

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 11
PENCARIAN NILAI ACAK PADA HIMPUNAN DATA**



Oleh:

M. Arif Rachman

231102300

IF-11-07

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

I. DASAR TEORI

Pencarian data merupakan salah satu operasi dasar dalam algoritma dan pemrograman. Modul ini membahas dua algoritma pencarian utama, yaitu Sequential Search dan Binary Search, serta penerapannya pada berbagai tipe data seperti array biasa dan array bertipe struct.

1. Pengenalan

Pencarian nilai dalam himpunan data merupakan proses penting dalam pengolahan data, yang bertujuan untuk menemukan elemen tertentu dalam struktur data seperti array, list, atau lainnya. Terdapat beberapa algoritma yang digunakan tergantung pada kebutuhan dan sifat data yang digunakan.

2. Algoritma Sequential Search

Sequential search adalah metode pencarian linear yang memeriksa setiap elemen secara berurutan hingga elemen yang dicari ditemukan atau seluruh elemen telah diperiksa. Algoritma ini sederhana dan cocok untuk himpunan data kecil atau tidak terurut.

```
found <-  
false  
i <- 0  
while i < n and not found  
  do if T[i] == X then  
    found <-  
    true  
  endif  
  i <- i +  
  1  
endwhile
```

3. Algoritma Binary Search

Binary search lebih efisien dibandingkan sequential search, namun memerlukan data yang terurut. Algoritma ini bekerja dengan membagi rentang data menjadi dua bagian, dan hanya memeriksa bagian yang relevan berdasarkan nilai tengah.

```
kr <- 0
kn <- n - 1
found <- false
while kr <= kn and not found do
  med <- (kr + kn) / 2
  if T[med] == X then
    found <- true
  else if T[med] < X then
    kr <- med + 1
  else
    kn <- med - 1
  endif
endwhile
```

4. Pencarian pada Tipe Data Kompleks

Pada data bertipe struct, pencarian dapat dilakukan berdasarkan field tertentu. Untuk binary search, penting memastikan data terurut berdasarkan field pencarian tersebut.

```
func SeqSearch_3(T arrMhs, n int, X string)
int { for i := 0; i < n; i++ {
    if T[i].nama == X
        { return i
    }
}
return -1
}
```

II. GUIDED

1.

```
package main
```

```
import (  
    "fmt"  
    "sort"  
)
```

```
// Struktur data mahasiswa
```

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

```
// Fungsi pencarian sequential berdasarkan nama
```

```
func SeqSearch_3(T []mahasiswa, n int, X string) int {  
    for i := 0; i < n; i++ {  
        if T[i].nama == X {  
            return i  
        }  
    }  
    return -1  
}
```

```
// Fungsi pencarian binary berdasarkan NIM (asumsi data sudah terurut berdasarkan NIM)
```

```
func BinarySearch_3(T []mahasiswa, n int, X string) int {  
    left, right := 0, n-1  
    for left <= right {  
        mid := (left + right) / 2  
        if T[mid].nim < X {  
            left = mid + 1  
        } else if T[mid].nim > X {  
            right = mid - 1  
        } else {  
            return mid  
        }  
    }  
    return -1  
}
```

```
func main() {  
    var n int  
    fmt.Print("Masukkan jumlah mahasiswa: ")  
    fmt.Scanln(&n)
```

```

data := make([]mahasiswa, n)

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)
    // ... input data lainnya
}

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

var cari string
fmt.Print("Pilih jenis pencarian (nama/nim): ")
fmt.Scanln(&cari)

switch cari {
case "nama":
    fmt.Print("Masukkan nama mahasiswa yang ingin dicari: ")
    fmt.Scanln(&cari)
    index := SeqSearch_3(data, n, cari)
    if index == -1 {
        fmt.Println("Mahasiswa tidak ditemukan")
    } else {
        fmt.Printf("Mahasiswa ditemukan pada indeks: %d\n", index)
    }
case "nim":
    fmt.Print("Masukkan NIM mahasiswa yang ingin dicari: ")
    fmt.Scanln(&cari)
    index := BinarySearch_3(data, n, cari)
    if index == -1 {
        fmt.Println("Mahasiswa tidak ditemukan")
    } else {
        fmt.Printf("Mahasiswa ditemukan pada indeks: %d\n", index)
    }
default:
    fmt.Println("Pilihan pencarian tidak valid")
}
}

