

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



**Oleh:**

**CINTA HERTANTIARA BINTANG**

**2311102287**

**IF 11 07**

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO**

**2024**

## 1. DASAR TEORI

### A. Sequential Search

Pencarian secara sekuensial adalah metode pencarian yang dilakukan dengan memeriksa setiap elemen dalam array satu per satu, mulai dari elemen pertama hingga terakhir. Proses pencarian ini akan berhenti ketika elemen yang dicari ditemukan, meskipun masih ada elemen yang belum diperiksa. Algoritma ini dikenal dengan nama Sequential Search karena pengecekan dilakukan secara berurutan dari data pertama hingga ditemukan atau mencapai data terakhir.

Langkah-langkah pada algoritma Sequential Search:

1. Asumsikan terdapat array integer  $T$  dengan indeks dari  $0$  hingga  $N-1$ , dan suatu nilai yang dicari, yaitu  $X$ .
2. Status pencarian digunakan untuk menandakan apakah data yang dicari ditemukan atau tidak, misalnya dengan variabel `found` bertipe boolean.
3. Pencarian dimulai dari  $T[0]$  hingga  $T[N-1]$ , setiap kali melakukan perbandingan dengan  $X$ , dan memperbarui status `found` jika ditemukan nilai yang sesuai.
4. Proses pencarian dihentikan apabila status `found` bernilai `true` (data ditemukan) atau jika sudah memeriksa semua elemen hingga  $T[N-1]$ .

### B. Binary Search

Binary Search adalah algoritma pencarian yang lebih efisien dibandingkan dengan pencarian sekuensial, tetapi hanya dapat digunakan pada array yang sudah terurut. Algoritma ini bekerja dengan membagi array menjadi dua bagian yang lebih kecil dan memeriksa elemen tengah. Jika elemen yang dicari lebih kecil atau lebih besar dari elemen tengah, pencarian dilanjutkan pada bagian yang sesuai.

Langkah-langkah pada algoritma Binary Search:

1. Ambil elemen tengah dari rentang data yang ada.
2. Jika elemen tengah lebih kecil dari nilai yang dicari, ubah rentang pencarian ke bagian kanan (di atas elemen tengah). Hal ini karena jika elemen tengah terlalu kecil, semua elemen di sebelah kiri juga akan terlalu kecil untuk dicari.
3. Jika elemen tengah lebih besar dari nilai yang dicari, ubah rentang pencarian ke bagian kiri (di bawah elemen tengah).
4. Proses ini berulang hingga ditemukan elemen yang dicari atau rentang pencarian tidak ada lagi.

Catatan: Algoritma Binary Search hanya berfungsi jika array terurut secara menaik (ascending). Jika array terurut secara menurun (descending), algoritma ini tidak akan berhasil dan perlu disesuaikan agar pencarian tetap dapat dilakukan dengan benar.

### **C. Pencarian pada Array Bertipe Data Struct**

Pencarian pada array yang berisi tipe data struktural (misalnya objek atau struktur data) tidak berbeda jauh dari tipe data dasar, namun kita perlu memperhatikan bahwa pencarian biasanya dilakukan berdasarkan kategori tertentu dalam struktur tersebut. Untuk algoritma Binary Search, array harus terurut berdasarkan kategori yang digunakan untuk pencarian.

Misalnya, jika kita ingin mencari berdasarkan NIM mahasiswa, maka array harus diurutkan terlebih dahulu berdasarkan NIM untuk menggunakan Binary Search. Jika array terurut berdasarkan kategori lain, seperti nama mahasiswa, maka Binary Search tidak dapat digunakan. Sebagai solusi, array dapat diurutkan terlebih dahulu sesuai dengan kategori yang diinginkan, atau alternatif lainnya adalah menggunakan algoritma Sequential Search yang lebih fleksibel dan dapat diterapkan pada array dengan berbagai kategori pencarian tanpa memerlukan urutan khusus.

## 2. GUIDED

### 1. Source Code

```
package main

import (
    "fmt"
)

// Definisi struct mahasiswa
type mahasiswa struct {
    nama      string
    nim       string
    kelas     string
    jurusan   string
    ipk       float64
}

// Definisi array bertipe struct mahasiswa
type arrMhs [2023]mahasiswa

// Sequential Search berdasarkan nama
func SeqSearch_3(T arrMhs, n int, X string) int {
    /* Mengembalikan indeks mahasiswa dengan nama X,
       atau -1 apabila tidak ditemukan pada array T
       yang berisi n data mahasiswa */
    var found int = -1
    var j int = 0
    for j < n && found == -1 {
        if T[j].nama == X {
            found = j
        }
    }
}
```

```

    }

    j = j + 1

}

return found
}

// Binary Search berdasarkan nim

func BinarySearch_3(T arrMhs, n int, X string) int {
    /* Mengembalikan indeks mahasiswa dengan nim X,
       atau -1 apabila tidak ditemukan pada array T
       yang berisi n data mahasiswa dan terurut membesar
       berdasarkan nim */

    var found int = -1

    var med int

    var kr int = 0

    var kn int = n - 1

    for kr <= kn && found == -1 {
        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
}

