metode, menghitung jumlah iterasi, dan mencetak apakah data ditemukan serta di indeks ke berapa.

2. **GUIDED 2**

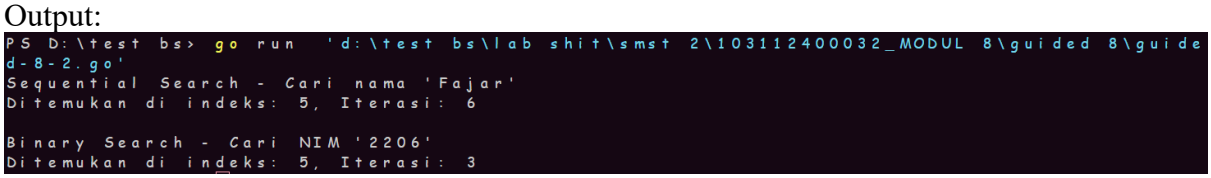
Source Code:

```
package main

import (
```

```
    "fmt"
    "sort"
)

type mahasiswa struct {
    nama, nim, kelas, jurusan string
    ipk                       float64
}

type arrMhs [2023]mahasiswa

// Sequential Search berdasarkan nama
func SeqSearch_3(T arrMhs, n int, X string) (int, int) {
    var found int = -1
    var j int = 0
    var iterasi int = 0

    for j < n && found == -1 {
        iterasi++
        if T[j].nama == X {
            found = j
        }
        j++
    }
    return found, iterasi
}

// Binary Search berdasarkan NIM (data harus sudah terurut berdasarkan nim)
func BinarySearch_3(T arrMhs, n int, X string) (int, int) {
    var found int = -1
    var med int
    var kr int = 0
    var kn int = n - 1
    var iterasi int = 0

    for kr <= kn && found == -1 {
        iterasi++
        med = (kr + kn) / 2
        if X < T[med].nim {
            kn = med - 1
        } else if X > T[med].nim {
            kr = med + 1
        } else {
            found = med
        }
    }
    return found, iterasi
}

func main() {
```

```
    var data arrMhs
    n := 10

    // Mengisi data secara manual
    data = arrMhs{
        {nama: "Ari", nim: "2201", kelas: "A", jurusan: "Informatika", ipk: 3.4},
        {nama: "Budi", nim: "2203", kelas: "A", jurusan: "Informatika", ipk: 3.6},
        {nama: "Cici", nim: "2202", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.5},
        {nama: "Dina", nim: "2205", kelas: "A", jurusan: "Informatika", ipk: 3.3},
        {nama: "Eko", nim: "2204", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.7},
        {nama: "Fajar", nim: "2206", kelas: "C", jurusan: "Informatika", ipk: 3.1},
        {nama: "Gita", nim: "2209", kelas: "C", jurusan: "Informatika", ipk: 3.8},
        {nama: "Hana", nim: "2208", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.2},
        {nama: "Iwan", nim: "2207", kelas: "C", jurusan: "Informatika", ipk: 3.0},
        {nama: "Joko", nim: "2210", kelas: "A", jurusan: "Informatika", ipk: 3.9},
    }

    // Pencarian Sequential Search berdasarkan nama
    namaDicari := "Fajar"
    idxSeq, iterSeq := SeqSearch_3(data, n, namaDicari)

    fmt.Printf("Sequential Search - Cari nama '%s'\n", namaDicari)
    if idxSeq != -1 {
        fmt.Printf("Ditemukan di indeks: %d, Iterasi: %d\n", idxSeq, iterSeq)
    } else {
        fmt.Printf("Tidak ditemukan, Iterasi: %d\n", iterSeq)
    }

    // Urutkan data berdasarkan NIM untuk binary search
    sort.Slice(data[:n], func(i, j int) bool {
        return data[i].nim < data[j].nim
    })

    // Pencarian Binary Search berdasarkan NIM
    nimDicari := "2206"
    idxBin, iterBin := BinarySearch_3(data, n, nimDicari)

    fmt.Printf("\nBinary Search - Cari NIM '%s'\n", nimDicari)
    if idxBin != -1 {
        fmt.Printf("Ditemukan di indeks: %d, Iterasi: %d\n", idxBin, iterBin)
    } else {
        fmt.Printf("Tidak ditemukan, Iterasi: %d\n", iterBin)
    }
}
```

