

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 11
PENCARIAN NILAI ACAK PADA HIMPUNAN DATA**



Oleh:

MUHAMMAD FAUZAN

103112400064

12 IF 01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2025

I. DASAR TEORI

11.1 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 - 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.

11.2 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.

Algoritma ini dikenal dengan nama Binary Search.

11.3 Pencarian pada Array Bertipe Data Struct

Algoritma pencarian pada array bertipe data struct tidak berbeda jauh dengan tipe data dasar, kita hanya perlu menambahkan field kategori dari pencarian yang akan dilakukan. Khusus untuk algoritma binary search, maka keterurutan array harus sama dengan field kategori dari pencariannya. Misalnya apabila pencarian berdasarkan nim mahasiswa, maka algoritma binary search hanya akan bisa digunakan apabila array terurut berdasarkan nim. Algoritma binary search tidak akan berjalan apabila array terurut berdasarkan nim mahasiswa, tetapi pencarian yang dilakukan berdasarkan field nama atau kategori yang lain selain nim. Solusinya adalah array harus diurutkan berdasarkan nim terlebih dahulu, atau gunakan algoritma sequential search.

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

Contoh 1

```
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]=%.f\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 stop %d: cek arr[%d]= %.f\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

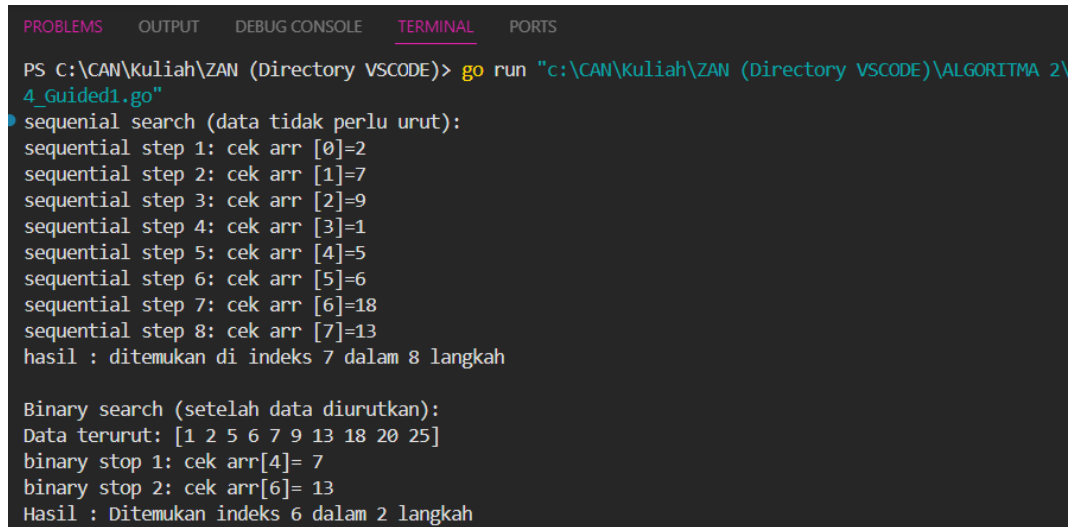
    fmt.Println("sequential search (data tidak perlu urut):")
    idxseq, iterseq := sequentialSearch(data, target)
    if idxseq != -1 {
        fmt.Printf("hasil : ditemukan di indeks %d dalam %d
langkah\n\n", idxseq, iterseq)
    } else {
        fmt.Printf("hasil : tidak ditemukan setelah %d langkah\n", iterseq)
    }

    //Binary earch array diurutkan
    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 indeks %d dalam %d langkah\n",
idxBin, iterBin)
    } else {
        fmt.Printf("Hasil : Tidak ditemukan Setelah %d langkah\n",
iterBin)
    }
}

