

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 8
PENCARIAN NILAI ACAK PADA HIMPUNAN DATA



Oleh:

Muhammad Gamel Al Ghifari

103112400028

IF-12-01

S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

8.1 Sequential Search

Pencarian secara sekuensial ini adalah pencarian yang dilakukan dari data pertama, kedua hingga terakhir secara satu persatu dan berurutan. Ciri khas dari pencarian ini adalah proses pencarian akan berhenti ketika data yang dicari ditemukan, walaupun masih terdapat data yang belum dicek nilainya.

Algoritma ini dikenal dengan nama Sequential Search, karena prosesnya melakukan pengecekan setiap elemen array secara satu persatu dan sekuensial dari data pertama hingga ditemukan atau data terakhir.

- 1.) Asumsi terdapat suatu array of integer T dengan indeks dari 0 hingga N-1, dan suatu nilai yang dicari pada array T, yaitu X.
- 2.) Status pencarian digunakan untuk menandakan data yang dicari ditemukan atau tidak, misalnya found dengan tipe boolean.
- 3.) Pencarian dilakukan dari T[0] sampai ke T[N-1], setiap kali perbandingan dengan X, update nilai found.
- 4.) Perulangan harus dihentikan apabila status pencarian found bernilai true (data ditemukan) atau T[N-1] telah dicek.

| | Notasi Algoritma | Notasi dalam bahasa Go |
|---|------------------------------|------------------------|
| 1 | found \leftarrow false | found = false |
| 2 | i \leftarrow 0 | i = 0 |
| 3 | while i < n and not found do | for i < n && !found { |
| 4 | found \leftarrow T[i] == X | found = T[i] == X |
| 5 | i \leftarrow i + 1 | i = i + 1 |
| 6 | endwhile | } |

8.2 Binary Search

Ide algoritma adalah: (dengan asumsi data terurut dari kecil membesar (ascending), dan data dengan indeks kecil ada di "kiri" dan indeks besar ada di "kanan")

- 1.) Ambil salah satu data dalam rentang data yang ada, algoritma di bawah menggunakan rentang dari kiri **kr** s.d. kanan **kn**. Untuk kemudahan dan alasan lainnya, biasanya yang diambil adalah paling tengah dalam rentang tersebut.
- 2.) Jika data terambil tersebut terlalu kecil, maka ubah/geser rentang data ke sebelah kanan posisi data tersebut. Ini karena jika data terambil terlalu kecil, maka semua data sebelah kirinya juga akan terlalu kecil dari yang ingin dicari.
- 3.) Begitu juga sebaliknya jika data terambil terlalu besar.

| | Notasi Algoritma | Notasi dalam bahasa Go |
|----|---|------------------------------------|
| 1 | $kr \leftarrow 0$ | $kr = 0$ |
| 2 | $kn \leftarrow n-1$ | $kn = n-1$ |
| 3 | $found \leftarrow false$ | $found = false$ |
| 4 | while $kr \leq kn$ and not found do | for $kr \leq kn \ \&\& \ !found$ { |
| 5 | $med \leftarrow (kr+kn) \text{ div } 2$ | $med = (kr+kn) / 2$ |
| 6 | if $a[med] < X$ then | if $a[med] < X$ { |
| 7 | $kr \leftarrow med + 1$ | $kr = med + 1$ |
| 8 | else if $a[med] > X$ then | } else if $a[med] > X$ { |
| 9 | $kn \leftarrow med - 1$ | $kn = med$ |
| 10 | else | } else { |
| 11 | $found \leftarrow true$ | $found = true$ |
| 12 | endif | } |
| 13 | endwhile | } |

II. 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("Sequential 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("Sequential Search (data tidak perluurut)")
    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 :

```
PS C:\Users\User\Documents\103112400028_MODUL8> go run "c:\Users\User\Documents\103112400028_MODUL8\GUIDED\1.go"
Sequential Search (data tidak perlu urut)
Sequential Step 1: cek arr[0] = 2.0
Sequential Step 2: cek arr[1] = 7.0
Sequential Step 3: cek arr[2] = 9.0
Sequential Step 4: cek arr[3] = 1.0
Sequential Step 5: cek arr[4] = 5.0
Sequential Step 6: cek arr[5] = 6.0
Sequential Step 7: cek arr[6] = 18.0
Sequential Step 8: cek arr[7] = 13.0
Hasil: Ditemukan di indeks 7 dalam 8 langkah

Binary Search (setelah data diurutkan):
Data terurut: [1 2 5 6 7 9 13 18 20 25]
Binary Step 1 cek arr[4] = 7.0
Binary Step 2 cek arr[7] = 18.0
Binary Step 3 cek arr[5] = 9.0
Binary Step 4 cek arr[6] = 13.0
Hasil: Ditemukan di indeks 6 dalam 4 langkah
PS C:\Users\User\Documents\103112400028_MODUL8>
```