```

```
// Fungsi utama untuk demonstrasi

func main() {

    var data arrMhs

    var n int

    // Input jumlah mahasiswa

    fmt.Print("Masukkan jumlah mahasiswa: ")

    fmt.Scanln(&n)

    // Input data mahasiswa

    fmt.Println("Masukkan data mahasiswa:")

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

        fmt.Printf("Mahasiswa %d\n", i+1)

        fmt.Print("Nama: ")

        fmt.Scanln(&data[i].nama)

        fmt.Print("NIM: ")

        fmt.Scanln(&data[i].nim)

        fmt.Print("Kelas: ")

        fmt.Scanln(&data[i].kelas)

        fmt.Print("Jurusan: ")

        fmt.Scanln(&data[i].jurusan)

        fmt.Print("IPK: ")

        fmt.Scanln(&data[i].ipk)

    }

    // Pencarian berdasarkan nama

    var cariNama string

    fmt.Print("Masukkan nama mahasiswa yang ingin dicari: ")

    fmt.Scanln(&cariNama)

    posNama := SeqSearch_3(data, n, cariNama)
```

```
        if posNama == -1 {

            fmt.Println("Mahasiswa dengan nama tersebut tidak
ditemukan.")

        } else {

            fmt.Printf("Mahasiswa ditemukan pada indeks: %d\n",
posNama)

        }

// Pencarian berdasarkan nim

var cariNIM string

fmt.Print("Masukkan NIM mahasiswa yang ingin dicari: ")

fmt.Scanln(&cariNIM)

posNIM := BinarySearch_3(data, n, cariNIM)

if posNIM == -1 {

    fmt.Println("Mahasiswa dengan NIM tersebut tidak
ditemukan.")

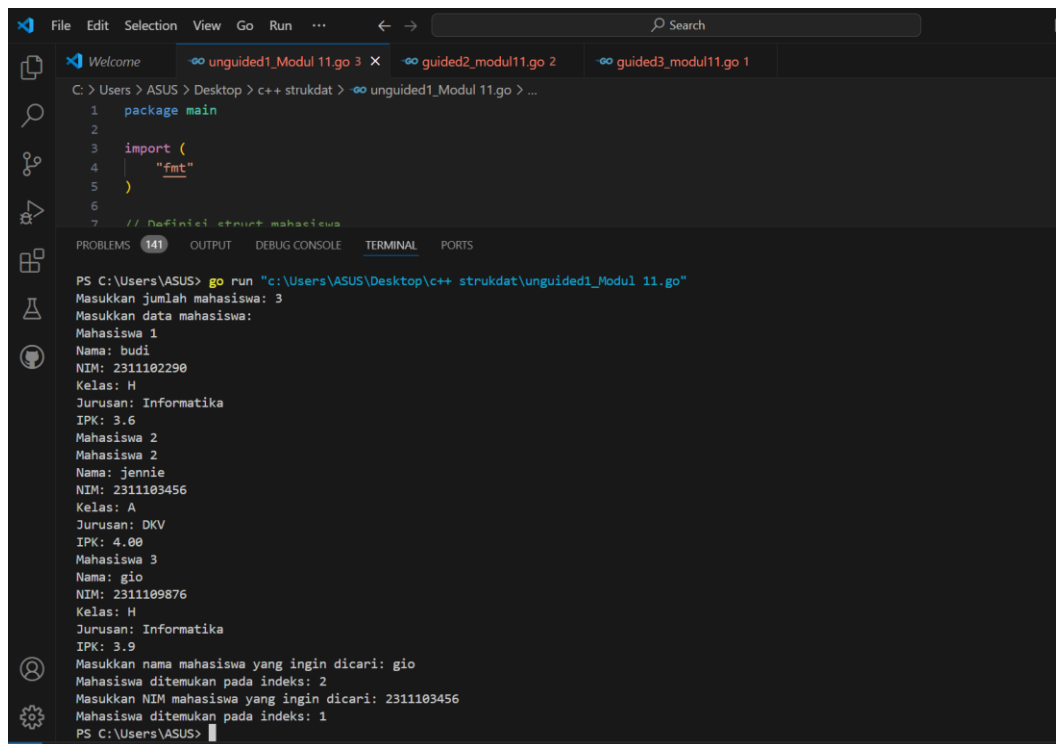
} else {

    fmt.Printf("Mahasiswa ditemukan pada indeks: %d\n",
posNIM)

}

}
```

## Screenshoot Program



The screenshot shows a Go program in VS Code with three tabs: `unguided1_Modul 11.go 3`, `guided2_modul11.go 2`, and `guided3_modul11.go 1`. The active file is `unguided1_Modul 11.go`, which contains the following code:

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 // Definisi struct mahasiswa
```

The terminal output shows the program execution and user input:

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\unguided1_Modul 11.go"
Masukkan jumlah mahasiswa: 3
Masukkan data mahasiswa:
Mahasiswa 1
Nama: budi
NIM: 2311102290
Kelas: H
Jurusan: Informatika
IPK: 3,6
Mahasiswa 2
Mahasiswa 2
Nama: jennie
NIM: 2311103456
Kelas: A
Jurusan: DKV
IPK: 4.00
Mahasiswa 3
Nama: gio
NIM: 2311109876
Kelas: H
Jurusan: Informatika
IPK: 3.9
Masukkan nama mahasiswa yang ingin dicari: gio
Mahasiswa ditemukan pada indeks: 2
Masukkan NIM mahasiswa yang ingin dicari: 2311103456
Mahasiswa ditemukan pada indeks: 1
PS C:\Users\ASUS>
```

## Deskripsi Program

Program ini mengelola data mahasiswa dengan fitur pencarian berdasarkan nama dan NIM. Data mahasiswa, seperti nama, NIM, kelas, jurusan, dan IPK, disimpan dalam array bertipe struct. Pencarian nama menggunakan *sequential search*, sedangkan pencarian NIM menggunakan *binary search* (dengan asumsi data NIM sudah terurut). Program akan menampilkan indeks mahasiswa yang ditemukan atau pesan jika data tidak ditemukan.



