

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 12 & 13
PENGURUTAN DATA**



Disusun Oleh :

Siti Madina Halim Siregar / 2311102243

S1IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Larik (Array)

Array adalah kumpulan data untuk menyimpan item bertipe data sama, biasanya pada pengurutan data adalah data dengan tipe sama. Setiap data disimpan dalam alamat memori yang berbeda yang disebut elemen array. Komponen-komponen dari array antara lain nama array, nilai array, indeks array, jenis array [Situmorang, 2016]. Adapun proses yang dapat dilakukan dalam sebuah array antara lain memasukkan elemen data, mencari elemen data, menghapus elemen data, mencetak atau menampilkan hasil dari proses pengolahan data, menyisipkan elemen data, mencari posisi elemen data tertentu dan proses mengurutkan elemen data [Situmorang, 2016].

Pengurutan Data (Sorting)

Data terkadang berada dalam bentuk yang tidak berpola ataupun dengan pola tertentu yang diinginkan. Tidak ada algoritma terbaik untuk semua keadaan, kadang kala sebuah algoritma sangat efisien ketika jumlah datanya sedikit, namun kinerjanya menjadi berkurang ketika jumlah data ditambahkan atau meningkat. Meskipun memiliki kemampuan komputasi yang lebih tinggi, namun jika menggunakan algoritma yang kurang efisien, maka akan membutuhkan waktu lebih lama. Sehingga untuk memecahkan permasalahan diperlukan sebuah algoritma yang efektif dan efisien agar persoalan komputasi serta terbatasnya alokasi memori dapat diatasi.

Algoritma Insertion Sort

Algoritma insertion sort, adalah metode pengurutan dengan cara menyisipkan elemen data pada posisi yang tepat. Pencarian posisi yang tepat dilakukan dengan melakukan pencarian berurutan didalam barisan elemen, selama pencarian posisi yang tepat dilakukan pergeseran elemen [Sitorus and Sembiring, 2012]. Pengurutan insertion sort sangat mirip dengan konsep permainan kartu, bahwa setiap kartu disisipkan secara berurutan dari kiri ke kanan sesuai dengan besar nilai kartu tersebut, dengan syarat apabila sebuah kartu disisipkan pada posisi tertentu kartu yang lain akan bergeser maju atau mundur sesuai dengan besaran nilai yang dimiliki [Ramadhani, 2015].

Algoritma Selection Sort

Algoritma selection sort sering juga disebut dengan metode maksimum atau minimum. Metode maksimum karena didasarkan pada pemilihan data atau elemen maksimum sebagai dasar pengurutan. Konsepnya dengan memilih elemen maksimum kemudian mempertukarkan elemen maksimum tersebut dengan elemen paling akhir untuk urutan ascending dan elemen pertama untuk descending. Algoritma selection sort disebut juga dengan metode minimum karena didasarkan pada pemilihan elemen minimum sebagai dasar pengurutan. Konsepnya dengan memilih elemen minimum kemudian mempertukarkan elemen minimum dengan elemen paling akhir untuk urutan ascending dan elemen pertama untuk urutan descending. Proses yang dilakukan oleh algoritma selection sort adalah mengambil nilai terbesar dari susunan data dan menggantikannya dengan data yang paling kanan [Ramadhani, 2015].

II. GUIDED

Soal Studi Case

Hercules, preman terkenal seantero ibukota, memiliki kerabat di banyak daerah. Tentunya Hercules sangat suka mengunjungi semua kerabatnya itu.

Diberikan masukan nomor rumah dari semua kerabatnya di suatu daerah, buatlah program rumahkerabat yang akan menyusun nomor-nomor rumah kerabatnya secara terurut membesar menggunakan algoritma selection sort.

Masukan dimulai dengan sebuah integer ($0 < n < 1000$), banyaknya daerah kerabat Hercules tinggal. Isi n baris berikutnya selalu dimulai dengan sebuah integer m ($0 < m < 1000000$) yang menyatakan banyaknya rumah kerabat di daerah tersebut, diikuti dengan rangkaian bilangan bulat positif, nomor rumah para kerabat.

Keluaran terdiri dari n baris, yaitu rangkaian rumah kerabatnya terurut membesar di masing-masing daerah.

No	Masukan	Keluaran
1	3 5 2 1 7 9 13 6 189 15 27 39 75 133 3 4 9 1	1 2 7 9 13 15 27 39 75 133 189 1 4 9

