LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL XI

PENCARIAN NILAI ACAK PADA HIMPUNAN DATA

Oleh :

NAMA : Felix Pedrosa Valentino

NIM : 103112400056

KELAS : IF – 12 – 01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

Sequential Search

Pencarian secara berurutan ini melibatkan pemeriksaan data mulai dari yang pertama hingga terakhir satu per satu. Karakter utama dari metode ini adalah pencarian akan selesai ketika item yang dicari ditemukan, meskipun masih ada elemen yang belum diperiksa. Algoritma ini disebut Pencarian Berurutan, karena dalam prosesnya, setiap elemen dalam array diperiksa satu demi satu secara berurutan dari elemen awal hingga item yang dicari atau sampai elemen terakhir.

1) Asumsikan ada sebuah array integer T dengan indeks dari 0 sampai N-1, serta sebuah nilai yang dicari dalam array T, yaitu X.

2) Status pencarian digunakan untuk menunjukkan apakah item yang dicari sudah ditemukan atau belum, misalnya dengan menggunakan tipe boolean bernama found.

3) Pencarian dilakukan dari T[0] hingga T[N-1], di mana setiap kali ada perbandingan dengan X, nilai found akan diperbarui.

4) Proses pencarian harus dihentikan jika status found bernilai true (item ditemukan) atau jika T[N-1] sudah diperiksa.

Binary Search

Algoritma ini adalah: (dengan asumsi data telah diurutkan secara meningkat, di mana data dengan indeks lebih rendah terletak di "kiri" dan indeks yang lebih tinggi di "kanan")

1) Pilih salah satu elemen dari kumpulan data yang ada; algoritma ini akan menggunakan rentang dari kiri kr hingga kanan kn. Untuk kemudahan dan beberapa alasan lainnya, biasanya elemen yang dipilih adalah yang berada di tengah rentang tersebut.

2) Jika elemen yang dipilih ternyata terlalu kecil, maka geser rentang data ke kanan dari posisi elemen tersebut. Hal ini karena jika elemen yang diambil terlalu kecil, semua elemen di sebelah kiri juga pasti akan lebih kecil dari nilai yang dicari.

3) Sebaliknya, jika elemen yang dipilih terlalu besar, tindakan serupa harus diambil.

Sebagai informasi, algoritma pencarian biner untuk array terurut membesar (*ascending*) akan berbeda dengan terurut mengecil (*descending*), sehingga algoritma ini tidak dapat berfungsi jika urutannya terbalik. Contohnya, jika array diatur dalam urutan menaik tetapi algoritma pencarian biner yang digunakan adalah untuk array yang diurutkan menurun, maka pencarian dengan metode biner tidak akan berhasil.

Pencarian pada Array Bertipe Data Struct

Metode pencarian dalam array dengan jenis data struktural hampir serupa dengan tipe data dasar. Yang perlu ditambahkan hanyalah field kategori untuk pencarian tersebut. Untuk algoritma binary search, penting agar array terurut sesuai dengan kategori pencarian. Contohnya, jika pencarian dilakukan berdasarkan NIM mahasiswa, maka algoritma binary search hanya dapat digunakan jika array diurutkan menurut NIM. Jika array sudah diurutkan berdasarkan NIM mahasiswa tetapi pencariannya dilakukan berdasarkan nama atau kategori lainnya, algoritma binary search tidak akan berfungsi. Oleh karena itu, array harus diurutkan berdasarkan NIM terlebih dahulu atau dapat menggunakan algoritma pencarian berurutan.