## 2. Source Code

```
package main

import "fmt"

// Define arrInt as an array of integers with a fixed size
type arrInt [4321]int

// Binary search function for descending sorted array
func BinarySearch_2(T arrInt, n int, X int) int {
    /* Mengembalikan indeks dari X apabila X ditemukan di dalam
    array T

    yang berisi n buah bilangan bulat terurut secara
    descending/menciut,

    atau -1 apabila X tidak ditemukan */

    var found int = -1
    var med int
    var kr int = 0
    var kn int = n - 1

    // Iterasi mencari elemen dengan binary search
    for kr <= kn && found == -1 {
        med = (kr + kn) / 2

        if X > T[med] { // Karena descending, jika X lebih
            besar, bergerak ke kiri
            kn = med - 1
        } else if X < T[med] { // Jika X lebih kecil, bergerak
            ke kanan
            kr = med + 1
        } else { // Jika ditemukan
            found = med
        }
    }
}
```

```

    }

    }

    return found
}

func main() {
    // Contoh penggunaan
    var data arrInt
    var n int
    // Input jumlah elemen
    fmt.Print("Masukkan jumlah elemen dalam array: ")
    fmt.Scanln(&n)
    // Input elemen array (harus terurut secara descending)
    fmt.Println("Masukkan elemen array (harus terurut
menurun):")

    for i := 0; i < n; i++ {
        fmt.Printf("Elemen %d: ", i+1)
        fmt.Scanln(&data[i])
    }

    // Input elemen yang dicari
    var search int
    fmt.Print("Masukkan elemen yang ingin dicari: ")
    fmt.Scanln(&search)

    // Panggil fungsi binary search
    result := BinarySearch_2(data, n, search)

    // Cetak hasil
    if result == -1 {

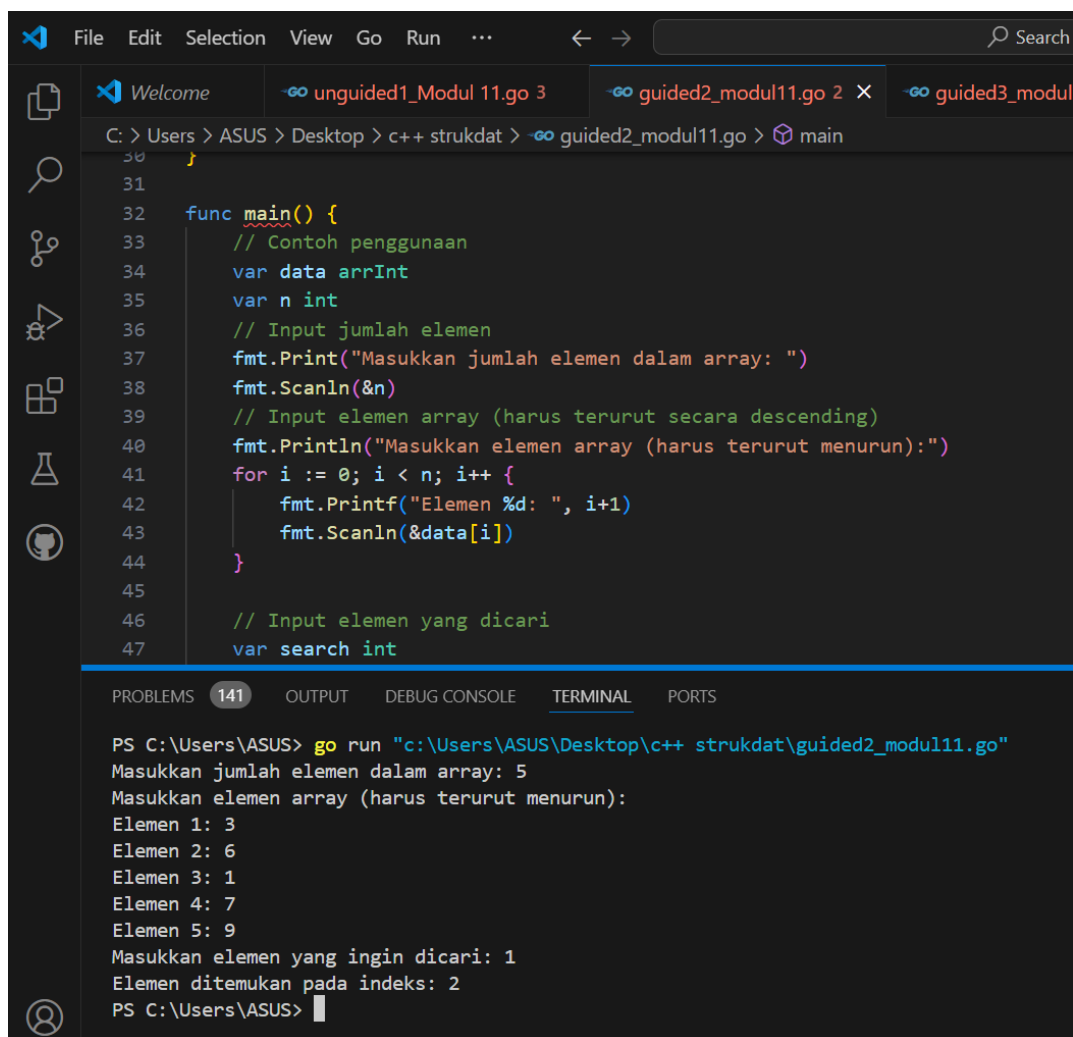
```

```

        fmt.Println("Elemen tidak ditemukan dalam array.")
    } else {
        fmt.Printf("Elemen ditemukan pada indeks: %d\n", result)
    }
}