Penjelasan : Program ini bertujuan untuk mencari posisi sebuah nilai dalam array (indeks ke berapa) dan jumlah langkah yang diperlukan menggunakan dua metode, sequential search dan binary search.

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)
    }

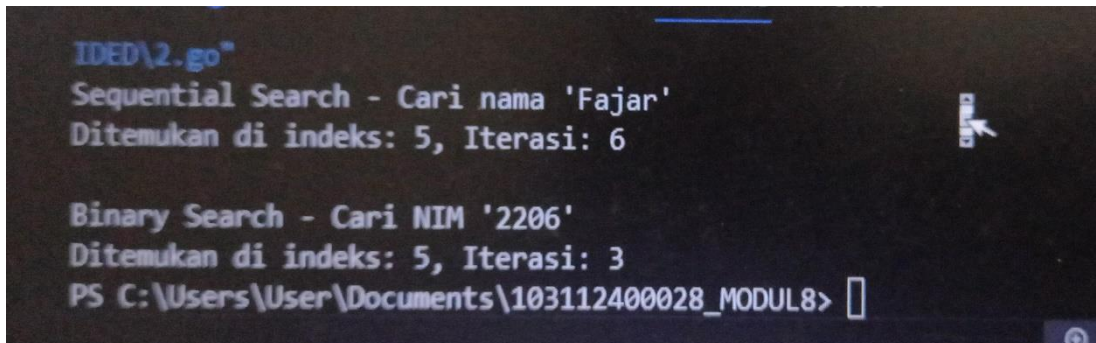
    // Urutkan data berdasarkan NIM untuk binary search
    sort.Slice(data[:n], func(i, j int) bool {
        return data[i].nim < data[j].nim
    })

    // 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 :



```

IDED\2.go"
Sequential Search - Cari nama 'Fajar'
Ditemukan di indeks: 5, Iterasi: 6

Binary Search - Cari NIM '2206'
Ditemukan di indeks: 5, Iterasi: 3
PS C:\Users\User\Documents\103112400028_MODUL8>

```

Penjelasan : Program ini bertujuan untuk mencari posisi data mahasiswa dalam array (slice) dan menghitung jumlah iterasi menggunakan dua metode, sequential search dan binary search.

III. UNGUIDED

1.) Source code

```
//Muhammad Gamel Al Ghifari
//103112400028
package main

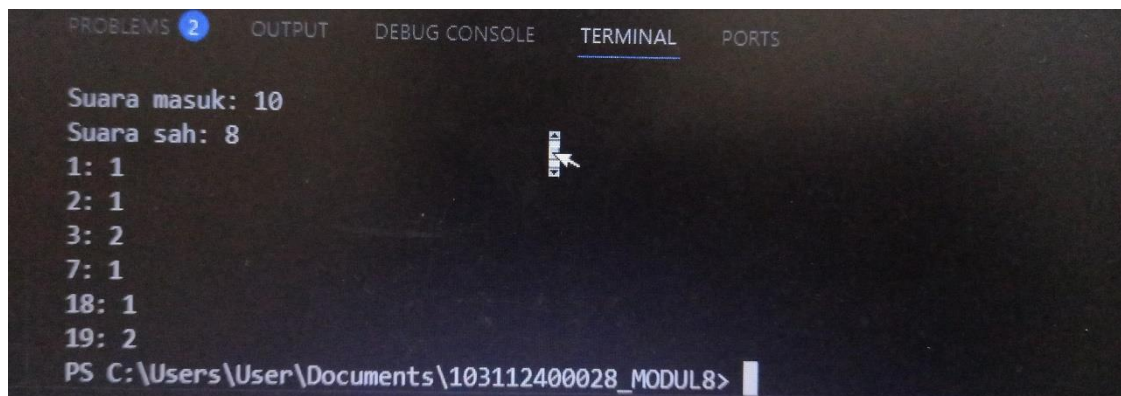
import (
    "fmt"
)

func main() {
    var n int
    vote := make(map[int]int)
    total, sah := 0, 0

    fmt.Println("Input suara (akhiri dengan 0):")
    for {
        fmt.Scan(&n)
        if n == 0 {
            break
        }
        total++
        if n >= 1 && n <= 20 {
            vote[n]++
            sah++
        }
    }

    fmt.Println("Suara masuk:", total)
    fmt.Println("Suara sah:", sah)
    for i := 1; i <= 20; i++ {
        if vote[i] > 0 {
            fmt.Printf("%d: %d\n", i, vote[i])
        }
    }
}
```