Output:



Deskripsi Program:

Program ini digunakan untuk melakukan pencarian data mahasiswa menggunakan dua metode: Sequential Search berdasarkan nama dan Binary Search berdasarkan NIM. Data mahasiswa disimpan dalam array statis bertipe struct yang mencakup nama, NIM, kelas, jurusan, dan IPK. Pencarian nama dilakukan secara linear (sequential), sementara pencarian NIM dilakukan secara biner setelah data diurutkan berdasarkan NIM. Program juga menghitung dan menampilkan jumlah iterasi yang dibutuhkan dalam setiap metode pencarian.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

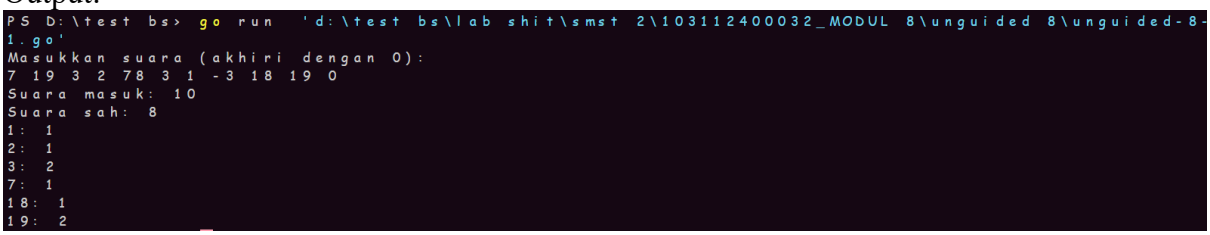
1. UNGUIDED 1

Source Code:

```
package main

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

const maxCalon = 20

func main() {
    var suaraMasuk, suaraSah int
    var hasil [maxCalon + 1]int // indeks 1-20 dipakai

    scanner := bufio.NewScanner(os.Stdin)
    fmt.Println("Masukkan suara (akhiri dengan 0):")
    scanner.Scan()
    input := scanner.Text()
    tokens := strings.Fields(input)

    for _, token := range tokens {
        num, err := strconv.Atoi(token)
        if err != nil {
            continue
        }
        if num == 0 {
            break
        }
        suaraMasuk++
        if num >= 1 && num <= maxCalon {
            suaraSah++
            hasil[num]++
        }
    }

    fmt.Printf("Suara masuk: %d\n", suaraMasuk)
    fmt.Printf("Suara sah: %d\n", suaraSah)
    for i := 1; i <= maxCalon; i++ {
        if hasil[i] > 0 {
            fmt.Printf("%d: %d\n", i, hasil[i])
        }
    }
```

```
}
```

Output:



Deskripsi Program:

Program ini digunakan untuk merekap hasil pemungutan suara dalam sebuah pemilihan dengan maksimal 20 calon. Pengguna memasukkan daftar angka dalam satu baris (masing-masing mewakili nomor calon), dan input diakhiri dengan angka 0. Program menghitung total suara masuk, suara sah (nomor calon 1–20), serta mencatat jumlah suara yang diterima setiap calon. Hanya calon yang memperoleh suara akan ditampilkan pada hasil akhir.