```

## Screenshoot Program



The screenshot shows a Go program in VS Code. The source code is as follows:

```

31
32 func main() {
33     // Contoh penggunaan
34     var data arrInt
35     var n int
36     // Input jumlah elemen
37     fmt.Print("Masukkan jumlah elemen dalam array: ")
38     fmt.Scanln(&n)
39     // Input elemen array (harus terurut secara descending)
40     fmt.Println("Masukkan elemen array (harus terurut menurun):")
41     for i := 0; i < n; i++ {
42         fmt.Printf("Elemen %d: ", i+1)
43         fmt.Scanln(&data[i])
44     }
45
46     // Input elemen yang dicari
47     var search int

```

The terminal output shows the program's execution:

```

PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\guided2_modul11.go"
Masukkan jumlah elemen dalam array: 5
Masukkan elemen array (harus terurut menurun):
Elemen 1: 3
Elemen 2: 6
Elemen 3: 1
Elemen 4: 7
Elemen 5: 9
Masukkan elemen yang ingin dicari: 1
Elemen ditemukan pada indeks: 2
PS C:\Users\ASUS>

```

## Deskripsi Program

Program ini melakukan pencarian elemen dalam sebuah array bilangan bulat yang terurut secara menurun menggunakan metode *binary search*. Pengguna memasukkan jumlah elemen, elemen array (dengan urutan menurun), dan elemen yang ingin dicari. Program kemudian mencari elemen tersebut dan mengembalikan indeksnya jika ditemukan atau

pesan bahwa elemen tidak ditemukan. Pencarian dilakukan dengan efisien menggunakan pembagian array menjadi dua bagian pada setiap iterasi.

### 3. Source Code

```
package main

import "fmt"

// Define arrStr as an array of strings with a fixed size
type arrStr [1234]string

// Sequential search function
func SeqSearch_1(T arrStr, n int, X string) int {
    /* Mengembalikan indeks dari X apabila X ditemukan di dalam
    array T
        yang berisi n buah teks, atau -1 apabila X tidak
    ditemukan */

    var found int = -1

    var j int = 0

    // Iterasi mencari elemen yang cocok
    for j < n && found == -1 {
        if T[j] == X {
            found = j
        }
        j = j + 1
    }

    return found
}

func main() {
    // Contoh penggunaan
```

```
var data arrStr

var n int

// Input jumlah elemen

fmt.Print("Masukkan jumlah elemen dalam array: ")

fmt.Scanln(&n)

// Input elemen array

fmt.Println("Masukkan elemen array:")

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

    fmt.Printf("Elemen %d: ", i+1)

    fmt.Scanln(&data[i])

}

// Input elemen yang dicari

var search string

fmt.Print("Masukkan elemen yang ingin dicari: ")

fmt.Scanln(&search)

// Panggil fungsi sequential search

result := SeqSearch_1(data, n, search)

// Cetak hasil

if result == -1 {

    fmt.Println("Elemen tidak ditemukan dalam array.")

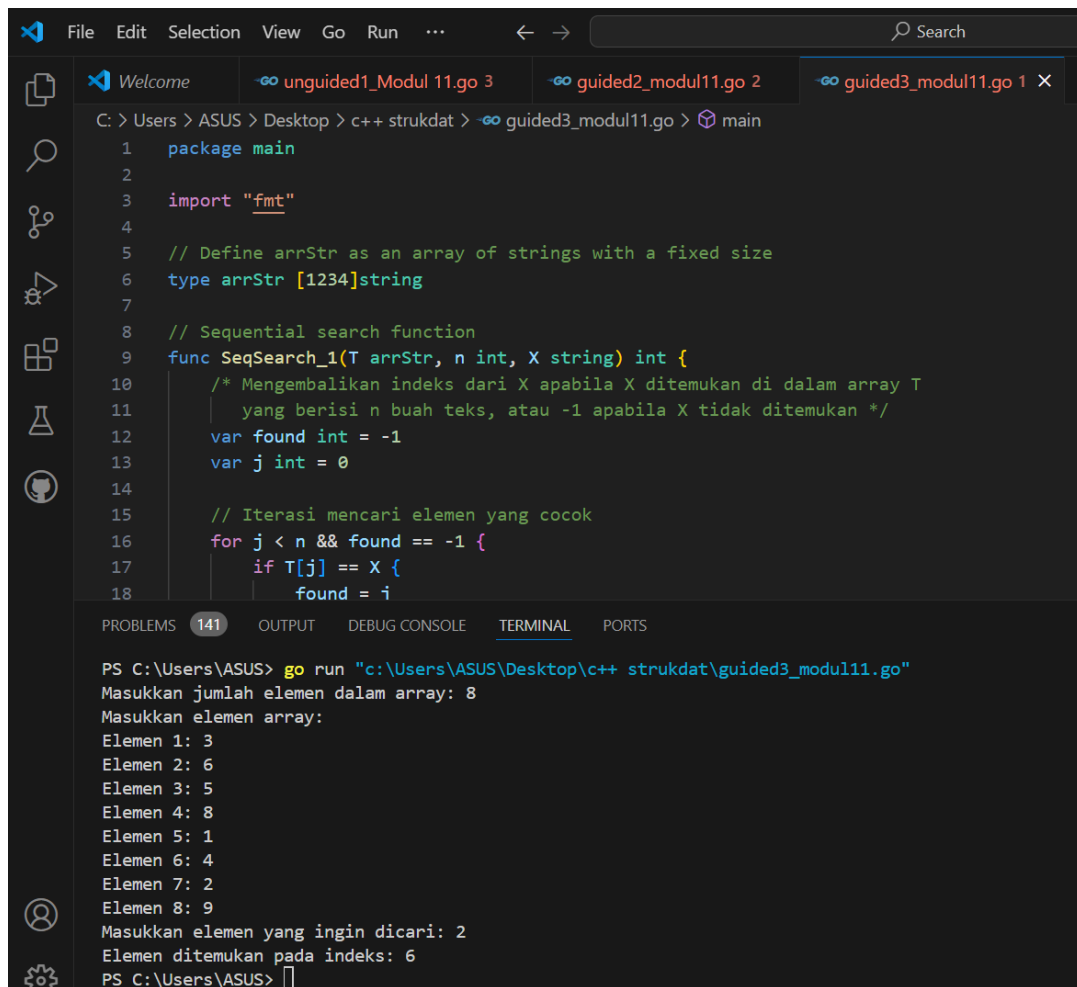
} else {

    fmt.Printf("Elemen ditemukan pada indeks: %d\n", result)

}

}
```