II. GUIDED

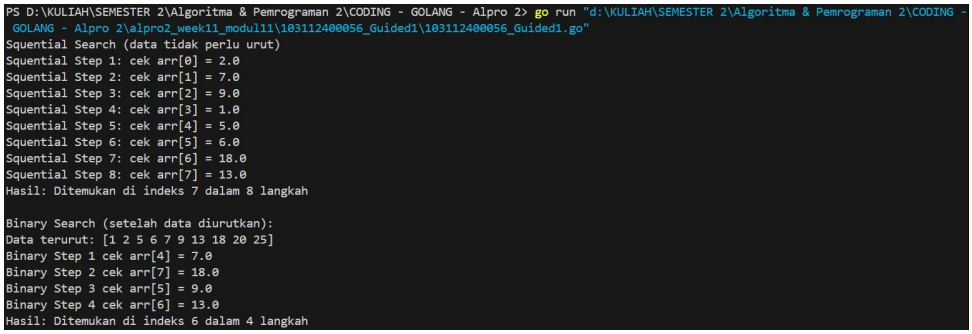
1. Guided 1

Source Code :

```
// Felix Pedrosa V

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 :



Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan bertujuan untuk melakukan pencarian nilai dalam sebuah array dengan memanfaatkan dua metode, yaitu Sequential Search dan Binary Search. Pertama, program ini mendefinisikan fungsi `sequentialSearch` yang mencari nilai target dalam array yang tidak terurut, sambil mencatat setiap langkah dan jumlah iterasi yang dilakukan selama proses pencarian. Selanjutnya, fungsi `binarySearch` digunakan untuk mencari nilai target dalam array yang sudah diurutkan, juga dengan mencatat langkah-langkah serta jumlah iterasi.

Dalam fungsi `main`, program menginisialisasi sebuah array berisi nilai float64 dan menetapkan nilai target yang akan dicari. Setelah melakukan pencarian menggunakan metode Sequential Search, hasilnya akan ditampilkan. Kemudian, proses pengurutan data dilakukan sebelum melanjutkan pencarian dengan metode Binary Search. Hasil dari kedua metode pencarian, termasuk indeks dan jumlah langkah yang diperlukan, ditampilkan di layar, memberikan gambaran yang jelas mengenai efisiensi masing-masing metode.

2. Guided 2

Source Code :

```
// Felix Pedrosa V

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 :

```
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -
 GOLANG - Alpro 2\alpro2_week11_modul11\103112400056_Guided2\103112400056_Guided2.go"
Sequential Search - Cari nama 'Fajar'
Ditemukan di indeks: 5, Iterasi: 6

Binary Search - Cari NIM '2206'
Ditemukan di indeks: 5, Iterasi: 3
```

Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan bertujuan untuk mengelola data mahasiswa serta melakukan pencarian data dengan dua metode, yaitu Sequential Search dan Binary Search. Di dalamnya terdapat struktur data mahasiswa yang menyimpan informasi penting seperti nama, NIM, kelas, jurusan, dan IPK.

Program ini menginisialisasi sebuah array bernama arrMhs yang mampu menampung hingga 2023 mahasiswa, dan mengisi data mahasiswa

secara manual. Pencarian dilakukan dengan metode Sequential Search berdasarkan nama mahasiswa, serta Binary Search berdasarkan NIM setelah data diurutkan terlebih dahulu. Hasil dari pencarian ini mencakup indeks mahasiswa yang ditemukan beserta jumlah iterasi yang dilakukan selama proses pencarian. Selain itu, program ini juga menampilkan hasil pencarian di layar, memberikan informasi apakah data yang dicari berhasil ditemukan atau tidak.