Sourcecode

```
package main

import (
    "fmt"
)

// Fungsi untuk mengurutkan array menggunakan selection sort
func selectionSort(arr []int, n int) {
    for i := 0; i < n-1; i++ {
        idxMin := i
        for j := i + 1; j < n; j++ {
            // Cari elemen terkecil
            if arr[j] < arr[idxMin] {
                idxMin = j
            }
        }
        // Tukar elemen terkecil dengan elemen di posisi i
        arr[i], arr[idxMin] = arr[idxMin], arr[i]
    }
}
```

```

}

func main() {
    var n int
    fmt.Print("Masukkan jumlah daerah kerabat (n): ")
    fmt.Scan(&n)

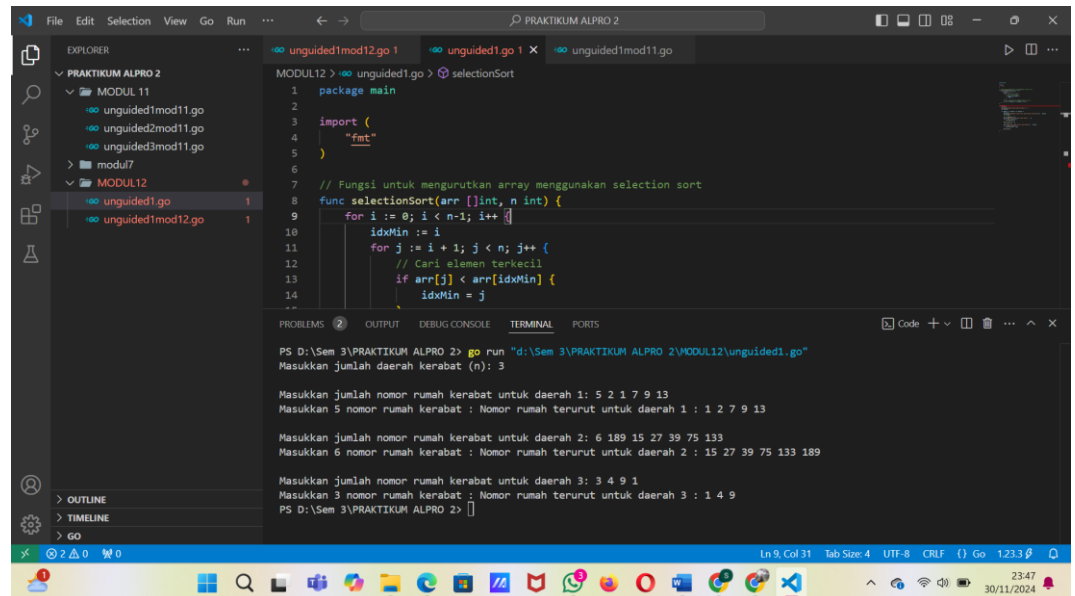
    for daerah := 1; daerah <= n; daerah++ {
        var m int
        fmt.Printf("\nMasukkan jumlah nomor rumah kerabat
untuk daerah %d: ", daerah)
        fmt.Scan(&m)

        arr := make([]int, m)
        fmt.Printf("Masukkan %d nomor rumah kerabat : ", m)
        for i := 0; i < m; i++ {
            fmt.Scan(&arr[i])
        }
        selectionSort(arr, m)

        fmt.Printf("Nomor rumah terurut untuk daerah %d : ",
daerah)
        for _, num := range arr {
            fmt.Printf("%d ", num)
        }
        fmt.Println()
    }
}

```

Screenshoot Output



The screenshot shows a Go IDE with a project named 'PRAKTIKUM ALPRO 2'. The Explorer panel on the left shows a directory structure with 'MODUL 11' and 'MODUL 12'. The main editor displays a Go file 'unguided1.go' with a selection sort implementation. The code includes a package declaration, an import for the 'fmt' package, and a function 'selectionSort' that sorts an array of integers using the selection sort algorithm. The function iterates through the array, finding the minimum element in the unsorted portion and swapping it with the first element of that portion. The output panel at the bottom shows the execution of the program, which prompts the user to enter the number of regions (n) and then the house numbers for each region. The output shows the sorted house numbers for three regions: 1: 5 2 1 7 9 13, 2: 6 189 15 27 39 75 133, and 3: 3 4 9 1.

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 // Fungsi untuk mengurutkan array menggunakan selection sort
8 func selectionSort(arr []int, n int) {
9     for i := 0; i < n-1; i++ {
10         idxMin := i
11         for j := i + 1; j < n; j++ {
12             // Cari elemen terkecil
13             if arr[j] < arr[idxMin] {
14                 idxMin = j
15             }
16         }
17         // Tukar elemen
18         arr[i], arr[idxMin] = arr[idxMin], arr[i]
19     }
20 }
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Sem 3\PRAKTIKUM ALPRO 2> go run "d:\Sem 3\PRAKTIKUM ALPRO 2\MODUL12\unguided1.go"

Masukkan jumlah daerah kerabat (n): 3

Masukkan jumlah nomor rumah kerabat untuk daerah 1: 5 2 1 7 9 13

Masukkan 5 nomor rumah kerabat : Nomor rumah terurut untuk daerah 1 : 1 2 7 9 13

Masukkan jumlah nomor rumah kerabat untuk daerah 2: 6 189 15 27 39 75 133

Masukkan 6 nomor rumah kerabat : Nomor rumah terurut untuk daerah 2 : 15 27 39 75 133 189

Masukkan jumlah nomor rumah kerabat untuk daerah 3: 3 4 9 1

Masukkan 3 nomor rumah kerabat : Nomor rumah terurut untuk daerah 3 : 1 4 9

PS D:\Sem 3\PRAKTIKUM ALPRO 2> []

Deskripsi Program

- Fungsi ini mengimplementasikan algoritma selection sort untuk mengurutkan array.
- Algoritma bekerja dengan:
 - Memilih elemen terkecil dalam array yang belum diurutkan.
 - Menukarkan elemen tersebut ke posisi yang sesuai.
- Parameter:
 - arr: array yang akan diurutkan.
 - n: panjang array.
- Meminta input jumlah rumah kerabat dalam suatu daerah (m).
- Membuat array berukuran m untuk menyimpan nomor rumah kerabat.
- Meminta pengguna mengisi nomor rumah.
- Mengurutkan array menggunakan fungsi selectionSort.
- Menampilkan nomor rumah yang telah diurutkan.