## Screenshoot Program



The screenshot shows a Go IDE with three tabs: `unguided1_Modul 11.go 3`, `guided2_modul11.go 2`, and `guided3_modul11.go 1`. The active tab is `guided3_modul11.go`, which contains the following Go code:

```
1 package main
2
3 import "fmt"
4
5 // Define arrStr as an array of strings with a fixed size
6 type arrStr [1234]string
7
8 // Sequential search function
9 func SeqSearch_1(T arrStr, n int, X string) int {
10     /* Mengembalikan indeks dari X apabila X ditemukan di dalam array T
11        yang berisi n buah teks, atau -1 apabila X tidak ditemukan */
12     var found int = -1
13     var j int = 0
14
15     // Iterasi mencari elemen yang cocok
16     for j < n && found == -1 {
17         if T[j] == X {
18             found = j
```

The terminal output shows the program's execution:

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\guided3_modul11.go"
Masukkan jumlah elemen dalam array: 8
Masukkan elemen array:
Elemen 1: 3
Elemen 2: 6
Elemen 3: 5
Elemen 4: 8
Elemen 5: 1
Elemen 6: 4
Elemen 7: 2
Elemen 8: 9
Masukkan elemen yang ingin dicari: 2
Elemen ditemukan pada indeks: 6
PS C:\Users\ASUS>
```

## Deskripsi Program

Program ini melakukan pencarian teks dalam sebuah array menggunakan metode *sequential search*. Pengguna memasukkan jumlah elemen, elemen-elemen array berupa teks, dan elemen yang ingin dicari. Program mencari elemen tersebut dengan memeriksa setiap elemen secara berurutan. Jika ditemukan, program mengembalikan indeks elemen; jika tidak, program mencetak pesan bahwa elemen tidak ditemukan. Metode ini cocok untuk array yang tidak terurut.

### 3. UNGUIDED

1. Pada pemilihan ketua RT yang baru saja berlangsung, terdapat 20 calon ketua yang bertanding memperebutkan suara warga. Perhitungan suara dapat segera dilakukan karena warga cukup mengisi formulir dengan nomor dari calon ketua RT yang dipilihnya. Seperti biasa, selalu ada pengisian yang tidak tepat atau dengan nomor pilihan di luar yang tersedia, sehingga data juga harus divalidasi. Tugas Anda untuk membuat program mencari siapa yang memenangkan pemilihan ketua RT. Buatlah program pilkart yang akan membaca, memvalidasi, dan menghitung suara yang diberikan dalam pemilihan ketua RT tersebut.

**Masukan** hanya satu baris data saja, berisi bilangan bulat valid yang kadang tersisipi dengan data tidak valid. Data valid adalah integer dengan nilai di antara 1 s.d. 20 (inklusif). Data berakhir jika ditemukan sebuah bilangan dengan nilai 0.

**Keluaran** dimulai dengan baris berisi jumlah data suara yang terbaca, diikuti baris yang berisi berapa banyak suara yang valid. Kemudian sejumlah baris yang mencetak data para calon apa saja yang mendapatkan suara.

No	Masukan	Keluaran
1	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

#### Source Code

```
package main

import (
    "fmt"
    "sort"
)

func main() {
    var dataSuara []int
    fmt.Print("Masukkan data suara: ")
```

```

for {
    var suara int
    _, err := fmt.Scan(&suara)
    if err != nil {
        break
    }
    if suara == 0 {
        break
    }
    dataSuara = append(dataSuara, suara)
}

totalSuara := len(dataSuara)
suaraValid := 0
hitungSuara := make(map[int]int)
for _, suara := range dataSuara {
    if 1 <= suara && suara <= 20 {
        suaraValid++
        hitungSuara[suara]++
    }
}

