MODUL PRAKTIKUM

ALGORITMA PEMROGRAMAN

S1 INFORMATIKA

Channel Telegram
Informatics Lab

Published by school of computing

Telkom University

Fakultas Informatika
School of Computing
Telkom University

informatics lab

MODUL 11. PENCARIAN NILAI ACAK PADA HIMPUNAN DATA

Berikut ini merupakan materi lanjutan untuk kasus pencarian data. Selain pencarian nilai ekstrim, kita terkadang juga mencari suatu data yang lebih spesifik. Misalnya mencari mahasiswa dengan nama atau nim tertentu, mencari rumah dengan alamat tertentu, dan lainnya. Berbeda dengan pencarian nilai ekstrim, yang mana nilai yang dicari selalu ditemukan, maka pada kasus pencarian ini terdapat kemungkinan bahwa data yang dicari tidak ditemukan. Selain itu pada kasus pencarian ini akan diperkenal algoritma pencarian yang memanfaatkan keterurutan data.

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-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.

Berikut ini adalah contoh notasi algoritma **sekuensial search** sesuai dengan penjelasan di atas.

	Notasi Algoritma	Notasi dalam bahasa Go
1	found ← false	found = false
2	i ← 0	i = 0
3	while i < n and not found do	for i < n && !found {
4	found ← T[i] == X	found = T[i] == X
5	i ← i + 1	i = i + 1
6	endwhile	}

Adapun contoh potongan program pencarian sebuah string pada array of string adalah sebagai berikut ini:

```
..  ...
5   type arrStr [1234]string
..  ...
15
16  func SeqSearch_1(T arrStr, n int, X string) bool {
17  /* mengembalikan true apabila X ditemukan di dalam array T yang berisi n buah
18  teks, atau false apabila X tidak ditemukan */
19      var found bool = false
20      var j int = 0
21      for j < n && !found {
22          found = T[j] == X
23          j = j + 1
24      }
25      return found
26  }
```

Seperti penjelasan yang sudah diberikan sebelumnya, bahwa pada pencarian indeks atau lokasi dari data yang dicari lebih penting dibandingkan data itu sendiri. Oleh karena itu algoritma pencarian di atas bisa dimodifikasi untuk menghasilkan indeks dari data yang dicari. Selain itu perlu dipersiapkan juga sebuah nilai untuk menyatakan apabila pencarian data tidak ditemukan, biasanya -1 atau bilangan di luar indeks dari array yang valid.


```
..  ...
5   type arrStr [1234]string
..  ...
15
16  func SeqSearch_1(T arrStr, n int, X string) int {
17  /* mengembalikan indeks dari X apabila X ditemukan di dalam array T yang berisi
18  n buah teks, atau -1 apabila X tidak ditemukan */
19      var found int = -1
20      var j int = 0
21      for j < n && found == -1 {
22          if T[j] == X {
23              found = j
24          }
25          j = j + 1
26      }
27      return found
28  }
```

Variasi yang lain

```
..  ...
5   type arrStr [1234]string
..  ...
15
16  func SeqSearch_1(T arrStr, n int, X string) int {
17
```

```
/* mengembalikan indeks dari X apabila X ditemukan di dalam array T yang berisi
n buah teks, atau -1 apabila X tidak ditemukan */
    var j int = 0
    for j < n-1 && T[j] != X {
        j = j + 1
    }
    if T[j] == X {
        return j
    }else{
        return -1
    }
}
```

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 sebalah 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***. Berikut ini adalah contoh notasi algoritma *binary search* sesuai dengan penjelasan di atas.

	Notasi Algoritma	Notasi dalam bahasa Go
1	kr ← 0	kr = 0
2	kn ← n-1	kn = n-1
3	found ← false	found = false
4	while kr <= kn and not found do	for kr <= kn && !found {
5	med ← (kr+kn) div 2	med = (kr+kn) / 2
6	if a[med] < X then	if a[med] < X {
7	kr ← med + 1	kr = med + 1
8	else if a[med] > X then	} else if a[med] > X {
9	kn ← med − 1	kn = med
10	else	} else {
11	found ← true	found = true
12	endif	}
13	endwhile	}

Sebagai catatan algoritma *binary search* untuk array terurut membesar (*ascending*) akan berbeda dengan terurut mengecil (*descending*), sehingga algoritma ini tidak akan berjalan apabila terbalik.