Soal Studi Case

Buatlah sebuah program yang digunakan untuk membaca data integer seperti contoh yang diberikan di bawah ini, kemudian diurutkan (menggunakan metoda insertion sort), dan memeriksa apakah data yang terurut berjarak sama terhadap data sebelumnya.

Masukan terdiri dari sekumpulan bilangan bulat yang diakhiri oleh bilangan negatif. Hanya bilangan non negatif saja yang disimpan ke dalam array.

Keluaran terdiri dari dua baris. Baris pertama adalah isi dari array setelah dilakukan pengurutan, sedangkan baris kedua adalah status jarak setiap bilangan yang ada di dalam array. "Data berjarak x" atau "data berjarak tidak tetap".

Contoh masukan dan keluaran

No	Masukan	Keluaran
1	31 13 25 43 1 7 19 37 -5	1 7 13 19 25 31 37 43 Data berjarak 6
2	4 40 14 8 26 1 38 2 32 -31	1 2 4 8 14 26 32 38 40 Data berjarak tidak tetap

Sourcecode

```
package main

import (
    "fmt"
)

// Fungsi untuk mengurutkan array menggunakan Insertion Sort
func insertionSort(arr []int, n int) {
    for i := 1; i < n; i++ {
        key := arr[i]
        j := i - 1

        // Geser elemen yang lebih besar dari key ke kanan
        for j >= 0 && arr[j] > key {
            arr[j+1] = arr[j]
            j--
        }
        arr[j+1] = key
    }
}
```

```

// Fungsi untuk memeriksa apakah selisih elemen array tetap
func isConstantDifference(arr []int, n int) (bool, int) {
    if n < 2 {
        return true, 0
    }

    difference := arr[1] - arr[0]
    for i := 1; i < n-1; i++ {
        if arr[i+1]-arr[i] != difference {
            return false, 0
        }
    }
    return true, difference
}

func main() {
    var arr []int
    var num int

    // Input data hingga bilangan negatif ditemukan
    fmt.Println("Masukkan data integer (akhiri dengan
bilangan negatif):")
    for {
        fmt.Scan(&num)
        if num < 0 {
            break
        }
        arr = append(arr, num)
    }

    n := len(arr)

    // Urutkan array menggunakan Insertion Sort
    insertionSort(arr, n)

    // Periksa apakah selisih elemen tetap
    isConstant, difference := isConstantDifference(arr, n)

    // Tampilkan hasil pengurutan
    fmt.Println("Array setelah diurutkan:")
    for _, val := range arr {
        fmt.Printf("%d ", val)
    }
    fmt.Println()
}

```



```

// Tampilkan status jarak
if isConstant {
    fmt.Printf("Data berjarak %d\n", difference)
} else {
    fmt.Println("Data berjarak tidak tetap")
}
}

```

Screenshoot Output

The screenshot shows a Go IDE with the following content:

Source Code (unguided2mod12.go):

```

1 package main
2
3 import (
4     "fmt"
5 )
6
7 // Fungsi untuk mengurutkan array menggunakan Insertion Sort
8 func insertionSort(arr []int, n int) {
9     for i := 1; i < n; i++ {
10        key := arr[i]
11        j := i - 1
12
13        // Geser elemen yang lebih besar dari key ke kanan
14        for j >= 0 && arr[j] > key {
15            arr[j+1] = arr[j]
16            j--
17        }
18        arr[j+1] = key
19    }
20 }
21
22 func main() {
23     // Test Case 1
24     arr1 := []int{31, 13, 25, 43, 1, 7, 19, 37, -5}
25     insertionSort(arr1, len(arr1))
26     fmt.Println("Array setelah diurutkan:")
27     for _, v := range arr1 {
28         fmt.Print(v, " ")
29     }
30     fmt.Println()
31     difference := arr1[1] - arr1[0]
32     isConstant := true
33     for i := 1; i < len(arr1); i++ {
34         if arr1[i] - arr1[i-1] != difference {
35             isConstant = false
36         }
37     }
38     if isConstant {
39         fmt.Println("Data berjarak", difference)
40     } else {
41         fmt.Println("Data berjarak tidak tetap")
42     }
43 }

```

Terminal Output:

```

PS D:\Sem 3\PRAKTIKUM ALPRO 2> go run "d:\Sem 3\PRAKTIKUM ALPRO 2\unguided2mod12.go"
Masukkan data integer (akhiri dengan bilangan negatif):
31 13 25 43 1 7 19 37 -5
Array setelah diurutkan:
1 7 13 19 25 31 37 43
Data berjarak 6
PS D:\Sem 3\PRAKTIKUM ALPRO 2> go run "d:\Sem 3\PRAKTIKUM ALPRO 2\unguided2mod12.go"
Masukkan data integer (akhiri dengan bilangan negatif):
4 40 14 8 26 1 38 2 32 -31
Array setelah diurutkan:
1 2 4 8 14 26 32 38 40
Data berjarak tidak tetap
PS D:\Sem 3\PRAKTIKUM ALPRO 2>

```

Deskripsi Program

- Fungsi ini memeriksa apakah selisih antar elemen dalam array yang telah diurutkan adalah konstan.
- Jika array berisi kurang dari dua elemen, dianggap selalu konstan dengan selisih 0.
- Hitung selisih pertama: $arr[1] - arr[0]$.
- Membandingkan selisih antar elemen Fungsi `isConstantDifference` dipanggil untuk memeriksa apakah selisih antar elemen konstan.
- Jika semua selisih sama, kembalikan `true` dan selisih tersebut.
- Jika ada selisih yang berbeda, kembalikan `false` dan 0.