type hasilSuara struct {
    calon int
    jumlah int
}

var hasil []hasilSuara
for calon, jumlah := range hitungSuara {

```



```

        hasil = append(hasil, hasilSuara{calon, jumlah})
    }

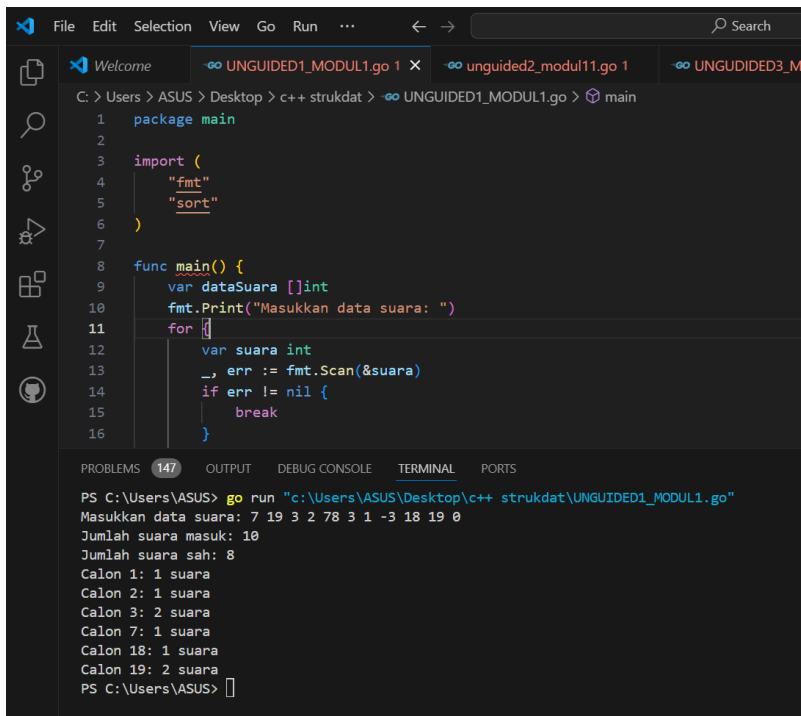
    sort.Slice(hasil, func(i, j int) bool {
        return hasil[i].calon < hasil[j].calon
    })

    fmt.Println("Jumlah suara masuk:", totalSuara)
    fmt.Println("Jumlah suara sah:", suaraValid)

    for _, h := range hasil {
        fmt.Printf("Calon %d: %d suara\n", h.calon,
h.jumlah)
    }
}

```

## Screenshoot Program



The screenshot shows a Go program in VS Code. The source code in the editor is as follows:

```

1  package main
2
3  import (
4      "fmt"
5      "sort"
6  )
7
8  func main() {
9      var dataSuara []int
10     fmt.Print("Masukkan data suara: ")
11     for {
12         var suara int
13         _, err := fmt.Scan(&suara)
14         if err != nil {
15             break
16         }

```

The terminal output at the bottom shows the program's execution:

```

PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\UNGUIDED1_MODUL1.go"
Masukkan data suara: 7 19 3 2 78 3 1 -3 18 19 0
Jumlah suara masuk: 10
Jumlah suara sah: 8
Calon 1: 1 suara
Calon 2: 1 suara
Calon 3: 2 suara
Calon 7: 1 suara
Calon 18: 1 suara
Calon 19: 2 suara
PS C:\Users\ASUS>

```

## Deskripsi Program

Program ini menghitung hasil pemungutan suara dari data suara yang dimasukkan pengguna. Data suara berupa angka terus dimasukkan hingga angka 0 atau input tidak valid diberikan. Suara dianggap valid jika berada dalam rentang 1 hingga 20. Program menghitung jumlah total suara masuk, jumlah suara valid, serta distribusi suara untuk setiap calon, lalu menampilkan hasilnya dalam urutan calon yang meningkat. Metode ini menggunakan peta (*map*) untuk menghitung suara per calon dan fungsi *sort.Slice* untuk mengurutkan hasil berdasarkan nomor calon.

2. Berdasarkan program sebelumnya, buat program pilkart yang mencari siapa pemenang pemilihan ketua RT. Sekaligus juga ditentukan bahwa wakil ketua RT adalah calon yang mendapatkan suara terbanyak kedua. Jika beberapa calon mendapatkan suara terbanyak yang sama, ketua terpilih adalah dengan nomor peserta yang paling kecil dan wakilnya dengan nomor peserta terkecil berikutnya.

**Masukan** hanya satu baris data saja, berisi bilangan bulat valid yang kadang tersisipi dengan data tidak valid. Data valid adalah bilangan bulat dengan nilai di antara 1 s.d. 20 (inklusif). Data berakhir jika ditemukan sebuah bilangan dengan nilai 0

**Keluaran** dimulai dengan baris berisi jumlah data suara yang terbaca, diikuti baris yang berisi berapa banyak suara yang valid. Kemudian tercetak calon nomor berapa saja yang menjadi pasangan ketua RT dan wakil ketua RT yang baru.

No	Masukan	Keluaran
1	7 19 3 2 78 3 1 -3 18 19 0	Suara masuk: 10 Suara sah: 8 Ketua RT: 3 Wakil ketua: 19

## Source Code

```
package main

import (
    "fmt"
    "sort"
)

type hasilSuara struct {
    calon    int
    jumlah  int
}
```

```
}

func main() {
    var dataSuara []int

    fmt.Print("Masukkan data suara : ")

    for {
        var suara int

        _, err := fmt.Scan(&suara)

        if err != nil || suara == 0 {
            break
        }

        dataSuara = append(dataSuara, suara)
    }

    totalSuara := len(dataSuara)

    suaraValid := 0

    hitungSuara := make(map[int]int)

    for _, suara := range dataSuara {
        if 1 <= suara && suara <= 20 {
            suaraValid++

            hitungSuara[suara]++
        }
    }

    var hasil []hasilSuara

    for calon, jumlah := range hitungSuara {
        hasil = append(hasil, hasilSuara{calon, jumlah})
    }
}
```

```

sort.Slice(hasil, func(i, j int) bool {

    if hasil[i].jumlah == hasil[j].jumlah {

        return hasil[i].calon < hasil[j].calon

    }

    return hasil[i].jumlah > hasil[j].jumlah

}))

fmt.Println("Jumlah suara masuk:", totalSuara)

fmt.Println("Jumlah suara sah:", suaraValid)

