LAPORAN PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN 2

MODUL 8

PENCARIAN NILAI ACAK PADA HIMPUNAN DATA

Oleh:

DWI OKTA SURYANINGRUM

103112400066

12-IF-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

a. Pencarian Data dalam Struktur Array

Pencarian data merupakan salah satu operasi dasar dalam struktur data yang bertujuan untuk menemukan elemen tertentu berdasarkan kriteria tertentu, seperti nilai atau identitas unik (misalnya, nama, NIM, atau alamat). Terdapat berbagai macam teknik pencarian data, dua di antaranya yang umum digunakan adalah pencarian sekuensial (sequential search) dan pencarian biner (binary search). Kedua metode ini memiliki karakteristik dan kompleksitas yang berbeda, tergantung pada kondisi data (terurut atau tidak terurut) dan efisiensi yang dibutuhkan.

1) Pencarian Sekuensial (Sequential Search)

Pencarian sekuensial adalah metode pencarian sederhana yang dilakukan dengan memeriksa setiap elemen dalam array secara berurutan dari elemen pertama hingga elemen terakhir. Pencarian ini tidak memerlukan data dalam keadaan terurut.

Prinsip kerja algoritma pencarian sekuensial adalah sebagai berikut:

1. Diasumsikan terdapat array T dengan panjang N dan sebuah elemen X yang ingin dicari.
2. Proses pencarian dimulai dari indeks 0 hingga N-1, dan pada setiap langkah, elemen T[i] dibandingkan dengan X.
3. Jika ditemukan kesamaan antara T[i] dan X, maka pencarian dihentikan dan posisi elemen dikembalikan.
4. Jika elemen tidak ditemukan setelah semua elemen dicek, maka proses pencarian dinyatakan gagal.

Kelebihan dari pencarian ini adalah kemudahannya dalam implementasi dan tidak memerlukan prasyarat data terurut. Namun,

kelemahannya adalah efisiensi yang rendah pada data berukuran besar, dengan kompleksitas waktu O(n).

2) Pencarian Biner (Binary Search)

Pencarian biner merupakan algoritma pencarian yang lebih efisien dibandingkan pencarian sekuensial, namun memiliki syarat utama bahwa data harus dalam keadaan **terurut menaik (ascending)**. Algoritma ini bekerja dengan cara membagi ruang pencarian menjadi dua bagian pada setiap langkah.

Langkah-langkah algoritma binary search adalah sebagai berikut:

1. Tentukan indeks awal (kiri) dan indeks akhir (kanan) dari array.
2. Hitung indeks tengah (tengah) sebagai (kiri + kanan)/2.
3. Bandingkan nilai X dengan T[tengah]:
   - Jika T[tengah] == X, maka pencarian berhasil.
   - Jika T[tengah] < X, maka pencarian dilanjutkan ke sebelah kanan (indeks tengah + 1 sampai kanan).
   - Jika T[tengah] > X, maka pencarian dilanjutkan ke sebelah kiri (indeks kiri sampai tengah - 1).
4. Proses ini diulang hingga data ditemukan atau ruang pencarian habis.

Keunggulan dari binary search adalah efisiensinya yang tinggi pada data besar, dengan kompleksitas waktu O(log n). Namun, metode ini hanya dapat digunakan pada data yang sudah terurut.

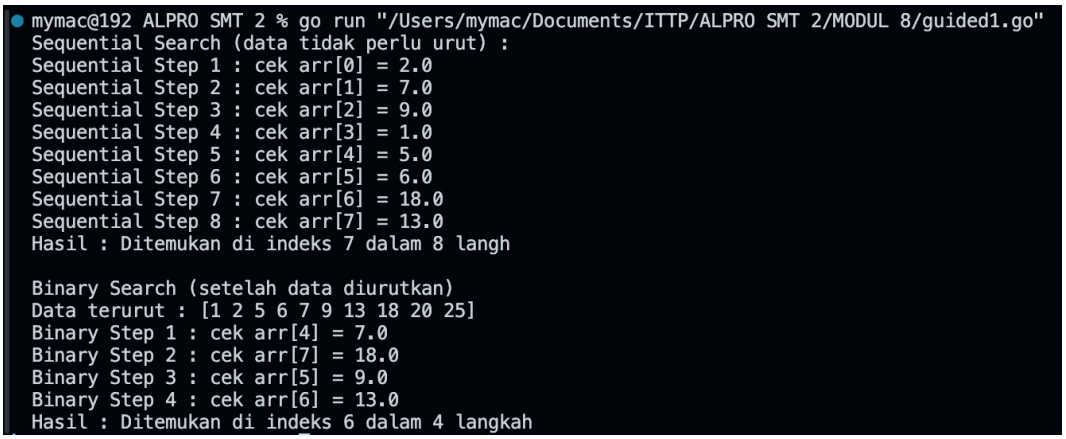
II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

a. Guided 1

```go
//  DWI OKTA SURYANINGRUM
//  103112400066

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() {
    // array awal
    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 langh\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)
    }
}
```