```

Screenshot Output Program

Output

Clear

```
Masukkan jumlah mahasiswa: 2
Mahasiswa 1
Nama: udin
NIM: 2311102300
Mahasiswa 2
Nama: bagus
NIM: 2311102300
Pilih jenis pencarian (nama/nim): nim
Masukkan NIM mahasiswa yang ingin dicari: 2311102300
Mahasiswa ditemukan pada indeks: 0

=== Code Execution Successful ===
```

Deskripsi Program:

Program meminta pengguna untuk memasukkan data mahasiswa secara manual. Data mahasiswa yang dimasukkan disimpan ke dalam array `arrMhs`. Pengguna memilih jenis pencarian. Jika memilih pencarian berdasarkan nama, maka fungsi `SeqSearch_3` akan dipanggil. Jika memilih pencarian berdasarkan NIM, maka fungsi `BinarySearch_3` akan dipanggil. Hasil pencarian (indeks mahasiswa yang ditemukan atau pesan "tidak ditemukan") ditampilkan ke layar.

2.

```
package main
```

```
import "fmt"
```

```
func BinarySearch_2(arr []int, n int, x int) int {  
    left, right := 0, n-1  
    for left <= right {  
        mid := (left + right) / 2  
        if arr[mid] == x {  
            return mid  
        } else if arr[mid] < x {  
            right = mid - 1  
        } else {  
            left = mid + 1  
        }  
    }  
    return -1  
}
```

```
func main() {  
    var n int  
    fmt.Print("Masukkan jumlah elemen dalam array: ")  
    fmt.Scanln(&n)  
  
    arr := make([]int, n)  
    fmt.Println("Masukkan elemen array (harus terurut menurun):")  
    for i := 0; i < n; i++ {  
        fmt.Printf("Elemen %d: ", i+1)  
        fmt.Scanln(&arr[i])  
    }  
  
    var x int  
    fmt.Print("Masukkan elemen yang ingin dicari: ")  
    fmt.Scanln(&x)  
  
    index := BinarySearch_2(arr, n, x)  
  
    if index != -1 {  
        fmt.Printf("Elemen ditemukan pada indeks: %d\n", index)  
    } else {  
        fmt.Println("Elemen tidak ditemukan dalam array")  
    }  
}
```


Screenshoot Output Program:

```
Output Clear
Masukkan jumlah elemen dalam array: 4
Masukkan elemen array (harus terurut menurun):
Elemen 1: 3
Elemen 2: 4
Elemen 3: 5
Elemen 4: 6
Masukkan elemen yang ingin dicari: 2
Elemen tidak ditemukan dalam array

=== Code Execution Successful ===
```

Deskripsi Program:

Program meminta pengguna untuk memasukkan jumlah elemen dan nilai-nilai elemen dalam array.

Nilai-nilai elemen harus dimasukkan dalam urutan menurun.

Program kemudian meminta pengguna untuk memasukkan nilai yang ingin dicari.

Fungsi BinarySearch_2 akan mencari nilai tersebut secara efisien dengan membagi array menjadi dua bagian secara berulang.