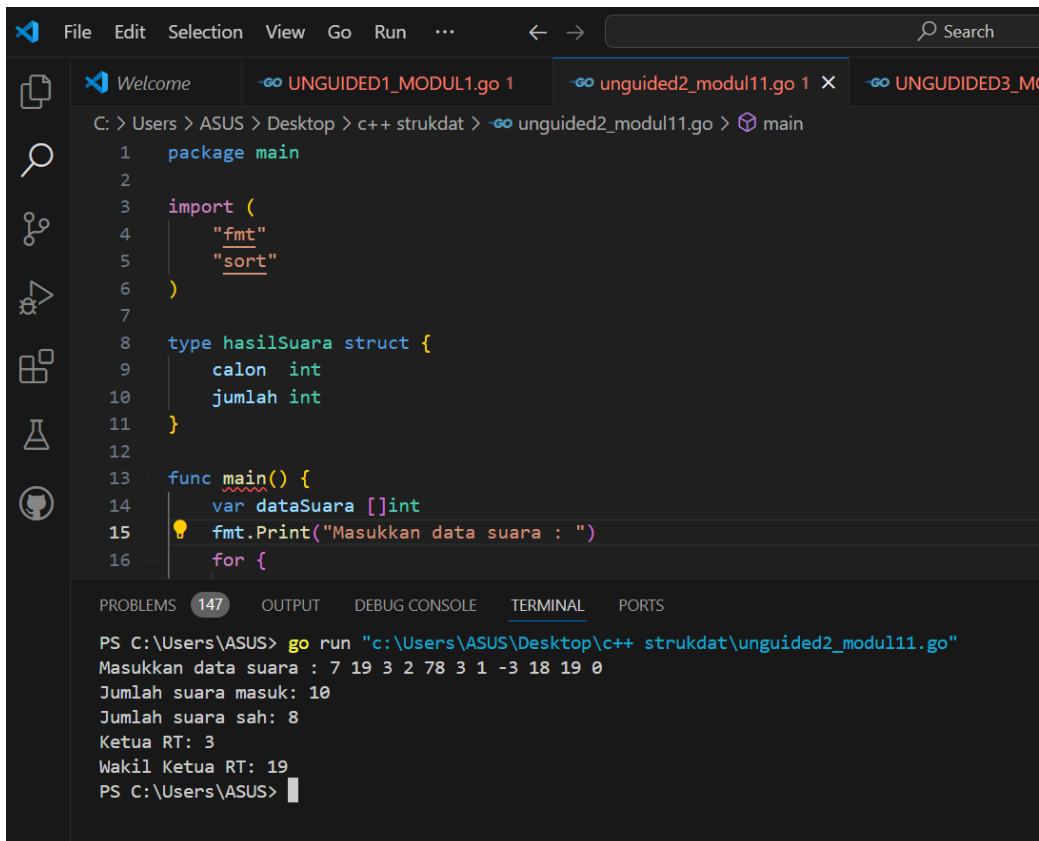
fmt.Printf("Ketua RT: %d\n", hasil[0].calon)

fmt.Printf("Wakil Ketua RT: %d\n", hasil[1].calon)

}

```

## Screenshoot Program



The screenshot shows a Go program in VS Code. The code defines a struct `hasilSuara` with fields `calon` and `jumlah`. The `main` function prompts the user to input data, which is then processed by the `sort.Slice` function. The terminal output shows the program running successfully with the following results:

```

PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\unguided2_modul11.go"
Masukkan data suara : 7 19 3 2 78 3 1 -3 18 19 0
Jumlah suara masuk: 10
Jumlah suara sah: 8
Ketua RT: 3
Wakil Ketua RT: 19
PS C:\Users\ASUS>

```

## Deskripsi Program

Program ini menghitung hasil pemungutan suara untuk memilih Ketua dan Wakil Ketua RT dari data suara yang dimasukkan pengguna. Data suara berupa angka terus dimasukkan hingga angka 0 atau input tidak valid diberikan. Suara valid berada dalam rentang 1 hingga 20. Program menghitung jumlah total suara, jumlah suara valid, dan mendistribusikan suara kepada setiap calon. Hasilnya diurutkan berdasarkan jumlah suara terbanyak, dengan prioritas calon bernomor lebih kecil jika jumlah suara sama. Program menampilkan Ketua RT (peraih suara terbanyak) dan Wakil Ketua RT (peringkat kedua).

3. Diberikan  $n$  data integer positif dalam keadaan terurut membesar dan sebuah integer lain  $k$ , apakah bilangan  $k$  tersebut ada dalam daftar bilangan yang diberikan? Jika ya, berikan indeksnya, jika tidak sebutkan "TIDAK ADA".

**Masukan** terdiri dari dua baris. Baris pertama berisi dua buah integer positif, yaitu  $n$  dan  $k$ .  $n$  menyatakan banyaknya data, dimana  $1 < n \leq 1000000$ .  $k$  adalah bilangan yang ingin dicari. Baris kedua berisi  $n$  buah data integer positif yang sudah terurut membesar.

**Keluaran** terdiri dari satu baris saja, yaitu sebuah bilangan yang menyatakan posisi data yang dicari ( $k$ ) dalam kumpulan data yang diberikan. Posisi data dihitung dimulai dari angka 0. Atau memberikan keluaran "TIDAK ADA" jika data  $k$  tersebut tidak ditemukan dalam kumpulan.

Program yang dibangun harus menggunakan subprogram dengan mengikuti kerangka yang sudah diberikan berikut ini.