```

Screenshots Output



```
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\4 Guided1.go"
sequential search (data tidak perlu urut):
sequential step 1: cek arr [0]=2
sequential step 2: cek arr [1]=7
sequential step 3: cek arr [2]=9
sequential step 4: cek arr [3]=1
sequential step 5: cek arr [4]=5
sequential step 6: cek arr [5]=6
sequential step 7: cek arr [6]=18
sequential step 8: cek arr [7]=13
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 stop 1: cek arr[4]= 7
binary stop 2: cek arr[6]= 13
Hasil : Ditemukan indeks 6 dalam 2 langkah
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini membandingkan dua metode pencarian data dalam array, yaitu sequential search dan binary search. Sequential search dilakukan langsung pada data acak, sementara binary search dilakukan setelah data diurutkan.

Program menampilkan langkah-langkah pencarian dan hasil berupa indeks serta jumlah langkah pencarian untuk masing-masing metode.

Contoh 2

```
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
})

```



```

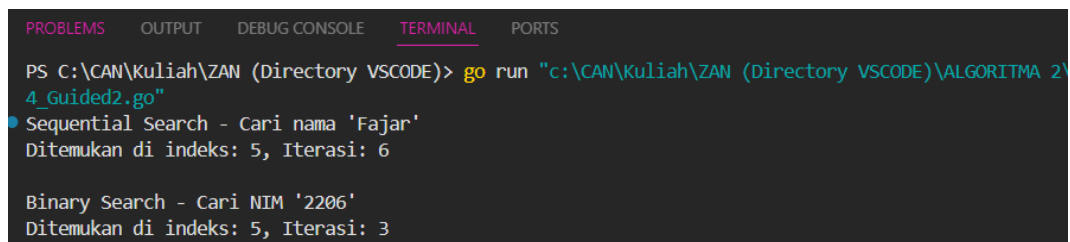
})

// 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)
}
}

```

Screenshots Output



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\4_Guided2.go"
Sequential Search - Cari nama 'Fajar'
Ditemukan di indeks: 5, Iterasi: 6

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

```

// Foto hasil dari menjalankan code

Deskripsi: Program ini melakukan pencarian data mahasiswa menggunakan dua metode: **sequential search** berdasarkan *nama* dan **binary search** berdasarkan *NIM*. Data mahasiswa dimasukkan secara manual, lalu sequential search mencari nama secara linear, sedangkan binary search membutuhkan data diurutkan berdasarkan NIM terlebih dahulu. Hasil pencarian menampilkan indeks lokasi data ditemukan dan jumlah iterasi yang dibutuhkan oleh masing-masing metode.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

Soal 1

```
// MUHAMMAD FAUZAN
// 103112400064
package main

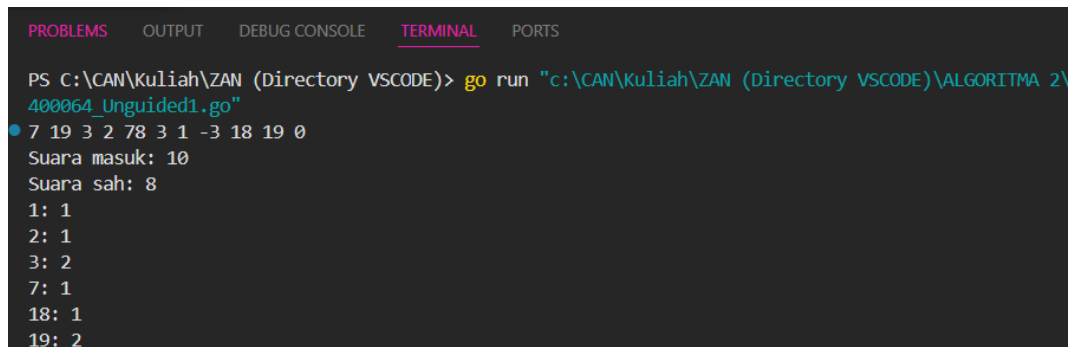
import "fmt"

func main() {
    const N = 20
    var suara [N + 1]int
    var input, total, valid int

    for {
        fmt.Scan(&input)
        if input == 0 {
            break
        }
        total++
        if input >= 1 && input <= N {
            suara[input]++
            valid++
        }
    }

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

Screenshots Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\400064_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
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini digunakan untuk menghitung hasil pemilihan ketua RT berdasarkan suara yang diberikan oleh warga dalam bentuk angka. Setiap suara divalidasi agar hanya angka dari 1 hingga 20 yang dianggap sah, sementara angka di luar rentang tersebut diabaikan. Program mencatat jumlah total suara yang masuk, jumlah suara yang sah, serta mencetak jumlah suara yang diperoleh masing-masing calon yang mendapat suara.

Soal 2