III. UNGUIDED

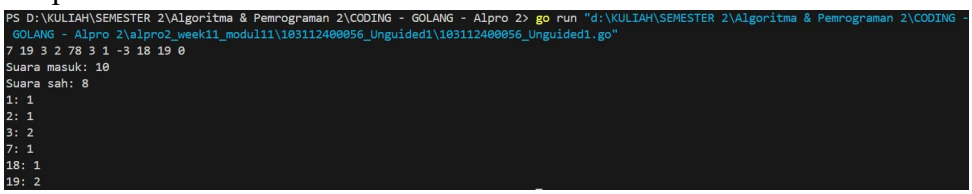
1. UnGuided 1

Source Code :

```
// Felix Pedrosa V

package main

import (
    "fmt"
    "sort"
)

// Fungsi pencarian biner sederhana
func cariIndex(data []int, cari int) bool {
    kiri := 0
    kanan := len(data) - 1
    for kiri <= kanan {
        tengah := (kiri + kanan) / 2
        if data[tengah] == cari {
            return true
        } else if data[tengah] < cari {
            kiri = tengah + 1
        } else {
            kanan = tengah - 1
        }
    }
    return false
}

// Fungsi pengecekan apakah input suara sah
func isValid(suara int, daftar []int) bool {
    return cariIndex(daftar, suara)
}

func main() {
    var suara int
    kandidatSah := make([]int, 20)
    for i := range kandidatSah {
        kandidatSah[i] = i + 1
    }
```

```
        sort.Ints(kandidatSah)

        hitungSuara := make([]int, 21)
        jumlahMasukan := 0
        jumlahSah := 0



        for {
            _, err := fmt.Scan(&suara)
            if err != nil {
                break
            }
            if suara == 0 {
                break
            }
            jumlahMasukan++
            if isValid(suara, kandidatSah) {
                hitungSuara[suara]++
                jumlahSah++
            }
        }

        fmt.Printf("Suara masuk: %d\n", jumlahMasukan)
        fmt.Printf("Suara sah: %d\n", jumlahSah)

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

Output :



Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan bertujuan untuk menghitung dan memverifikasi suara dalam pemilihan ketua RT yang melibatkan 20 kandidat ketua. Dengan menerapkan metode pencarian biner, program akan memeriksa apakah nomor suara yang dimasukkan oleh warga termasuk dalam kategori nominasi yang benar, yaitu bilangan bulat dari 1 hingga 20. Warga diminta untuk memasukkan nomor suara secara terpisah, dan penginputan akan berakhir dengan angka 0. Setelah itu, program akan menunjukkan total suara yang diterima, jumlah suara yang sah, serta rincian mengenai berapa banyak suara yang diperoleh oleh masing-masing kandidat ketua.

2. UnGuided 2

Source Code :

```
// Felix Pedrosa V

package main

import (
    "fmt"
    "sort"
)

type Kandidat struct {
    Nomor int
    Suara int
}

func main() {
    var suara int
    jumlahMasukan := 0
    jumlahSah := 0

    hitungSuara := make(map[int]int)

    for {
        _, err := fmt.Scan(&suara)
        if err != nil {
```

```
            break
        }
        if suara == 0 {
            break
        }
        jumlahMasukan++

        if suara >= 1 && suara <= 20 {
            hitungSuara[suara]++
            jumlahSah++
        }
    }

    // Buat slice dari hasil untuk disortir
    var hasil []Kandidat
    for nomor, jumlah := range hitungSuara {
        hasil = append(hasil, Kandidat{Nomor: nomor, Suara:
jumlah})
    }

    // Urutkan berdasarkan suara terbanyak, jika sama urutkan
berdasarkan nomor terkecil
    sort.Slice(hasil, func(i, j int) bool {
        if hasil[i].Suara == hasil[j].Suara {
            return hasil[i].Nomor < hasil[j].Nomor
        }
        return hasil[i].Suara > hasil[j].Suara
    })

    // Tampilkan hasil
    fmt.Printf("Suara masuk: %d\n", jumlahMasukan)
    fmt.Printf("Suara sah: %d\n", jumlahSah)

    if len(hasil) == 0 {
        fmt.Println("Tidak ada suara sah, tidak ada ketua dan wakil.")
    } else if len(hasil) == 1 {
        fmt.Printf("Ketua RT: %d\n", hasil[0].Nomor)
        fmt.Println("Wakil Ketua: Tidak tersedia (hanya satu calon
sah).")
    } else {
```

```
            fmt.Printf("Ketua RT: %d\n", hasil[0].Nomor)
            fmt.Printf("Wakil Ketua: %d\n", hasil[1].Nomor)
        }
    }
```

Output :

