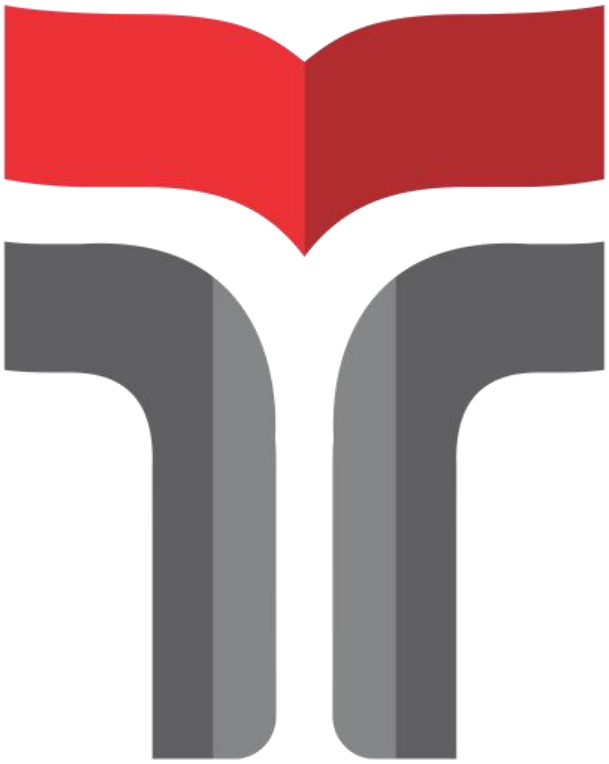
LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 8

MATERI



Oleh:

ABISAR FATHIR

103112400068

12-IF-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

- **Sequential Search (Pencarian Sekuensial):**

  Dilakukan dengan memeriksa elemen satu per satu dari awal hingga akhir.

  Cocok untuk data yang tidak terurut.

  Implementasinya sederhana, tetapi kurang efisien untuk data besar.

- **Binary Search (Pencarian Biner):**

  Dilakukan dengan membagi dua ruang pencarian secara berulang.

  Hanya dapat digunakan pada data yang **terurut**.

  Jauh lebih efisien daripada sequential search karena kompleksitasnya O(log n).

  Perlu penyesuaian jika data terurut menurun (descending).

- **Pencarian pada Array Bertipe Struct:**

  Pencarian dapat dilakukan berdasarkan satu atribut tertentu seperti nama atau nim.

  Binary search hanya bisa digunakan jika data sudah terurut berdasarkan atribut pencarian tersebut.

- **Implementasi dalam Studi Kasus:**

  Soal 1 dan 2 mempraktikkan pencarian dan penghitungan frekuensi dalam pemilihan ketua RT, menggunakan sequential search.

  Soal 3 menunjukkan penerapan binary search untuk menemukan posisi elemen dalam array besar yang sudah terurut

II. GUIDED

1. Program ke 1

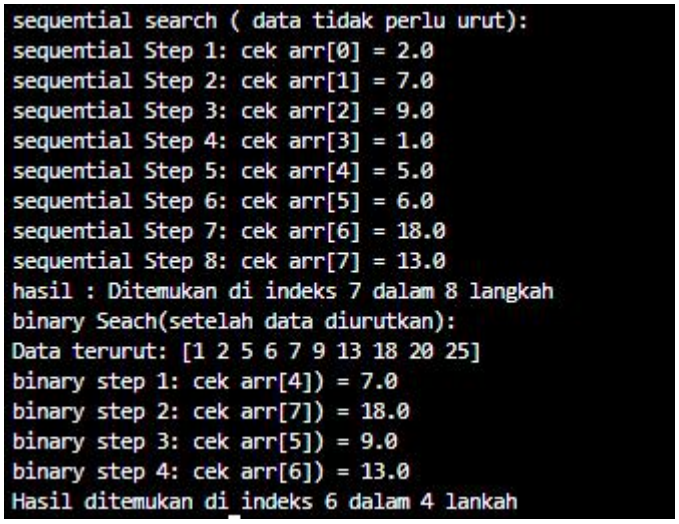
Source Code:

```
package main


import (

"fmt"

"sort"

)


func sequentialSearch(arr []float64, target float64) (int, int) {

iteration := 0

for i, val := range arr {

iteration++

fmt.Printf("sequential Step %d: cek arr[%d] = %.1f\n", iteration, i, val)

if val == target {

return i, iteration

}

}

return -1, iteration

}

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("sequential search ( data tidak perlu urut):")

idxSeq, iterSeq := sequentialSearch(data, target)
```

```
if idxSeq != 1 {

fmt.Printf("hasil : Ditemukan di indeks %d dalam %d langkah\n", idxSeq, iterSeq)

} else {

fmt.Printf("tidak ditemukan hasil dalam %d langkah\n", iterSeq)

}

sort.Float64s(data)

fmt.Println("binary Seach(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 lankah \n", idxBin, iterBin)

} else {


fmt.Printf(" tidak ditemukan dalam %d lankah \n", iterBin)

}

}
```