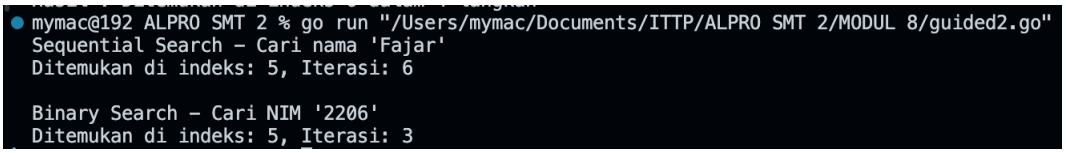
Output :


mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/guided1.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 langh
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

b. Guided 2

```go
//  DWI OKTA SURYANINGRUM
//  103112400066

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 :

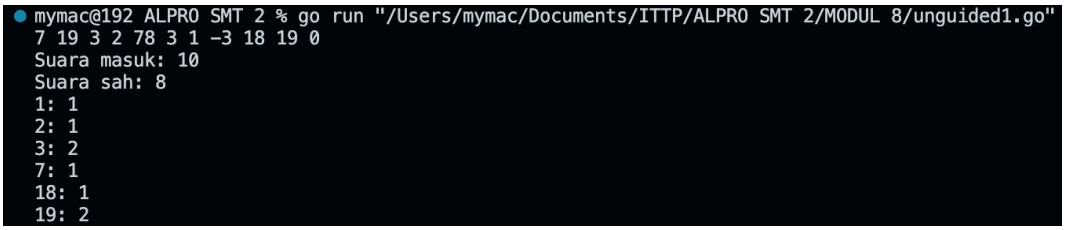

mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/guided2.go"
Sequential Search - Cari nama 'Fajar'
Ditemukan di indeks: 5, Iterasi: 6
Binary Search - Cari NIM '2206'
Ditemukan di indeks: 5, Iterasi: 3

III. UNGUIDED

a. Unguided 1

```go
//  DWI OKTA SURYANINGRUM
//  103112400066

package main

import (
    "fmt"
)

func main() {
    // map untuk nyimpen jumlah suara tiap kandidat dengan range (1-20)
    hitung := make(map[int]int)

    masuk := 0 // jumlah semua suara yang masuk
    sah := 0   // jumlah suara yang sah

    var input int
    for {
        fmt.Scan(&input)

        if input == 0 {
            break // kalau ketemu 0, berhenti baca input
        }

        masuk++ // hitung total suara masuk

        if input >= 1 && input <= 20 {
            hitung[input]++ // suara valid, masukin ke map
            sah++
        }
    }

    // menampilkan hasil akhir
    fmt.Println("Suara masuk:", masuk)
    fmt.Println("Suara sah:", sah)

    // cetak hasil masing-masing kandidat (yang dapet suara aja)
    for i := 1; i <= 20; i++ {
        if jumlah, ada := hitung[i]; ada {
            fmt.Printf("%d: %d\n", i, jumlah)
        }
    }
}

```

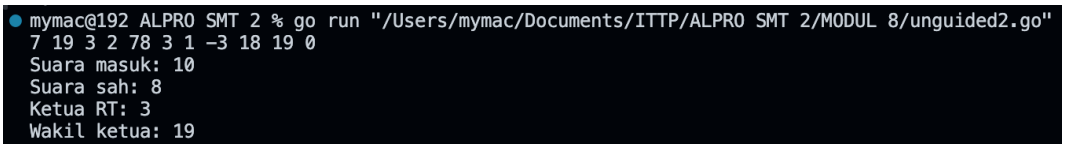
Output :


mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/unguided1.go"
7 19 3 2 78 3 1 -3 18 19 0
Suara masuk: 10
Suara sah: 8
1: 1
2: 1
3: 2
7: 1
18: 1
19: 2

b. Unguided 2

```go
//  DWI OKTA SURYANINGRUM
//  103112400066

package main

import (
    "fmt"
    "sort"
)

func main() {
    // map buat nyimpen jumlah suara untuk masing-masing kandidat
    suara := make(map[int]int)

    totalMasuk := 0 // semua input yang masuk (selain 0)
    suaraSah := 0   // suara yang valid (antara 1-20)

    var input int
    for {
        fmt.Scan(&input)

        if input == 0 {
            break // 0 artinya selesai input
        }

        totalMasuk++

        if input >= 1 && input <= 20 {
            suara[input]++ // menghitung suara kandidat
            suaraSah++
        }
    }