III. UNGUIDED

1. Soal Studi Case

Belakangan diketahui ternyata Hercules itu tidak berani menyeberang jalan, maka selalu diusahakan agar hanya menyeberang jalan sesedikit mungkin, hanya diujung jalan. Karena nomor rumah sisi kiri jalan selalu ganjil dan sisi kanan jalan selalu genap, maka buatlah program kerabat dekat yang akan menampilkan nomor rumah mulai dari nomor yang ganjil lebih dulu terurut membesar dan kemudian menampilkan nomor rumah dengan nomor genap terurut mengecil.

Format Masukan masih persis sama seperti sebelumnya.

Keluaran terdiri dari n baris, yaitu rangkaian rumah kerabatnya terurut membesar untuk nomor ganjil, diikuti dengan terurut mengecil untuk nomor genap, di masing-masing daerah.

No	Masukan	Keluaran
1	3 5 2 1 7 9 13 6 189 15 27 39 75 133 3 4 9 1	1 13 12 8 2 15 27 39 75 133 189 8 4 2

Keterangan: Terdapat 3 daerah dalam contoh masukan. Baris kedua berisi campuran bilangan ganjil dan genap. Baris berikutnya hanya berisi bilangan ganjil, dan baris terakhir hanya berisi bilangan genap.

Petunjuk:

Waktu pembacaan data, bilangan ganjil dan genap dipisahkan ke dalam dua array yang berbeda, untuk kemudian masing-masing diurutkan tersendiri.

Atau, tetap disimpan dalam satu array, diurutkan secara keseluruhan.

Tetapi pada waktu pencetakan, mulai dengan mencetak semua nilai ganjil lebih dulu, kemudian setelah selesai cetaklah semua nilai genapnya.

Sourcecode

```
package main

import (
    "fmt"
    "sort"
)

func main() {
    var totalDaerah int
    fmt.Print("Masukkan jumlah daerah kerabat: ")
    fmt.Scan(&totalDaerah)

    for i := 1; i <= totalDaerah; i++ {
```

```

        var totalRumah int

        fmt.Printf("\nMasukkan jumlah rumah di daerah %d: ",
i)      fmt.Scan(&totalRumah)

        nomorRumah := make([]int, totalRumah)

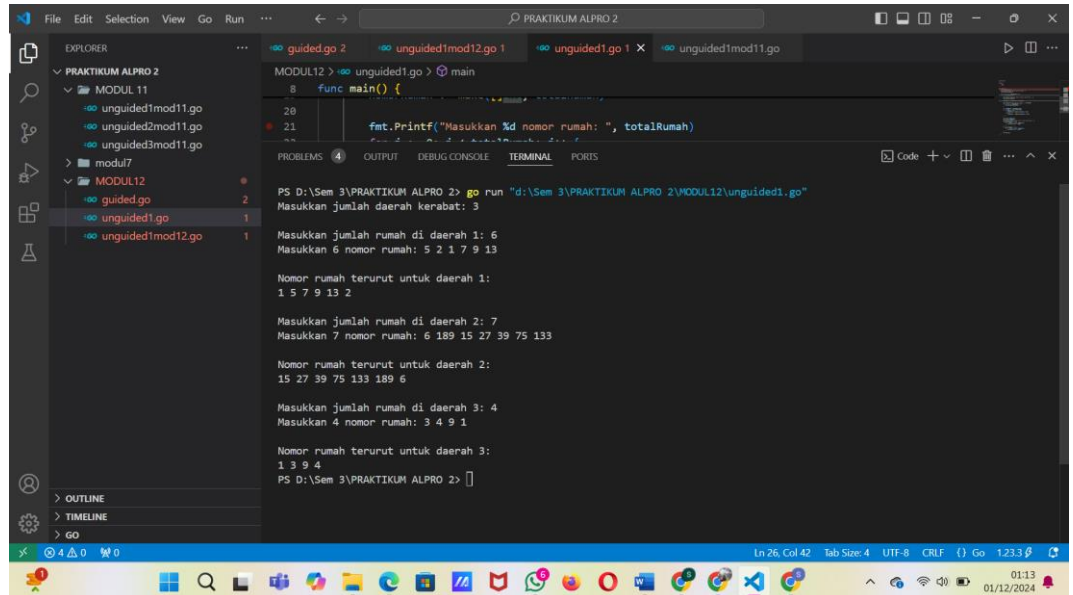
        fmt.Printf("Masukkan %d nomor rumah: ", totalRumah)
        for j := 0; j < totalRumah; j++ {
            fmt.Scan(&nomorRumah[j])
        }

        var rumahGanjil, rumahGenap []int
        for _, nomor := range nomorRumah {
            if nomor%2 == 0 {
                rumahGenap = append(rumahGenap, nomor)
            } else {
                rumahGanjil = append(rumahGanjil, nomor)
            }
        }

        sort.Ints(rumahGanjil)
        sort.Ints(rumahGenap)
        fmt.Printf("\nNomor rumah terurut untuk daerah
%d:\n", i)
        if len(rumahGanjil) > 0 {
            for _, nomor := range rumahGanjil {
                fmt.Printf("%d ", nomor)
            }
        }
        if len(rumahGenap) > 0 {
            for _, nomor := range rumahGenap {
                fmt.Printf("%d ", nomor)
            }
        }
        fmt.Println()
    }
}