Output Program:



1. Deskripsi Program:Program digunakan untuk contoh penggunaan search sequantial dan binary dengan contoh mencari angka 8 yang berada di index 7

2. Program ke 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 {
```

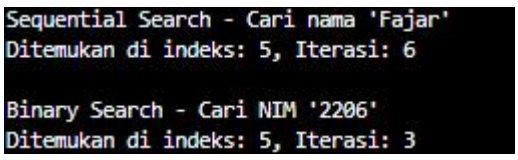
```
found = j
}
j++
}
return found, iterasi
}

// 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 Program



:

Deskripsi Program:Program Program ini melakukan pencarian data mahasiswa dalam sebuah array statis menggunakan dua metode pencarian:

III. UNGUIDED

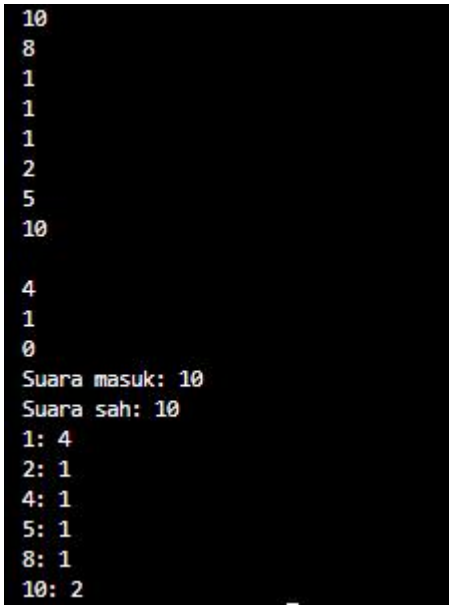
1. Program ke 1

Source Code:

```
package main


import "fmt"


func main() {
const MAX_CANDIDATE = 20
var suaraMasuk, suaraSah int
var suara [MAX_CANDIDATE + 1]int
var input int


for {
fmt.Scan(&input)
if input == 0 {
break
}
suaraMasuk++
if input >= 1 && input <= MAX_CANDIDATE {
suara[input]++
suaraSah++
```

```
}

}



fmt.Printf("Suara masuk: %d\n", suaraMasuk)

fmt.Printf("Suara sah: %d\n", suaraSah)



for i := 1; i <= MAX_CANDIDATE; i++ {

if suara[i] > 0 {

fmt.Printf("%d: %d\n", i, suara[i])

}

}

}
```

Output Program:



Deskripsi Program:ini dibuat untuk menghitung hasil pemilihan ketua RT berdasarkan suara warga. Dalam pemilihan ini terdapat **20 calon** (dengan nomor 1 sampai 20), dan warga memberikan suara dengan menuliskan nomor calon. Namun, beberapa warga mungkin menuliskan angka di luar rentang tersebut, sehingga program juga harus **memvalidasi suara**.