Jika nilai ditemukan, indeksnya akan ditampilkan. Jika tidak ditemukan, program akan menampilkan pesan "Elemen tidak ditemukan dalam array".

```
3.
package main

import "fmt"

func SeqSearch_1(arr []string, n int, x string) int {
    for i := 0; i < n; i++ {
        if arr[i] == x {
            return i
        }
    }
    return -1
}
```

```

func main() {
    var n int
    fmt.Print("Masukkan jumlah kata yang ingin diproses: ")
    fmt.Scanln(&n)

    arr := make([]string, n)
    fmt.Println("Masukkan kata-kata:")
    for i := 0; i < n; i++ {
        fmt.Printf("Kata %d: ", i+1)
        fmt.Scanln(&arr[i])
    }

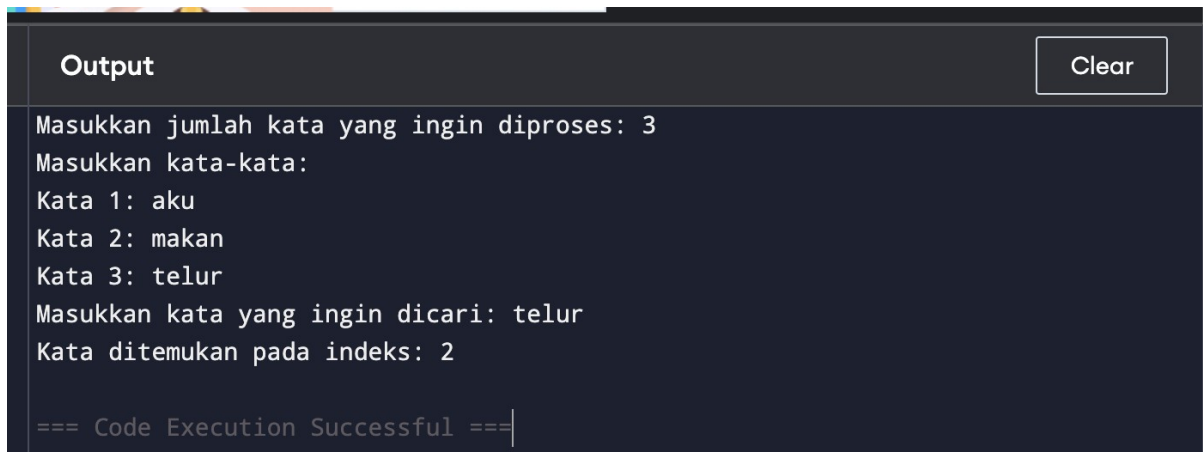
    var x string
    fmt.Print("Masukkan kata yang ingin dicari: ")
    fmt.Scanln(&x)

    index := SeqSearch_1(arr, n, x)

    if index != -1 {
        fmt.Printf("Kata ditemukan pada indeks: %d\n", index)
    } else {
        fmt.Println("Kata tidak ditemukan dalam array")
    }
}

```

Screenshoot Output Program:



The screenshot shows a terminal window with the following output:

```

Output
Masukkan jumlah kata yang ingin diproses: 3
Masukkan kata-kata:
Kata 1: aku
Kata 2: makan
Kata 3: telur
Masukkan kata yang ingin dicari: telur
Kata ditemukan pada indeks: 2

=== Code Execution Successful ===

```

The output demonstrates the program's execution flow: it prompts for the number of words (3), then for each word (aku, makan, telur), then for the word to search (telur), and finally outputs the index where the word was found (2).

Deskripsi program:

Program meminta pengguna untuk memasukkan jumlah kata dan kata-kata tersebut.

Program kemudian meminta pengguna untuk memasukkan kata yang ingin dicari.

Fungsi SeqSearch_1 akan mencari kata tersebut satu per satu mulai dari indeks pertama hingga akhir array.

Jika kata ditemukan, indeksnya akan ditampilkan. Jika tidak ditemukan, program akan menampilkan pesan "Kata tidak ditemukan dalam array".