```

Screenshoot Output



```
func main() {  
    20  
    21     fmt.Printf("Masukkan %d nomor rumah: ", totalRumah)  
  
    22  
    23  
    24  
    25  
    26  
    27  
    28  
    29  
    30  
    31  
    32  
    33  
    34  
    35  
    36  
    37  
    38  
    39  
    40  
    41  
    42  
    43  
    44  
    45  
    46  
    47  
    48  
    49  
    50  
    51  
    52  
    53  
    54  
    55  
    56  
    57  
    58  
    59  
    60  
    61  
    62  
    63  
    64  
    65  
    66  
    67  
    68  
    69  
    70  
    71  
    72  
    73  
    74  
    75  
    76  
    77  
    78  
    79  
    80  
    81  
    82  
    83  
    84  
    85  
    86  
    87  
    88  
    89  
    90  
    91  
    92  
    93  
    94  
    95  
    96  
    97  
    98  
    99  
    100  
}
```

PS D:\Sem 3\PRAKTIKUM ALPRO 2> go run "d:\Sem 3\PRAKTIKUM ALPRO 2\MODUL12\unguided1.go"

Masukkan jumlah daerah kerabat: 3

Masukkan jumlah rumah di daerah 1: 6

Masukkan 6 nomor rumah: 5 2 1 7 9 13

Nomor rumah terurut untuk daerah 1:

1 5 7 9 13 2

Masukkan jumlah rumah di daerah 2: 7

Masukkan 7 nomor rumah: 6 189 15 27 39 75 133

Nomor rumah terurut untuk daerah 2:

15 27 39 75 133 189 6

Masukkan jumlah rumah di daerah 3: 4

Masukkan 4 nomor rumah: 3 4 9 1

Nomor rumah terurut untuk daerah 3:

1 3 9 4

PS D:\Sem 3\PRAKTIKUM ALPRO 2>

Deskripsi Program

- Setiap nomor rumah diurutkan ke dalam array rumahGanjil atau rumahGenap.
- rumahGanjil dicetak lebih dulu, diikuti rumahGenap.
- Urutan ini sesuai petunjuk bahwa bilangan ganjil harus tampil lebih dulu.
- Urutan ini sesuai petunjuk bahwa bilangan ganjil harus tampil lebih dulu.
- Kode ini otomatis menyesuaikan input. Jika hanya ada bilangan ganjil atau genap, hanya bilangan yang relevan yang dicetak.
- Gunakan sort.Ints untuk mengurutkan keduanya secara independen.

2. Soal Studi Case

Kompetisi pemrograman yang baru saja berlalu diikuti oleh 17 tim dari berbagai perguruan tinggi ternama. Dalam kompetisi tersebut, setiap tim berlomba untuk menyelesaikan sebanyak mungkin problem yang diberikan. Dari 13 problem yang diberikan, ada satu problem yang menarik. Problem tersebut mudah dipahami, hampir semua tim mencoba untuk menyelesaikannya, tetapi hanya 3 tim yang berhasil. Apa sih problemnya?

"Median adalah nilai tengah dari suatu koleksi data yang sudah terurut. Jika jumlah data genap, maka nilai median adalah rerata dari kedua nilai tengahnya. Pada problem ini, semua data merupakan bilangan bulat positif, dan karenanya rerata nilai tengah dibulatkan ke bawah." Telkom University.

Buatlah program median yang mencetak nilai median terhadap seluruh data yang sudah terbaca, jika data yang dibaca saat itu adalah 0.

Masukan berbentuk rangkaian bilangan bulat. Masukan tidak akan berisi lebih dari 1000000 data, tidak termasuk bilangan 0. Data 0 merupakan tanda bahwa median harus dicetak, tidak termasuk data yang dicari mediannya. Data masukan diakhiri dengan bilangan bulat -5313.

Keluaran adalah median yang diminta, satu data per baris.

No	Masukan	Keluaran
1	7 23 11 0 5 19 2 29 3 13 17 0 -5313	11 12

Sampai bilangan 0 yang pertama, data terbaca adalah 7 23 11, setelah tersusun: 7 11 23, maka median saat itu adalah 11. Sampai bilangan 0 yang kedua, data adalah 7 23 11 5 19 2 29 3 13 17, setelah tersusun diperoleh: 2 3 5 7 11 13 17 19 23 29. Karena ada 10 data, genap, maka median adalah $(11+13)/2=12$.

Petunjuk:

Untuk setiap data bukan 0 (dan bukan marker -5313) simpan ke dalam array, Dan setiap kali menemukan bilangan 0, urutkanlah data yang sudah tersimpan dengan menggunakan metode Insertion sort dan ambil mediannya.