Output :

A screenshot of a Visual Studio Code terminal window. The terminal has tabs for PROBLEMS (with a blue circle containing the number 2), OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected and underlined), and PORTS. The output text in the terminal is as follows:

```
Suara masuk: 10
Suara sah: 8
1: 1
2: 1
3: 2
7: 1
18: 1
19: 2
PS C:\Users\User\Documents\103112400028_MODUL8>
```

A mouse cursor is visible over the terminal text.

Penjelasan : Program ini bertujuan untuk membaca, memvalidasi, dan menghitung suara yang diberikan dalam pemilihan ketua RT. Output berupa total suara masuk, total suara sah, dan sejumlah baris yang mencetak data para calon mana saja yang mendapatkan suara dan jumlah suara yang didapatkan masing masing calon.

2.) Source code

```
//Muhammad Gamel Al Ghifari
//103112400028
package main

import (
    "fmt"
    "sort"
)

func main() {
    var x, total, sah int
    v := make(map[int]int)

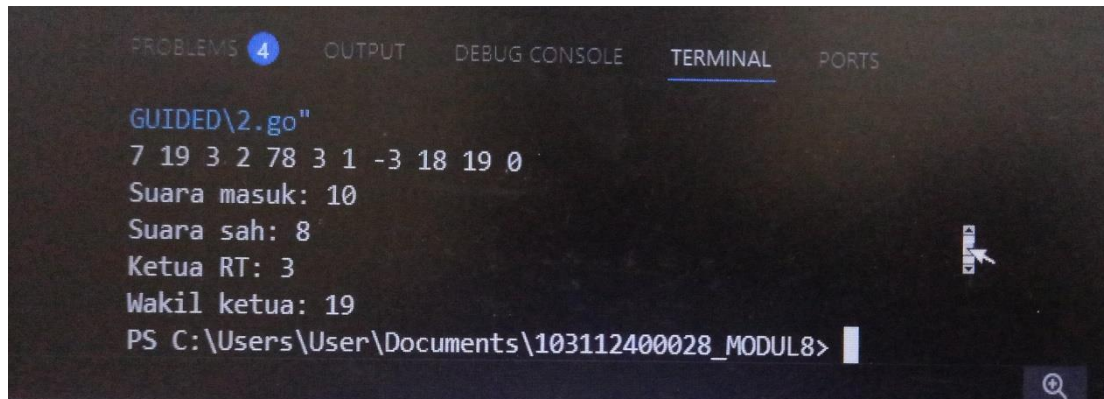
    for fmt.Scan(&x); x != 0; fmt.Scan(&x) {
        total++
        if x >= 1 && x <= 20 {
            v[x]++
            sah++
        }
    }

    type pair struct{ no, jml int }
    var data []pair
    for i := 1; i <= 20; i++ {
        if v[i] > 0 {
            data = append(data, pair{i, v[i]})
        }
    }

    sort.Slice(data, func(i, j int) bool {
        if data[i].jml == data[j].jml {
            return data[i].no < data[j].no
        }
        return data[i].jml > data[j].jml
    })

    fmt.Println("Suara masuk:", total)
    fmt.Println("Suara sah:", sah)
    if len(data) > 0 {
        fmt.Println("Ketua RT:", data[0].no)
    }
    if len(data) > 1 {
        fmt.Println("Wakil ketua:", data[1].no)
    }
}
```

Output :

A screenshot of a Go IDE's terminal window. The terminal has tabs for 'PROBLEMS' (with a blue circle containing the number 4), 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected and underlined), and 'PORTS'. The output text in the terminal is as follows:

```
GUIDED\2.go"  
7 19 3 2 78 3 1 -3 18 19 0  
Suara masuk: 10  
Suara sah: 8  
Ketua RT: 3  
Wakil ketua: 19  
PS C:\Users\User\Documents\103112400028_MODUL8>
```

Penjelasan : Program ini bertujuan untuk membaca, memvalidasi, dan menghitung suara yang diberikan dalam pemilihan ketua RT, mencari siapa pemenang pemilihan ketua RT. Program meminta input data yang berisi bilangan bulat yang kadang disisipi dengan data yang tidak valid. Output berupa total suara masuk, total suara sah, pasangan ketua RT dan wakil ketua.