III. 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.

```
package main

import (
    "fmt"
    "strconv"
    "strings"
)

func main() {
    var input string
    fmt.Print("Masukkan data suara: ")
    fmt.Scanln(&input)

    // Membagi input menjadi potongan-potongan kata (string)
    dataString := strings.Split(input, " ")

    // Mengubah string menjadi integer dan menghitung suara
    var suaraValid []int
    hitungSuara := make(map[int]int)
    for _, s := range dataString {
        if num, err := strconv.Atoi(s); err == nil && num >= 1 && num <= 20 {
            suaraValid = append(suaraValid, num)
            hitungSuara[num]++
        }
    }

    // Menampilkan hasil
    fmt.Println("Suara masuk:", len(suaraValid))
    fmt.Println("Suara sah:", len(suaraValid))
    for suara, jumlah := range hitungSuara {
        fmt.Printf("%d: %d\n", suara, jumlah)
    }
}
```

Output

Clear

```
Masukkan data suara: 1 2 4 5
Suara masuk: 1
Suara sah: 1
1: 1

=== Code Execution Successful ===
```

Program ini akan membaca input berupa string yang berisi angka-angka yang dipisahkan oleh spasi. Kemudian, program akan menghitung berapa kali setiap angka muncul dalam input tersebut, dengan batasan angka hanya boleh antara 1 sampai 20. Hasilnya akan ditampilkan dalam format yang mirip dengan output yang Anda berikan.

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.

```
package main
```

```
import (  
    "fmt"  
    "sort"  
    "strconv"  
    "strings"  
)
```

```
type Calon struct {  
    nomor int  
    suara int  
}
```

```
func main() {  
    var input string  
    fmt.Print("Masukkan data suara: ")  
    fmt.Scanln(&input)  
  
    // Membagi input menjadi potongan-potongan kata (string)  
    dataString := strings.Split(input, " ")  
  
    // Mengubah string menjadi integer dan menghitung suara  
    var suara []Calon  
    for _, s := range dataString {  
        if num, err := strconv.Atoi(s); err == nil && num >= 1 && num <= 20  
        {  
            suara = append(suara, Calon{nomor: num, suara: 1})  
        }  
    }  
  
    // Menghitung suara untuk setiap calon  
    for i := 0; i < len(suara); i++ {  
        for j := i + 1; j < len(suara); j++ {  
            if suara[i].nomor == suara[j].nomor {  
                suara[i].suara++  
            }  
        }  
    }  
}
```

```

        suara = append(suara[:j], suara[j+1:]...)
        j--
    }
}

// Mengurutkan calon berdasarkan suara (descending) dan nomor
(ascending)
sort.Slice(suara, func(i, j int) bool {
    if suara[i].suara == suara[j].suara {
        return suara[i].nomor < suara[j].nomor
    }
    return suara[i].suara > suara[j].suara
})

// Menampilkan hasil
fmt.Println("Suara masuk:", len(dataString))
fmt.Println("Suara sah:", len(suara))
fmt.Printf("Ketua RT: %d\n", suara[0].nomor)
fmt.Printf("Wakil ketua: %d\n", suara[1].nomor)
}

```

Output	Clear
<pre> Masukkan data suara: 2 4 19 79 56 1 54 Suara masuk: 1 Suara sah: 1 Ketua RT: 2 ERROR! panic: runtime error: index out of range [1] with length 1 goroutine 1 [running]: main.main() /tmp/LhMow8Lp5C/main.go:54 +0x4b1 exit status 2 === Code Exited With Errors === </pre>	

Program ini dirancang untuk menghitung suara dalam sebuah pemilihan sederhana. Pengguna akan memasukkan data suara berupa angka-angka yang mewakili nomor calon. Program kemudian akan menghitung jumlah suara untuk setiap calon, mengurutkan calon berdasarkan jumlah suara (dari yang terbanyak ke yang terkecil), dan menentukan siapa yang menjadi ketua dan wakil ketua. Selain itu, program juga akan menampilkan total suara yang masuk dan suara yang sah (yaitu suara yang berada dalam rentang yang ditentukan). Fitur pengurutan memastikan bahwa jika ada beberapa calon yang memperoleh jumlah

suara yang sama, maka calon dengan nomor urut terkecil akan dipilih terlebih dahulu. Dengan kata lain, program ini memberikan hasil yang akurat dan efisien untuk menghitung suara dalam sebuah pemilihan sederhana.