Sourcecode

```
package main

import (
    "fmt"
)

func insertionSort(arr []int, num int) []int {
    arr = append(arr, num)

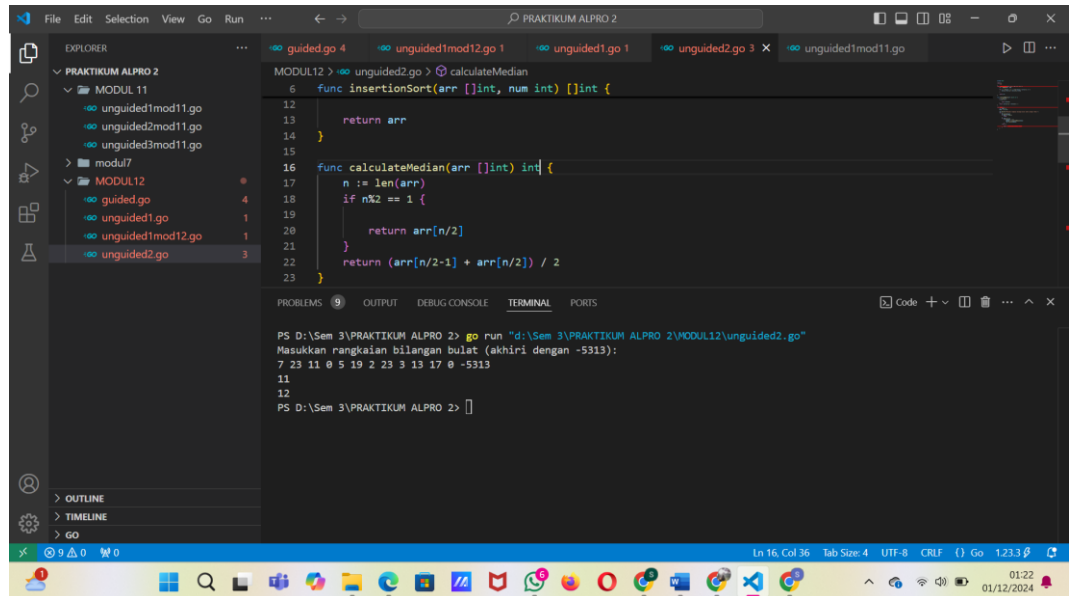
    for i := len(arr) - 1; i > 0 && arr[i] < arr[i-1]; i-- {
        arr[i], arr[i-1] = arr[i-1], arr[i]
    }

    return arr
}

func calculateMedian(arr []int) int {
    n := len(arr)
    if n%2 == 1 {
        return arr[n/2]
    }
    return (arr[n/2-1] + arr[n/2]) / 2
}

func main() {
    var input int
    data := []int{}
    fmt.Println("Masukkan rangkaian bilangan bulat (akhiri dengan -5313):")
    for {
        fmt.Scan(&input)
        if input == -5313 {
            break
        }
        if input == 0 {
            if len(data) > 0 {
                median := calculateMedian(data)
                fmt.Println(median)
            }
        } else {
            data = insertionSort(data, input)
        }
    }
}
```

Screenshoot Output



```
MODUL12 > go run "d:\Sem 3\PRAKTIKUM ALPRO 2\MODUL12\unguided2.go"
Masukkan rangkaian bilangan bulat (akhiri dengan -5313):
7 23 11 0 5 19 2 23 3 13 17 0 -5313
11
12
PS D:\Sem 3\PRAKTIKUM ALPRO 2>
```

Deskripsi Program

- Setiap kali bilangan baru (bukan 0 atau -5313) dibaca, bilangan tersebut dimasukkan ke array dengan posisi yang sesuai menggunakan metode insertion sort.
- Array selalu terurut setelah elemen baru dimasukkan.
- Saat 0 dibaca, median dihitung:
- Jika jumlah elemen ganjil, median adalah elemen tengah.
- Jika jumlah elemen genap, median adalah rata-rata dua elemen tengah (dibulatkan ke bawah).
- Median dicetak setelah dihitung.
- Membaca input dihentikan saat marker -5313 ditemukan.

3. Soal Studi Case

Sebuah program perpustakaan digunakan untuk mengelola data buku di dalam suatu perpustakaan. Misalnya terdefinisi struct dan array seperti berikut ini:

```
const nMax : integer = 7919
type Buku = <
    id, judul, penulis, penerbit : string
    eksemplar, tahun, rating : integer >

type DaftarBuku = array [ 1..nMax] of Buku
Pustaka : DaftarBuku
nPustaka: integer
```

Masukan terdiri dari beberapa baris. Baris pertama adalah bilangan bulat N yang menyatakan banyaknya data buku yang ada di dalam perpustakaan. N baris berikutnya, masing-masingnya adalah data buku sesuai dengan atribut atau field pada struct. Baris terakhir adalah bilangan bulat yang menyatakan rating buku yang akan dicari.

Keluaran terdiri dari beberapa baris. Baris pertama adalah data buku terfavorit, baris kedua adalah lima judul buku dengan rating tertinggi, selanjutnya baris terakhir adalah data buku yang dicari sesuai rating yang diberikan pada masukan baris terakhir.

