

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 8
PENCARIAN NILAI ACAK PADA HIMPUNAN DATA**



Oleh:

BERTHA ADELA

103112400041

IF-12-01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025**

I. DASAR TEORI

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 $N-1$, 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.

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.

II. GUIDED

• GUIDED 1

Code:

```
1  package main
2  import( "fmt"
3         "sort"
4  )
5
6  func sequentialSearch(arr []float64, target float64) (int, int) {
7      iterations := 0
8      for i, val := range arr {
9          iterations++
10         fmt.Printf("Sequential Step %d: cek arr[%d] = %.1f\n", iterations, i, val)
11         if val == target {
12             return i, iterations
13         }
14     }
15     return -1, iterations
16 }
```

```
18 func binarySearch(arr []float64, target float64) (int, int) {
19     iterations := 0
20     low := 0
21     high := len(arr)-1
22
23     for low <= high {
24         iterations++
25         mid := (low + high)/2
26         fmt.Printf("Binary Steps %d: cek arr[%d] = %.1f\n", iterations, mid, arr[mid])
27         if arr[mid] == target {
28             return mid, iterations
29         } else if target < arr[mid] {
30             high = mid - 1
31         } else {
32             low = mid + 1
33         }
34     }
35     return -1, iterations
36 }
```

```
38 func main() {
39     //Array awal
40     data := []float64{2,7,9,1,5,6,18,13,25,20}
41     target := 13.0
42
43     fmt.Println("Sequentiel Search (data tidak perlu urut):")
44     idxSeq, iterSeq := sequentialSearch(data, target)
45     if idxSeq != -1 {
46         fmt.Printf("Hasil: Ditemukan di indeks %d dalam %d langkah\n\n", idxSeq, iterSeq)
47     } else {
48         fmt.Printf("Hasil: Tidak ditemukan setelah %d langkah\n\n", iterSeq)
49     }
50
51     sort.Float64s(data)
52     fmt.Println("Binary Search (setelah data diurutkan):")
53     fmt.Println("Data terurut:", data)
54
55     idxBin, iterBin := binarySearch(data, target)
56     if idxBin != -1 {
57         fmt.Printf("Hasil: Ditemukan di indeks %d dalam %d langkah\n", idxBin, iterBin)
58     } else {
59         fmt.Printf("Hasil: Tidak ditemukan setelah %d langkah\n", iterBin)
60     }
61 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\User
uited1.go"
Sequentiel 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 langkah

Binary Search (setelah data diurutkan):
Data terurut: [1 2 5 6 7 9 13 18 20 25]
Binary Steps 1: cek arr[4] = 7.0
Binary Steps 2: cek arr[7] = 18.0
Binary Steps 3: cek arr[5] = 9.0
Binary Steps 4: cek arr[6] = 13.0
Hasil: Ditemukan di indeks 6 dalam 4 langkah
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program diatas merupakan contoh penerapan Sequential Search dan Binary Search.

• GUIDED 2

Code:

```
SMT2 > Pertemuan9 > 103112400041_Guided2.go > SeqSearch_3

1  package main
2
3  import (
4      "fmt"
5      "sort"
6  )
7
8  type mahasiswa struct {
9      nama, nim, kelas, jurusan string
10     ipk                        float64
11 }
12
13 type arrMhs [2023]mahasiswa
14
15 // Sequential Search berdasarkan nama
16 func SeqSearch_3(T arrMhs, n int, X string) (int, int) {
17     var found int = -1
18     var j int = 0
19     var iterasi int = 0
20
21     for j < n && found == -1 {
22         iterasi++
23         if T[j].nama == X {
24             found = j
25         }
26         j++
27     }
28     return found, iterasi
29 }
30
31 // Binary Search berdasarkan NIM (data harus sudah terurut b
32 func BinarySearch_3(T arrMhs, n int, X string) (int, int) {
33     var found int = -1
34     var med int
35     var kr int = 0
36     var kn int = n - 1
37     var iterasi int = 0
38
39     for kr <= kn && found == -1 {
40         iterasi++
41         med = (kr + kn) / 2
42         if X < T[med].nim {
43             kn = med - 1
44         } else if X > T[med].nim {
45             kr = med + 1
46         } else {
47             found = med
48         }
49     }
50     return found, iterasi
51 }
```