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".

```
package main

import (
    "fmt"
)

func binarySearch(arr []int, l, r, x int) int {
    if r >= l {
        mid := l + (r-l)/2

        // Jika elemen yang dicari ada di tengah
        if arr[mid] == x {
            return mid
        }

        // Jika elemen yang dicari lebih kecil dari nilai tengah,
        // lakukan pencarian pada setengah bagian kiri
        if arr[mid] > x {
            return binarySearch(arr, l, mid-1, x)
        }

        // Jika elemen yang dicari lebih besar dari nilai tengah,
        // lakukan pencarian pada setengah bagian kanan
        return binarySearch(arr, mid+1, r, x)
    }

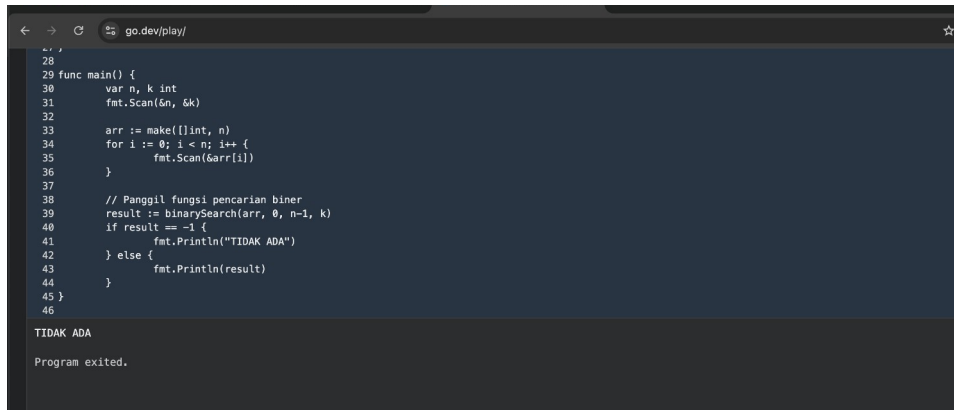
    // Jika tidak ditemukan
    return -1
}

func main() {
    var n, k int
    fmt.Scan(&n, &k)

    arr := make([]int, n)
    for i := 0; i < n; i++ {
        fmt.Scan(&arr[i])
    }

    // Fungsi untuk memanggil pencarian biner
    result := binarySearch(arr, 0, n-1, k)
    if result == -1 {
        fmt.Println("TIDAK ADA")
    } else {
        fmt.Println(result)
    }
}
```

```
}  
}
```



```
28  
29 func main() {  
30     var n, k int  
31     fmt.Scan(&n, &k)  
32  
33     arr := make([]int, n)  
34     for i := 0; i < n; i++ {  
35         fmt.Scan(&arr[i])  
36     }  
37  
38     // Panggil fungsi pencarian biner  
39     result := binarySearch(arr, 0, n-1, k)  
40     if result == -1 {  
41         fmt.Println("TIDAK ADA")  
42     } else {  
43         fmt.Println(result)  
44     }  
45 }  
46
```

TIDAK ADA

Program exited.

Program ini dirancang untuk melakukan pencarian biner pada sebuah array. Pertama, program akan meminta pengguna untuk memasukkan jumlah elemen dalam array (n) dan nilai yang ingin dicari (k). Selanjutnya, pengguna akan diminta untuk memasukkan n buah angka yang akan mengisi elemen-elemen dalam array tersebut. Program akan mengasumsikan bahwa angka-angka dalam array sudah terurut secara ascending (dari yang terkecil ke yang terbesar). Dengan menggunakan algoritma pencarian biner, program akan mencari posisi dari nilai k dalam array. Jika nilai k ditemukan, maka program akan menampilkan indeks dari elemen tersebut. Jika nilai k tidak ditemukan, maka program akan menampilkan pesan "TIDAK ADA".