2. Program ke 2

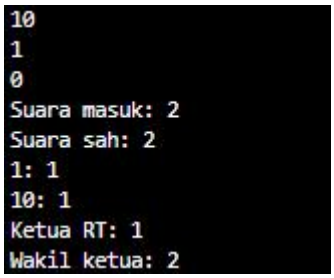
Source Code:

```
package main


import "fmt"


func main() {

const MAX_CANDIDATE = 20

var suaraMasuk, suaraSah int

var suara [MAX_CANDIDATE + 1]int

var input int
```

```
for {
fmt.Scan(&input)
if input == 0 {
break
}
suaraMasuk++
if input >= 1 && input <= MAX_CANDIDATE {
suara[input]++
suaraSah++
}
}

fmt.Printf("Suara masuk: %d\n", suaraMasuk)
fmt.Printf("Suara sah: %d\n", suaraSah)

for i := 1; i <= MAX_CANDIDATE; i++ {
if suara[i] > 0 {
fmt.Printf("%d: %d\n", i, suara[i])
}
}

ketua, wakil := 0, 0
```

```
max1, max2 := -1, -1


for i := 1; i <= MAX_CANDIDATE; i++ {
if suara[i] > max1 {
max2 = max1
wakil = ketua
max1 = suara[i]
ketua = i
} else if suara[i] == max1 && i < ketua {
max2 = max1
wakil = ketua
ketua = i
} else if suara[i] > max2 && suara[i] < max1 {
max2 = suara[i]
wakil = i
} else if suara[i] == max2 && i < wakil {
wakil = i
}
}


fmt.Printf("Ketua RT: %d\n", ketua)
fmt.Printf("Wakil ketua: %d\n", wakil)
}
```

Output Program:



Deskripsi Program:Program ini digunakan untuk pilkart yang mencari siapa pemenang pemilihan ketua RT. Sekaligus juga ditentukan bahwa wakil ketua RT adalah calon yang mendapatkan suara terbanyak kedua.

3. Program ke 3

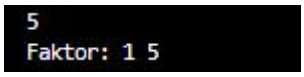
Source Code:

```
package main


import "fmt"


const NMAX = 1000000

var data [NMAX]int


func main() {

var n, k int

fmt.Scan(&n, &k)

isiArray(n)
```

```
idx := posisi(n, k)

if idx == -1 {

fmt.Println("TIDAK ADA")

} else {

fmt.Println(idx)

}

}




func isiArray(n int) {

for i := 0; i < n; i++ {

fmt.Scan(&data[i])

}

}




func posisi(n, k int) int {

low := 0

high := n - 1

for low <= high {

mid := (low + high) / 2

if data[mid] == k {

return mid
```

```
} else if data[mid] < k {

low = mid + 1

} else {

high = mid - 1

}

}

return -1

}
```

Output Program:



Deskripsi Program : ini digunakan untuk mencari apakah suatu bilangan k terdapat dalam sebuah array data[] yang berisi n bilangan bulat positif **terurut membesar**. Jika ditemukan, program akan mencetak indeksnya. Jika tidak, akan mencetak "TIDAK ADA".

IV. KESIMPULAN

Kesimpulan Modul 8: Pencarian Nilai Acak pada Himpunan Data

Modul 8 membahas algoritma pencarian data dalam struktur data array dengan dua pendekatan utama:

Sequential Search (Pencarian Sekuensial) melakukan pencarian secara berurutan dari elemen pertama hingga terakhir. Proses pencarian berhenti ketika data yang dicari ditemukan atau seluruh data sudah diperiksa. Metode ini cocok untuk data yang tidak terurut (unordered) dengan kompleksitas waktu rata-rata O(n).

Binary Search (Pencarian Biner) mensyaratkan data harus terurut (ascending atau descending). Metode ini membagi ruang pencarian menjadi dua bagian di setiap iterasi dan menggunakan prinsip "divide and conquer" untuk efisiensi. Kompleksitas waktu pencarian biner adalah O(log n).

Kedua algoritma pencarian ini dapat dimodifikasi untuk mengembalikan nilai boolean (true/false) yang menunjukkan apakah data ditemukan atau indeks/posisi data yang dicari dalam array (atau -1 jika tidak ditemukan).

Pencarian pada array bertipe data struct mirip dengan tipe data dasar, namun perlu menentukan field yang menjadi kriteria pencarian. Untuk Binary Search, keterurutan array harus berdasarkan field yang sama dengan kriteria pencarian.

Modul ini juga menyediakan latihan soal yang menerapkan konsep pencarian dalam kasus nyata, seperti perhitungan suara pemilihan dan pencarian bilangan dalam array terurut.

V. REFERENSI