```

53 func main() {
54     var data arrMhs
55     n := 10
56
57     // Mengisi data secara manual
58     data = arrMhs{
59         {nama: "Ari", nim: "2201", kelas: "A", jurusan: "Informatika", ipk: 3.4},
60         {nama: "Budi", nim: "2203", kelas: "A", jurusan: "Informatika", ipk: 3.6},
61         {nama: "Cici", nim: "2202", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.5},
62         {nama: "Dina", nim: "2205", kelas: "A", jurusan: "Informatika", ipk: 3.3},
63         {nama: "Eko", nim: "2204", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.7},
64         {nama: "Fajar", nim: "2206", kelas: "C", jurusan: "Informatika", ipk: 3.1},
65         {nama: "Gita", nim: "2209", kelas: "C", jurusan: "Informatika", ipk: 3.8},
66         {nama: "Hana", nim: "2208", kelas: "B", jurusan: "Sistem Informasi", ipk: 3.2},
67         {nama: "Iwan", nim: "2207", kelas: "C", jurusan: "Informatika", ipk: 3.0},
68         {nama: "Joko", nim: "2210", kelas: "A", jurusan: "Informatika", ipk: 3.9},
69     }
70
71     // Pencarian Sequential Search berdasarkan nama
72     namaDicari := "Fajar"
73     idxSeq, iterSeq := SeqSearch_3(data, n, namaDicari)
74
75     fmt.Printf("Sequential Search - Cari nama '%s'\n", namaDicari)
76     if idxSeq != -1 {
77         fmt.Printf("Ditemukan di indeks: %d, Iterasi: %d\n", idxSeq, iterSeq)
78     } else {
79         fmt.Printf("Tidak ditemukan, Iterasi: %d\n", iterSeq)
80     }
81
82     // Urutkan data berdasarkan NIM untuk binary search
83     sort.Slice(data[:n], func(i, j int) bool {
84         return data[i].nim < data[j].nim
85     })
86
87     // Pencarian Binary Search berdasarkan NIM
88     nimDicari := "2206"
89     idxBin, iterBin := BinarySearch_3(data, n, nimDicari)
90
91     fmt.Printf("\nBinary Search - Cari NIM '%s'\n", nimDicari)
92     if idxBin != -1 {
93         fmt.Printf("Ditemukan di indeks: %d, Iterasi: %d\n", idxBin, iterBin)
94     } else {
95         fmt.Printf("Tidak ditemukan, Iterasi: %d\n", iterBin)
96     }
97 }

```

Output:

```

PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\O
uid2.go"
Sequential Search - Cari nama 'Fajar'
Ditemukan di indeks: 5, Iterasi: 6

Binary Search - Cari NIM '2206'
Ditemukan di indeks: 5, Iterasi: 3
PS C:\Users\levina\OneDrive\Documents\golang>

```

Penjelasan:

Program di atas menerapkan sequential search untuk mencari nama mahasiswa dan binary search untuk mencari NIM.

III. UNGUIDED

- UNGUIDED 1

Code:

```
SMT2 > Pertemuan9 > go 10311240041_Unguided1.go > ...
1 //BERTHA ADELA
2 //10311240041
3 package main
4 import "fmt"
5 func main() {
6     var pilihan int
7     var arr [100]int
8     pilihan = -1
9     jumlahMasukan := 0
10    jumlahSah := 0
11    var frekuensi [21]int
12
13    for pilihan != 0 {
14        fmt.Scan(&pilihan)
15        if pilihan != 0 {
16            arr[jumlahMasukan] = pilihan
17            jumlahMasukan = jumlahMasukan + 1
18
19            if pilihan >= 1 && pilihan <= 20 {
20                frekuensi[pilihan] = frekuensi[pilihan] + 1
21                jumlahSah = jumlahSah + 1
22            }
23        }
24    }
25
26    fmt.Printf("Suara masuk: %d\n", jumlahMasukan)
27    fmt.Printf("Suara sah: %d\n", jumlahSah)
28
29    for i := 1; i <= 20; i++ {
30        if frekuensi[i] > 0 {
31            fmt.Printf("%d: %d\n", i, frekuensi[i])
32        }
33    }
34}
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levi
nguided1.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
```

Penjelasan:

Program ini berguna untuk menghitung suara masuk, suara sah, dan menampilkan suara yang sah pada pilkart.

• UNGUIDED 2

Code:

```
SMT2 > Pertemuan9 > -go 103112400041_Unguided2.go > ...
1 //BERTHA ADELA
2 //103112400041
3 package main
4 import "fmt"
5 func main() {
6     var pilihan int
7     var arr [100]int
8     pilihan = -1
9     jumlahMasukan := 0
10    jumlahSah := 0
11    var frekuensi [21]int
12
13    for pilihan != 0 {
14        fmt.Scan(&pilihan)
15        if pilihan != 0 {
16            arr[jumlahMasukan] = pilihan
17            jumlahMasukan = jumlahMasukan + 1
18
19            if pilihan >= 1 && pilihan <= 20 {
20                frekuensi[pilihan] = frekuensi[pilihan] + 1
21                jumlahSah = jumlahSah + 1
22            }
23        }
24    }
25    fmt.Printf("Suara masuk: %d\n", jumlahMasukan)
26    fmt.Printf("Suara sah: %d\n", jumlahSah)
27
28    for i := 1; i <= 20; i++ {
29        frekuensi[i] = frekuensi[i]
30    }
31
32    max1, max2 := 0, 0
33    ketua, wakil := 0, 0
34
35    for i := 1; i <= 20; i++ {
36        if frekuensi[i] > max1 {
37            max2 = max1
38            wakil = ketua
39
40            max1 = frekuensi[i]
41            ketua = i
42        } else if frekuensi[i] > max2 {
43            max2 = frekuensi[i]
44            wakil = i
45        }
46    }
47
48    if ketua != 0 {
49        fmt.Printf("Ketua RT: %d\n", ketua)
50    }
51    if wakil != 0 {
52        fmt.Printf("Wakil Ketua: %d\n", wakil)
53    }
54 }
```


Output:



```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\nguided2.go"
7 19 3 2 78 3 1 -3 18 19 0
Suara masuk: 10
Suara sah: 8
Ketua RT: 3
Wakil Ketua: 19
PS C:\Users\levina\OneDrive\Documents\golang> 
```

Penjelasan:

Program ini menyatakan Ketua RT dan Wakil Ketua dari suara terbanyak.

- **UNGUIDED 3**

Code:

```
SMT2 > Pertemuan9 >  103112400041_Unguided3.go >  posisi

1  //BERTHA ADELA
2  //103112400041
3
4  package main
5  import "fmt"
6  const NMAX = 1000000
7  var data [NMAX]int
8  func main(){
9      var n, k int
10     fmt.Scanln(&n, &k)
11     isiArray(n)
12     posisi(n, k)
13     if posisi(n, k) != -1 {
14         fmt.Println(posisi(n,k))
15     } else {
16         fmt.Println("TIDAK ADA")
17     }
18 }

19 func isiArray(n int){
20     for i := 0; i < n; i++ {
21         fmt.Scan(&data[i])
22     }
23 }
24 func posisi(n, k int) int {
25     low := 0
26     high := n-1
27
28     for low <= high {
29         mid := (low + high)/2
30         if data[mid] == k {
31             return mid
32         } else if k < data[mid] {
33             high = mid - 1
34         } else {
35             low = mid + 1
36         }
37     }
38     return -1
39 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\nguided3.go"
12 534
1 3 8 16 32 123 323 323 534 543 823 999
8
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\nguided3.go"
12 535
1 3 8 16 32 123 323 323 534 543 823 999
TIDAK ADA
PS C:\Users\levina\OneDrive\Documents\golang> █
```

Penjelasan:

Program ini akan mencari apakah bilangan k tersebut ada dalam daftar bilangan yang diberikan? Jika ya, berikan indeksnya, jika tidak sebutkan "TIDAK ADA".

IV. KESIMPULAN

Sequential Search dan Binary Search sangat berguna untuk mencari suatu data yang ada di antara banyak data. Sequential Search digunakan untuk data acak maupunurut. Sequential Search bekerja dengan cara memastikan nilai yang sedang dicari dengan elemen-elemen, yaitu dari elemen pertama hingga elemen terakhir. Binary Search digunakan untuk data yang sudah urut menaik atau urut menurun. Binary Search bekerja dengan menandai kiri, tengah, dan kanan. Kemudian memastikan apakah tengah merupakan angka yang dicari, jika tidak maka program memilih antara kanan atau kiri sesuai dengan besar kecilnya angka yang sedang dicari. Cara tersebut akan terus diulang hingga angka tengah terakhir merupakan angka yang sedang dicari.

REFERENSI

MODUL 8 PENCARIAN NILAI ACAK PADA HIMPUNAN DATA