    // menampilkan ringkasan suara
    fmt.Println("Suara masuk:", totalMasuk)
    fmt.Println("Suara sah:", suaraSah)

    // struct buat nampung pasangan kandidat dan jumlah suara
    type Pasangan struct {
        nomor  int
        jumlah int
    }

    var hasil []Pasangan

    // memasukkan ke slice untuk bisa diurutkan
    for nomor, jumlah := range suara {
        hasil = append(hasil, Pasangan{nomor, jumlah})
    }

    // mengurutkan berdasarkan jumlah suara terbanyak, lalu nomor terkecil
    sort.Slice(hasil, func(i, j int) bool {
        if hasil[i].jumlah == hasil[j].jumlah {
            return hasil[i].nomor < hasil[j].nomor
        }
        return hasil[i].jumlah > hasil[j].jumlah
    })

    // memastikan minimal ada 2 kandidat untuk ditampilkan
    if len(hasil) >= 2 {
        fmt.Println("Ketua RT:", hasil[0].nomor)
        fmt.Println("Wakil ketua:", hasil[1].nomor)
    } else if len(hasil) == 1 {
        fmt.Println("Ketua RT:", hasil[0].nomor)
        fmt.Println("Wakil ketua: TIDAK ADA")
    } else {
        fmt.Println("TIDAK ADA suara sah yang masuk.")
    }
}
```

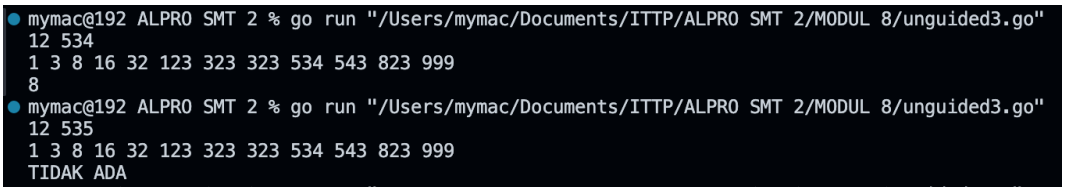
Output :


mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/unguided2.go"
7 19 3 2 78 3 1 -3 18 19 0
Suara masuk: 10
Suara sah: 8
Ketua RT: 3
Wakil ketua: 19

c. Unguided 3

```go
//  DWI OKTA SURYANINGRUM
//  103112400066

package main

import (
    "fmt"
)

const NMAX = 1000000

// arrau global untuk menyimpan data
var data [NMAX]int


func main() {
    var n, k int

    // membaca n (jumlah data) dan k (angka yang ingin dicari)
    fmt.Scan(&n, &k)

    // memanggil fungsi isiArray untuk mengisi array data sebanyak n angka
    isiArray(n)

    // mencari posisi k dalam array
    pos := posisi(n, k)

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

// fungsi unruk membaca n data dan menyimpannya ke dalam array
func isiArray(n int) {
    for i := 0; i < n; i++ {
        fmt.Scan(&data[i]) // Baca satu per satu dan simpan ke array
    }
}

// fungsi untuk mencari posisi k dalam array data
func posisi(n, k int) int {
    // kareba data sudah terurut, kita bisa pakai binary search
    low := 0
    high := n - 1

    for low <= high {
        mid := (low + high) / 2

        if data[mid] == k {
            return mid // Ketemu, langsung return posisi
        } else if data[mid] < k {
            low = mid + 1 // Cari di kanan
        } else {
            high = mid - 1 // Cari di kiri
        }
    }

    return -1 // Kalau sampai sini, berarti tidak ketemu
}

```

Output :


mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/unguided3.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
mymac@192 ALPRO SMT 2 % go run "/Users/mymac/Documents/ITTP/ALPRO SMT 2/MODUL 8/unguided3.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA

IV. KESIMPULAN

pencarian data dalam struktur array dapat dilakukan menggunakan dua metode utama: pencarian sekuensial dan pencarian biner. Pencarian sekuensial mudah diimplementasikan dan tidak memerlukan data terurut, tetapi kurang efisien untuk data berukuran besar dengan kompleksitas O(n). Di sisi lain, pencarian biner lebih efisien dengan kompleksitas O(log n), namun hanya dapat digunakan pada data yang sudah terurut. Pemilihan metode pencarian bergantung pada kondisi data dan kebutuhan efisiensi.