Lengkapi subprogram-subprogram dibawah ini, sesuai dengan I.S. dan F.S yang diberikan.

```
procedure DaftarkanBuku(in/out pustaka : DaftarBuku, n : integer)
{I.S. sejumlah n data buku telah siap para piranti masukan
 F.S. n berisi sebuah nilai, dan pustaka berisi sejumlah n data buku}

procedure CetakTerfavorit(in pustaka : DaftarBuku, in n : integer)
{I.S. array pustaka berisi n buah data buku dan belum terurut
 F.S. Tampilan data buku (judul, penulis, penerbit, tahun)
terfavorit, yaitu memiliki rating tertinggi}

procedure UrutBuku( in/out pustaka : DaftarBuku, n : integer )
{I.S. Array pustaka berisi n data buku
 F.S. Array pustaka terurut menurun/mengecil terhadap rating.
Catatan: Gunakan metoda Insertion sort}

procedure Cetak5Terbaru( in pustaka : DaftarBuku, n integer )
{I.S. pustaka berisi n data buku yang sudah terurut menurut rating
 F.S. Laporan 5 judul buku dengan rating tertinggi
Catatan: Isi pustaka mungkin saja kurang dari 5}

procedure CariBuku(in pustaka : DaftarBuku, n : integer, r : integer )
{I.S. pustaka berisi n data buku yang sudah terurut menurut rating
 F.S. Laporan salah satu buku (judul, penulis, penerbit, tahun,
eksemplar, rating) dengan rating yang diberikan. Jika tidak ada buku
dengan rating yang ditanyakan, cukup tuliskan "Tidak ada buku dengan
rating seperti itu". Catatan: Gunakan pencarian biner/belah dua.}
```


Sourcecode

```
package main

import (
    "bufio"
    "fmt"
    "os"
    "strconv"
    "strings"
)

type Buku struct {
    ID        int
    Judul     string
    Penulis   string
    Penerbit  string
    Eksemplar int
    Tahun     int
    Rating    int
}

type DaftarBuku []Buku

func DaftarkanBuku(pustaka *DaftarBuku, n int) {
    reader := bufio.NewReader(os.Stdin)
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan data buku %d (format: ID Judul\nPenulis Penerbit Eksemplar Tahun Rating):\n", i+1)
        data, _ := reader.ReadString('\n')
        data = strings.TrimSpace(data)
        fields := strings.Split(data, " ")

        id, _ := strconv.Atoi(fields[0])
        eksemplar, _ := strconv.Atoi(fields[4])
        tahun, _ := strconv.Atoi(fields[5])
        rating, _ := strconv.Atoi(fields[6])

        buku := Buku{
            ID:        id,
            Judul:    fields[1],
            Penulis:   fields[2],
            Penerbit:  fields[3],
            Eksemplar: eksemplar,
            Tahun:     tahun,
            Rating:    rating,
        }
    }
}
```

```

        *pustaka = append(*pustaka, buku)
    }
}
func CetakTerfavorit(pustaka DaftarBuku, n int) {
    if n == 0 {
        fmt.Println("Tidak ada data buku.")
        return
    }

    terfavorit := pustaka[0]
    for _, buku := range pustaka {
        if buku.Rating > terfavorit.Rating {
            terfavorit = buku
        }
    }

    fmt.Println("Buku terfavorit:")
    fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s, Tahun: %d\n",
        terfavorit.Judul, terfavorit.Penulis,
        terfavorit.Penerbit, terfavorit.Tahun)
}
func UrutBuku(pustaka *DaftarBuku, n int) {
    for i := 1; i < n; i++ {
        key := (*pustaka)[i]
        j := i - 1
        for j >= 0 && (*pustaka)[j].Rating < key.Rating {
            (*pustaka)[j+1] = (*pustaka)[j]
            j--
        }
        (*pustaka)[j+1] = key
    }
}
func Cetak5Terbaru(pustaka DaftarBuku, n int) {
    fmt.Println("Lima buku dengan rating tertinggi:")
    limit := 5
    if n < 5 {
        limit = n
    }
    for i := 0; i < limit; i++ {
        fmt.Printf("%s\n", pustaka[i].Judul)
    }
}
func CariBuku(pustaka DaftarBuku, n int, r int) {