3.) Source code

```
//Muhammad Gamel Al Ghifari
//103112400028
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(pos)
    } else {
        fmt.Println("TIDAK ADA")
    }
}

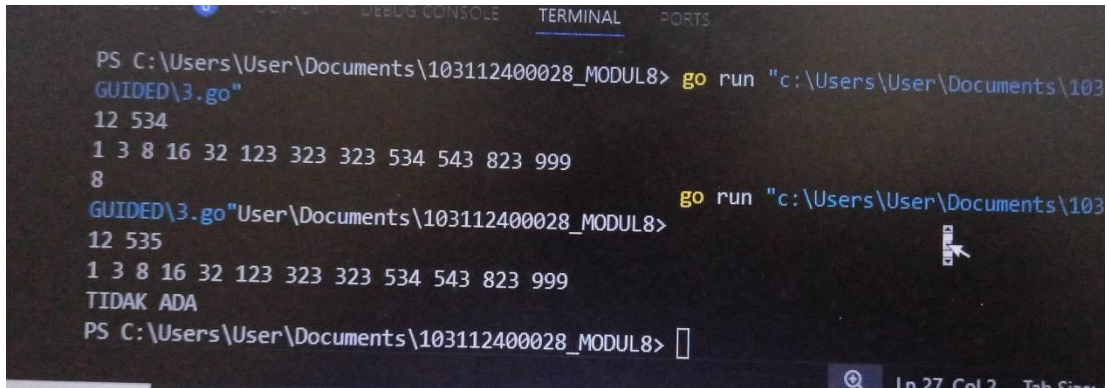
func isiArray(n int) {
    for i := 0; i < n; i++ {
        fmt.Scan(&data[i])
    }
}

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 :

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The prompt is 'PS C:\Users\User\Documents\103112400028_MODUL8>'. The user enters 'go run "c:\Users\User\Documents\103112400028_MODUL8\3.go"'. The program outputs '12 534', followed by a space-separated list of numbers: '1 3 8 16 32 123 323 323 534 543 823 999', and then '8'. The user enters 'go run "c:\Users\User\Documents\103112400028_MODUL8\3.go"'. The program outputs '12 535', followed by the same list of numbers, and then 'TIDAK ADA'. The prompt is 'PS C:\Users\User\Documents\103112400028_MODUL8>'.

```
PS C:\Users\User\Documents\103112400028_MODUL8> go run "c:\Users\User\Documents\103112400028_MODUL8\3.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
PS C:\Users\User\Documents\103112400028_MODUL8> go run "c:\Users\User\Documents\103112400028_MODUL8\3.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA
PS C:\Users\User\Documents\103112400028_MODUL8>
```

Penjelasan : Program ini bertujuan untuk menentukan apakah bilangan k ada di dalam array dan menentukan ada di indeks ke berapa bilangan k tersebut, serta mencetak pesan **TIDAK ADA** jika k tidak ditemukan. Program meminta input yang terdiri dari dua baris. Baris pertama berisi dua buah integer positif, yaitu n dan k menyatakan banyaknya data, dan baris kedua berisi n buah data integer positif yang sudah terurut membesar.

IV. KESIMPULAN

Modul ini membandingkan dua metode pencarian data, yaitu Sequential Search dan Binary Search. Hasilnya menunjukkan bahwa Binary Search lebih efisien karena membutuhkan langkah pencarian lebih sedikit, tetapi hanya dapat digunakan jika data telah diurutkan terlebih dahulu. Sedangkan Sequential Search dapat digunakan pada data yang tidak terurut, namun memerlukan lebih banyak langkah pencarian. Binary Search lebih efisien daripada Sequential Search, namun hanya dapat digunakan jika data sudah dalam keadaan terurut.

REFERENSI

Telkom University. (2025). *Modul Praktikum Algoritma dan Pemrograman 2*.