2. UNGUIDED 2

Source Code:

```
package main

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

type Calon struct {
    Nomor int
    Suara int
}

func main() {
    scanner := bufio.NewScanner(os.Stdin)
    scanner.Scan()
    input := scanner.Text()

    parts := strings.Fields(input)

    var (
        totalSuara int
        suaraValid int
        suaraCalon = make(map[int]int)
    )
```

```
    for _, part := range parts {
        num, err := strconv.Atoi(part)
        if err != nil {
            continue
        }

        totalSuara++

        if num == 0 {
            break
        }

        if num >= 1 && num <= 20 {
            suaraValid++
            suaraCalon[num]++
        }
    }

    var calons []Calon
    for nomor, suara := range suaraCalon {
        calons = append(calons, Calon{Nomor: nomor, Suara: suara})
    }

    sort.Slice(calons, func(i, j int) bool {
        if calons[i].Suara == calons[j].Suara {
            return calons[i].Nomor < calons[j].Nomor
        }
        return calons[i].Suara > calons[j].Suara
    })

    fmt.Printf("Suara masuk: %d\n", totalSuara)
    fmt.Printf("Suara sah: %d\n", suaraValid)

    if len(calons) == 0 {
        fmt.Println("Tidak ada calon yang mendapatkan suara")
    } else if len(calons) == 1 {
        fmt.Printf("Ketua RT: %d\n", calons[0].Nomor)
        fmt.Println("Wakil ketua: Tidak ada")
    } else {
        fmt.Printf("Ketua RT: %d\n", calons[0].Nomor)
        fmt.Printf("Wakil ketua: %d\n", calons[1].Nomor)
    }
}
```

Output:

```
PS D:\test bs> go run 'd:\test bs\lab shit\smst 2\103112400032_MODUL 8\unguided 8\unguided-8-2.go'
7 19 3 2 78 3 1 -3 18 19 0
Suara masuk: 11
Suara sah: 8
Ketua RT: 3
Wakil ketua: 19
```

Deskripsi Program:

Program ini digunakan untuk menghitung dan menentukan hasil pemilihan Ketua dan Wakil Ketua RT berdasarkan input suara dalam satu baris. Suara sah adalah yang bernilai 1–20, sedangkan nilai 0 digunakan untuk mengakhiri input. Program mencatat jumlah total suara, suara sah, dan menghitung jumlah suara tiap calon menggunakan map. Setelah itu, data calon diurutkan berdasarkan jumlah suara (terbanyak ke terkecil), dan jika sama, berdasarkan nomor calon yang lebih kecil. Program kemudian menampilkan pemenang sebagai Ketua dan Wakil Ketua RT.

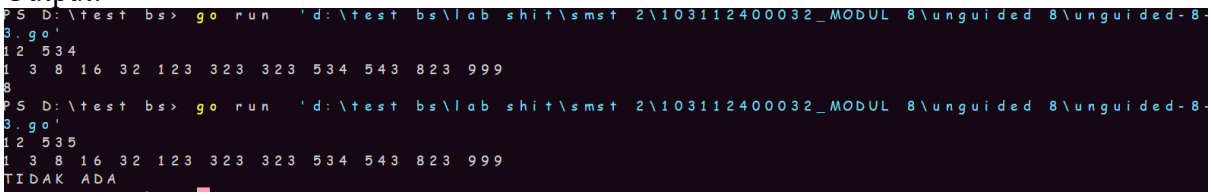
3. UNGUIDED 3

Source Code:

```
package main

package main

import "fmt"

const NMAX = 1000000

var data [NMAX]int

func main() {
    var n, k int
    fmt.Scan(&n, &k)

    isiArray(n)
    pos := posisi(n, k)

    if pos == -1 {
        fmt.Println("TIDAK ADA")
    } else {
        fmt.Println(pos)
    }
}

func isiArray(n int) {

    for i := 0; i < n; i++ {
        fmt.Scan(&data[i])
    }
}

func posisi(n, k int) int {

    left := 0
    right := n - 1

    for left <= right {
        mid := left + (right-left)/2
```

```
        if data[mid] == k {
            return mid
        } else if data[mid] < k {
            left = mid + 1
        } else {
            right = mid - 1
        }
    }

    return -1
}
```

Output:



Deskripsi Program:

Program ini digunakan untuk mencari posisi sebuah bilangan k dalam array data menggunakan binary search. Program menerima input jumlah elemen n dan nilai yang dicari k, lalu membaca n angka dan menyimpannya dalam array. Fungsi posisi melakukan pencarian biner dengan asumsi bahwa data sudah dalam keadaan terurut. Jika nilai ditemukan, program mencetak indeksnya; jika tidak, mencetak "TIDAK ADA".