```

```

    low, high := 0, n-1

    for low <= high {
        mid := (low + high) / 2
        if pustaka[mid].Rating == r {
            buku := pustaka[mid]
            fmt.Println("Buku ditemukan:")
            fmt.Printf("Judul: %s, Penulis: %s, Penerbit: %s, Tahun: %d, Eksemplar: %d, Rating: %d\n",
                buku.Judul, buku.Penulis, buku.Penerbit,
                buku.Tahun, buku.Eksemplar, buku.Rating)
            return
        } else if pustaka[mid].Rating < r {
            high = mid - 1
        } else {
            low = mid + 1
        }
    }

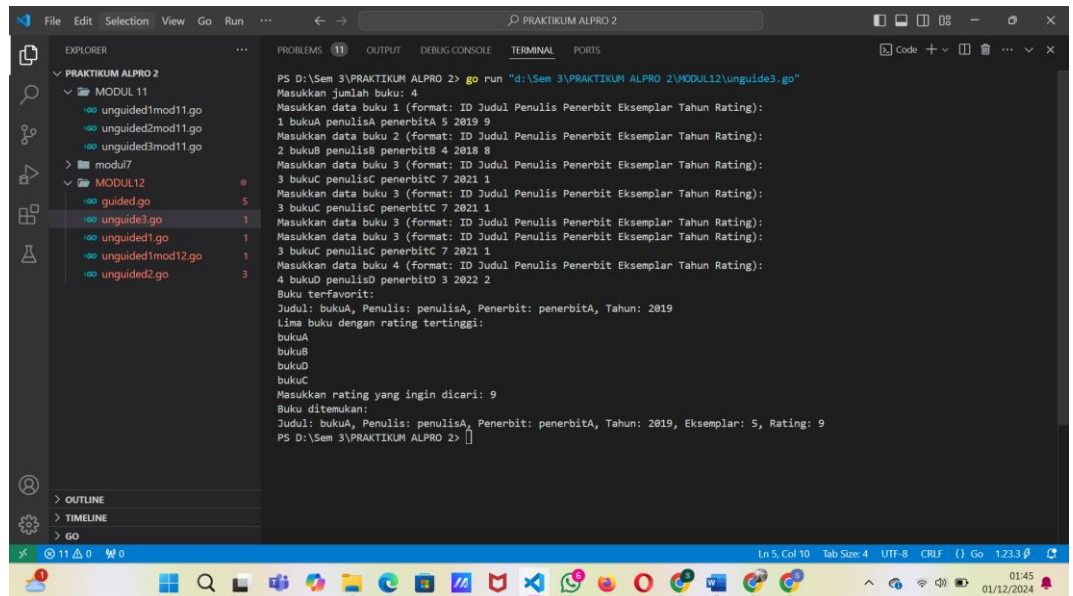
    fmt.Println("Tidak ada buku dengan rating seperti itu.")
}

func main() {
    var pustaka DaftarBuku
    var n, rating int

    reader := bufio.NewReader(os.Stdin)
    fmt.Print("Masukkan jumlah buku: ")
    input, _ := reader.ReadString('\n')
    n, _ = strconv.Atoi(strings.TrimSpace(input))
    DaftarkanBuku(&pustaka, n)
    CetakTerfavorit(pustaka, n)
    UrutBuku(&pustaka, n)
    Cetak5Terbaru(pustaka, n)
    fmt.Print("Masukkan rating yang ingin dicari: ")
    input, _ = reader.ReadString('\n')
    rating, _ = strconv.Atoi(strings.TrimSpace(input))
    CariBuku(pustaka, n, rating)
}

```

Screenshoot Output



```
PS D:\Sem 3\PRAKTIKUM ALPRO 2> go run "d:\Sem 3\PRAKTIKUM ALPRO 2\MODUL12\unguide3.go"
Masukkan jumlah buku: 4
Masukkan data buku 1 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
1 bukuA penulisA penerbitA 5 2019 9
Masukkan data buku 2 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
2 bukuB penulisB penerbitB 4 2018 8
Masukkan data buku 3 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
3 bukuC penulisC penerbitC 7 2021 1
Masukkan data buku 3 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
3 bukuC penulisC penerbitC 7 2021 1
Masukkan data buku 3 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
3 bukuC penulisC penerbitC 7 2021 1
Masukkan data buku 4 (format: ID Judul Penulis Penerbit Eksemplar Tahun Rating):
4 bukuD penulisD penerbitD 3 2022 2
Buku terfavorit:
Judul: bukuA, Penulis: penulisA, Penerbit: penerbitA, Tahun: 2019
Line buku dengan rating tertinggi:
bukuA
bukuB
bukuD
bukuC
Masukkan rating yang ingin dicari: 9
Buku ditemukan:
Judul: bukuA, Penulis: penulisA, Penerbit: penerbitA, Tahun: 2019, Eksemplar: 5, Rating: 9
PS D:\Sem 3\PRAKTIKUM ALPRO 2> ]
```

Deskripsi Program

- **DaftarkanBuku:**
 - Membaca data buku dari input dan menyimpannya ke dalam array pustaka.
- **CetakTerfavorit:**
 - Mencetak buku dengan rating tertinggi. Prosedur ini mencari buku dengan rating tertinggi secara linear.
- **UrutBuku:**
 - Mengurutkan array pustaka secara menurun berdasarkan rating menggunakan Insertion Sort.
- **Cetak5Terbaru:**
 - Mencetak judul dari 5 buku dengan rating tertinggi. Jika buku kurang dari 5, akan mencetak semuanya.
- **CariBuku:**
 - Mencari buku berdasarkan rating menggunakan Binary Search pada array yang sudah diurutkan.

Kesimpulan

Algoritma insertion sort, adalah metode pengurutan dengan cara menyisipkan elemen data pada posisi yang tepat. Pencarian posisi yang tepat dilakukan dengan melakukan pencarian berurutan didalam barisan elemen, selama pencarian posisi yang tepat dilakukan pergeseran elemen. Algoritma selection sort sering juga disebut dengan metode maksimum atau minimum. Metode maksimum karena didasarkan pada pemilihan data atau elemen maksimum sebagai dasar pengurutan.

Daftar Pustaka

<https://media.neliti.com/media/publications/414701-none-931dcaa.pdf>