```
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -
 GOLANG - Alpro 2\alpro2_week11_modul11\103112400056_Unguided2\103112400056_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
```

Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan bertujuan untuk menghitung dan menganalisis suara dalam pemilihan ketua RT yang diikuti oleh 20 kandidat. Pengguna dapat memasukkan suara dalam bentuk bilangan bulat, di mana nilai yang diterima berkisar antara 1 hingga 20, dan proses akan selesai ketika angka 0 dimasukkan. Setelah semua suara terkumpul, program akan menghitung jumlah total suara yang diterima serta suara yang sah. Suara yang sah kemudian akan diatur dan diurutkan berdasarkan jumlah suara terbanyak, dan jika ada kandidat dengan jumlah suara yang sama, mereka akan diurutkan menurut nomor urut. Hasil pemilihan akan ditampilkan, mencakup siapa yang terpilih sebagai ketua RT dan wakilnya, atau menyatakan jika tidak ada suara yang sah.

3. UnGuided 3

Source Code :

```
// Felix Pedrosa V

package main

import "fmt"

const NMAX = 1000000
var arrayData [NMAX]int

func main() {
    var jumlah, target int
    fmt.Scan(&jumlah, &target)
```

```
    isiData(jumlah)
    hasil := cariPosisi(jumlah, target)

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

func isiData(jml int) {
    for i := 0; i < jml; i++ {
        fmt.Scan(&arrayData[i])
    }
}

func cariPosisi(jml, cari int) int {
    kiri := 0
    kanan := jml - 1

    for kiri <= kanan {
        tengah := (kiri + kanan) / 2
        if arrayData[tengah] == cari {
            return tengah
        } else if arrayData[tengah] < cari {
            kiri = tengah + 1
        } else {
            kanan = tengah - 1
        }
    }
    return -1
}
```

Output :

```
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2\alpro2_week11_modul11\103112400056_Unguided3\103112400056_Unguided3.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2\alpro2_week11_modul11\103112400056_Unguided3\103112400056_Unguided3.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA
```

Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan bertujuan untuk membantu dalam proses pemilihan ketua RT dengan membaca dan menghitung suara yang diberikan oleh masyarakat serta memvalidasinya. Pengguna diminta untuk memasukkan serangkaian bilangan bulat yang mewakili suara untuk calon ketua RT, di mana angka yang diterima berada dalam kisaran 1 hingga 20. Program ini menerapkan cara pengisian data dan pencarian posisi melalui metode pencarian biner. Setelah menerima suara, program akan melakukan pemeriksaan untuk memastikan bahwa semua suara yang dimasukkan berada dalam batas yang sesuai. Jika tidak ada suara yang valid, akan muncul pesan "TIDAK ADA". Sebaliknya, program akan menunjukkan posisi suara dari calon ketua RT yang terpilih serta total suara yang sah.

IV. KESIMPULAN

Kesimpulan dari modul ini mengenai pencarian nilai acak dalam kumpulan data menguraikan dua algoritma utama, yaitu Pemindaian Berurutan dan Pemindaian Biner, yang digunakan untuk menemukan data dalam array. Pemindaian Berurutan adalah teknik yang mencari elemen secara bertahap dari awal hingga akhir, dan proses akan berhenti begitu data yang diinginkan ditemukan. Metode ini bisa digunakan pada data yang tidak teratur, meskipun kurang efisien jika diterapkan pada dataset besar. Sementara itu, Pemindaian Biner membutuhkan data yang teratur dan melakukan pencarian dengan membagi rentang yang dicari menjadi dua bagian, sehingga lebih efisien dibandingkan Pemindaian Berurutan. Selain itu, modul ini juga membahas pencarian dalam array yang menggunakan tipe data struct, yang memerlukan keteraturan berdasarkan kategori pencarian. Dengan pemahaman yang jelas tentang kedua algoritma ini, pengguna dapat memilih metode pencarian yang paling tepat sesuai dengan kondisi data yang dimiliki.

V. REFERENSI

Modul 11 - Praktikum Alpro 2