```
// MUHAMMAD FAUZAN
// 103112400064
package main

import "fmt"

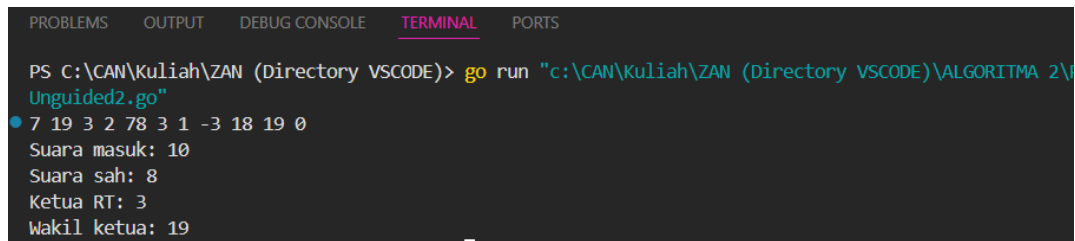
func main() {
    const N = 20
    var suara [N + 1]int
    var input, total, valid int
    for {
        fmt.Scan(&input)
        if input == 0 {
            break
        }
        total++
        if input >= 1 && input <= N {
            suara[input]++
            valid++
        }
    }

    fmt.Printf("Suara masuk: %d\n", total)
    fmt.Printf("Suara sah: %d\n", valid)

    var ketua, wakil int
    for i := 1; i <= N; i++ {
        if suara[i] > suara[ketua] || (suara[i] == suara[ketua] && i <
ketua) {
            wakil = ketua
            ketua = i
        } else if suara[i] > suara[wakil] || (suara[i] == suara[wakil] && i <
wakil && i != ketua) {
            wakil = i
        }
    }

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

Screenshots Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\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
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini merupakan pengembangan dari soal sebelumnya, yang tidak hanya menghitung suara sah dari para calon ketua RT, tetapi juga menentukan siapa yang menjadi ketua dan wakil ketua RT. Ketua ditentukan berdasarkan suara terbanyak, dan jika ada lebih dari satu calon dengan jumlah suara terbanyak yang sama, maka calon dengan nomor peserta lebih kecil akan dipilih.

Soal 3

```
// MUHAMMAD FAUZAN
// 103112400064
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 {
    kr := 0
    kn := n - 1

    for kr <= kn {
        mid := (kr + kn) / 2
        if data[mid] == k {
            return mid
        } else if data[mid] < k {
            kr = mid + 1
        } else {
            kn = mid - 1
        }
    }
    return -1
}
```

Screenshots Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\
Unguided3.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\
Unguided3.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini digunakan untuk mencari posisi suatu bilangan k dalam sekumpulan data yang sudah terurut membesar menggunakan algoritma pencarian biner (binary search). Program menerima jumlah data dan bilangan yang dicari, kemudian membaca data secara efisien. Jika bilangan ditemukan dalam array, program mencetak posisi (indeks) kemunculannya, jika tidak ditemukan maka program mencetak "TIDAK ADA".

IV. KESIMPULAN

V. REFERENSI

Modul Praktikum Algoritma dan Pemrograman 2. (2025). *Modul 11: Pencarian Nilai Acak Pada Himpunan Data*. Fakultas Informatika, Telkom University.