```
package main
import "fmt"

const NMAX = 1000000
var data [NMAX]int

func main(){
/* buatlah kode utama yang membaca baris pertama (n dan k). kemudian data
diisi
oleh prosedur isiArray(n), dan pencarian oleh fungsi posisi(n,k), dan
setelah
itu output dicetak. */
}
```

```

func isiArray(n int){
/* I.S. terdefinisi integer n, dan sejumlah n data sudah siap pada piranti
masukan.
   F.S. Array data berisi n (<=NMAX) bilangan */
}

func posisi(n, k int) int {
/* mengembalikan posisi k dalam array data dengan n elemen. Posisi dimulai
dari
   posisi 0. Jika tidak ada kembalikan -1 */
}

```

**Contoh masukan dan keluaran:**

No	Masukan	Keluaran	Penjelasan
1	12 534 1 3 8 16 32 123 323 323 534 543 823 999	8	Data 534 berada pada posisi ke-8 dihitng dari awal data.
2	12 535 1 3 8 16 32 123 323 323 534 543 823 999	TIDAK ADA	

## 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)
    result := posisi(n, k)
    if result == -1 {
        fmt.Println("TIDAK ADA")
    } else {

```

```

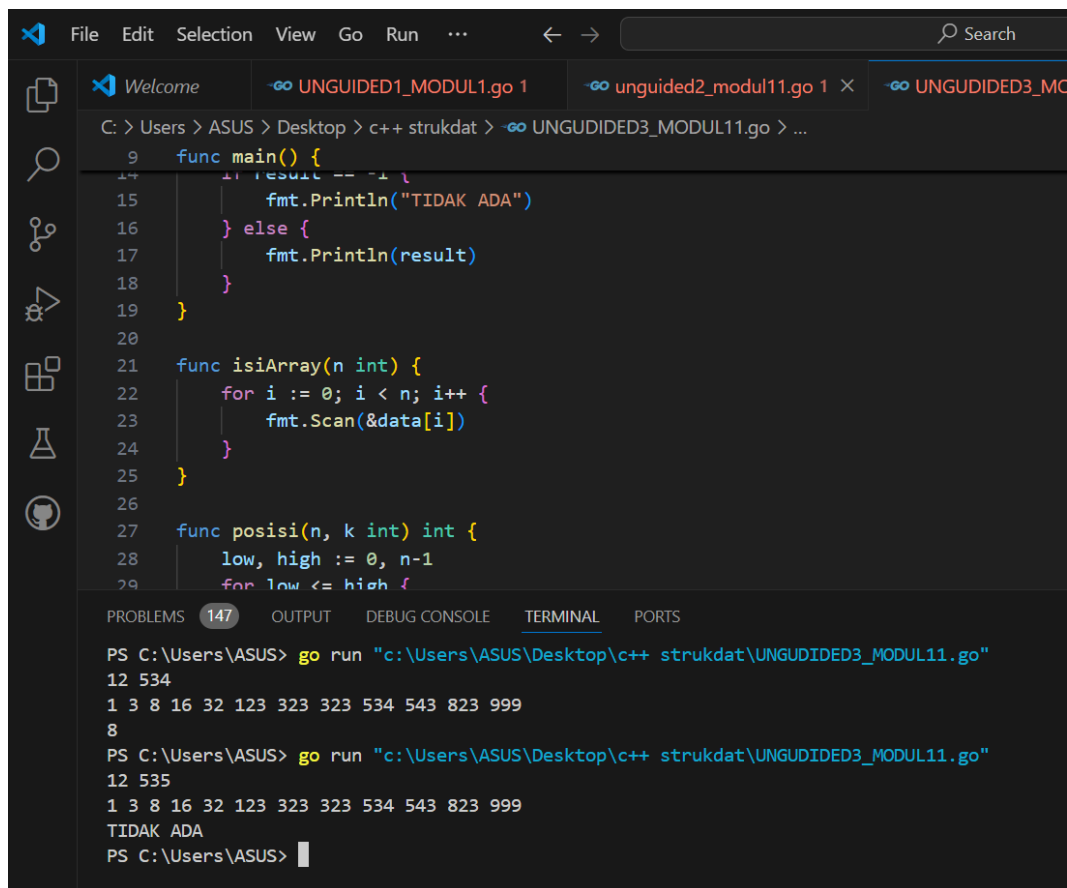
        fmt.Println(result)
    }
}

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

func posisi(n, k int) int {
    low, high := 0, 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
}

```

## Screenshoot Program



```
9 func main() {
10     result := -1
15     fmt.Println("TIDAK ADA")
16 } else {
17     fmt.Println(result)
18 }
19 }
20
21 func isiArray(n int) {
22     for i := 0; i < n; i++ {
23         fmt.Scan(&data[i])
24     }
25 }
26
27 func posisi(n, k int) int {
28     low, high := 0, n-1
29     for low <= high {
```

PROBLEMS 147 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\UNGUDIDED3_MODUL11.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
PS C:\Users\ASUS> go run "c:\Users\ASUS\Desktop\c++ strukdat\UNGUDIDED3_MODUL11.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA
PS C:\Users\ASUS>
```

## Deskripsi Program

Program ini mencari posisi elemen dalam array menggunakan *binary search*. Pengguna memasukkan jumlah elemen array (n), nilai yang dicari (k), dan elemen-elemen array dalam urutan menaik. Fungsi posisi melakukan pencarian dengan membagi array menjadi dua bagian hingga elemen ditemukan atau tidak ada lagi yang bisa diperiksa. Jika elemen ditemukan, program mencetak indeksnya; jika tidak, program mencetak "TIDAK ADA". Metode ini efisien untuk array terurut dengan kompleksitas waktu  $O(\log n)$ .