Misalnya array terurut secara membesar, tetapi algoritma binary search yang digunakan untuk array terurut mengecil. Hal ini mengakibatkan pencarian binary search tidak akan berhasil. Selanjutnya bagaimana contoh algoritma pencarian data apabila dituliskan ke dalam suatu fungsi pencarian dan diketahui array terurut secara mengecil atau descending.

```
..  ...
5   type arrInt [4321]int
..  ...
15  func BinarySearch_1(T arrInt, n int, X string) bool {
16  /* mengembalikan true apabila X ditemukan di dalam array T yang berisi n buah
17  bilangan bulat terurut secara descending/mengecil, atau false apabila X tidak
18  ditemukan */
19      var found bool = false
20      var med int
21      var kr int = 0
22      var kn int = n - 1
23      for kr <= kn && !found {
24          med = (kr + kn) / 2
25          if X > T[med] {
26              kn = med - 1
27          }else if X < T[med] {
28              kr = med + 1
29          }else{
30              found = true
31          }
32      }
33      return found
34  }
35
```


Adapun variasi lain yang mengembalikan indeks dari data yang dicari adalah sebagai berikut:

```
..  ...
5   type arrInt [4321]int
..  ...
15  func BinarySearch_2(T arrInt, n int, X string) int {
16  /* mengembalikan indeks dari X apabila X ditemukan di dalam array T yang berisi
17  n buah bilangan bulat terurut secara descending/mengecil, atau -1 apabila X
18  tidak ditemukan */
19      var found int = -1
20      var med int
21      var kr int = 0
22      var kn int = n - 1
23      for kr <= kn && found == -1 {
24          med = (kr + kn) / 2
25          if X > T[med] {
26              kn = med - 1
27          }else if X < T[med] {
28              kr = med + 1
29          }else{
30              found = med
31          }
32      }
33      return found
34  }
```

Proses binary search akan berakhir apabila nilai `kr > kn` (data tidak ditemukan) atau `found` sudah tidak bernilai `false` atau `-1` (data ditemukan).

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 filed nama atau kategori yang lain selain nim. Solusinya adalah array harus diurutkan berdasarkan nim terlebih dahulu, atau gunakan algoritma *sequential search*.

Barikut adalah contoh apabila array mahasiswa terurut berdasarkan nim secara membesar (ascending) dan dilakukan pencarian menggunakan algoritma sekuensial dan biner.

```
..  ...
5   type mahasiswa struct {
..      nama, nim, kelas, jurusan string
..      ipk float64
..  }
..  type arrMhs [2023]mahasiswa
..  ...
15  func SeqSearch_3(T arrMhs, n int, X string) int {
16  /* mengembalikan indeks mahasiswa dengan nama X, atau -1 apabila tidak ditemukan
17  pada array T yang berisi n data mahasiswa */
18      var found int = -1
19      var j int = 0
20      for j < n && found == -1 {
21          if T[j].nama == X {
22              found = j
23          }
24          j = j + 1
25      }
26      return found
27  }
28
29  func BinarySearch_3(T arrMhs, n int, X string) int {
30  /* mengembalikan indeks mahasiswa dengan nim X, atau -1 apabila tidak ditemukan
31  pada array T yang berisi n data mahasiswa dan terurut membesar berdasarkan nim
32  */
33      var found int = -1
34      var med int
35      var kr int = 0
36      var kn int = n - 1
37      for kr <= kn && found == -1 {
38          med = (kr + kn) / 2
39          if X < T[med].nim {
40              kn = med - 1
41          }else if X > T[med].nim {
```

```
            kr = med + 1
        }else{
            found = med
        }
    }
    return found
}
```

11.4 Soal Latihan Modul 12

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

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

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 <= 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 dihitung dari awal data.
2	12 535 1 3 8 16 32 123 323 323 534 543 823 999	TIDAK ADA	